Thermostabilization and general protein design

This tutorial will walk through thermostabilization of the bacterial amino acid transporter protein LeuT. As the name suggests, thermostabilization refers to the process of increasing the stability of a given backbone conformation, while design more broadly refers to the optimization of a protein's sequence given the structure. We will optimize the sequence of two conformations of this protein and compare the results to both each other and to the native sequence.

NOTE: Because LeuT is a membrane protein, there are certain steps in this tutorial that are specific to membrane proteins. To apply this tutorial to a soluble protein, you may skip these steps.

Two models have been provided for you:

- 2A65_full.pdb, which represents an outward-facing conformation of LeuT that crystallizes easily in the native sequence, and
- 3TT3_full.pdb, which represents an inward-facing conformation of LeuT that can only be crystallized with four thermostabilizing mutations (Y268A, K288A, T354V, S355A). **These mutations have been removed from this model.**

Although both structures can be downloaded from RCSB, some loops were not crystallographically resolved. These have been added into the provided models using RosettaCM.

Part I: Set up input files:

NOTE: All files are provided for these tutorials. RosettaCM was used to add missing residues to the two models of LeuT we are using (PDB: 2A65 and PDB: 3TT3). It may be helpful, however, to go through these steps using the RosettaCM tutorial.

1. Change your current directory to thermostabilization, then create a directory called my_files. This will be your working directory.

   ```
   cd ~/rosetta_workshop/protein_design/thermostabilization
   mkdir my_files
   cd my_files
   ```

2. Copy the protein models to your directory. These are the models/templates we will be using for our design analysis (alternatively, you may download PDB models 2A65 and 3TT3 from rcsb.org and fill in the missing loops using RosettaCM; since the sequences are otherwise identical, you do not need to generate a threaded model to do this).

   ```
   cp ../input_files/2A65_full.pdb .
   cp ../input_files/3TT3_full.pdb .
   ```

3. Rosetta accounts for the position of the membrane by using a span file. This allows it to place and optimize the position of a "dummy residue" that will simulate a 30 angstrom-thick plane reflecting the unique properties of the membrane environment. Get a span file for LeuT. A prepared span file is located in the input directory:

   ```
   cp ../input_files/LeuT.span .
   ```

   a. To obtain a span file from scratch, you must first obtain the FASTA file from the protein model (you can also go to the RCSB website and search for the protein using its PDB ID: https://www.rcsb.org/structure/2a65. Note that the fasta file associated with the inward-facing, 3TT3 model will have the four thermostabilizing mutations). Also note that this script will probably throw an error, but nonetheless correctly generate a fasta file. **If you are interested in running this script on your personal computer, by sure to have BioPython installed.**

```
python ~/rosetta_workshop/rosetta/tools/protein_tools/scripts/get_fasta_from_pdb.py \
  2A65_full.pdb A LeuT_WT.fasta
```

The file has been provided:

```
cp ../input_files/LeuT.fasta .
```

b. Go to the Stockholm Bioinformatics Center website (http://octopus.cbr.su.se) and provide the FASTA sequence. The OCTOPUS algorithm is among the most effective sequence-based approaches to determining membrane protein topology. Click "Submit OCTOPUS". When the results are available (<60 seconds), follow the link for the **OCTOPUS topology file (.txt)**. Save the contents as **LeuT.octopus** in your working directory.

c. Convert the octopus file into a span file using the available script. Be sure to pipe the output to the file of interest.

```
perl ~/rosetta_workshop/rosetta/tools/membrane_tools/octopus2span.pl LeuT.octopus > LeuT.span
```

Part II: Transform and Score the wildtype outward-facing and inward-facing conformations.

1. Options files allow us to avoid typing in our command line options every time we run the protocol. If we are refining a protocol, they also allow us to maintain a record of what we have tried in the past. Prepare an options file titled **score.options** with the following command line options:

```
-score:weights mpframework_smooth_fa_2012.wts
-in:membrane
-mp:setup:spanfiles LeuT.span
-mp:scoring:hbond true
-mp:transform:optimize_embedding true
-mp:setup:transform_into_membrane true
```

This particular file provides membrane-specific flags for scoring membrane proteins (scoring soluble proteins is, not surprisingly, easier). This file has already been provided:

```
cp ../scripts/score.options .
```

NOTE: the default score function is **ref2015.wts**, which is used for soluble proteins. If the -score:weights is not declared, ref2015.wts is used.

2. Transform the proteins into the membrane and evaluate the outward-facing and inward-facing conformations of wildtype LeuT. The options file describes the parameters for scoring in a membrane environment.

```
~/rosetta_workshop/rosetta/main/source/bin/score_jd2.linuxgccrelease @score.options \
  -in:file:s 2A65_full.pdb 3TT3_full.pdb -out:pdb
```

3. Open the pdb files, and look to see if the proteins have been transposed into the membrane. The "membrane residue" (MEM) should be near the center of the proteins.

```
pymol 2A65_full_0001.pdb 3TT3_full_0001.pdb
```

4. Open the score file. Which structure is predicted to be more energetically favorable? Experimentally, the outward-facing conformation (PDB: 2A65) is more stable and easier to crystallize.

```
gedit score.sc &
```

5. Before we move on, we'll want to rename our transformed pdb files.

```
mv 2A65_full_0001.pdb 2A65_in_mem.pdb
mv 3TT3_full_0001.pdb 3TT3_in_mem.pdb
```

Part III: Reintroduce thermostabilizing mutations

1. Rosetta uses resfiles to guide protein design. These determine which residues will be "designed" (i.e. will be swapped out for other residue types) and which will be left untouched. The introduction of the site-directed mutations that were used to thermostabilize the inward-facing structure will be achieved using the following resfile:

   ```
   NATAA
   START
   268 A PIKAA A
   288 A PIKAA A
   354 A PIKAA V
   355 A PIKAA A
   ```

   In resfile syntax, default behavior is defined prior to the **"START"** line. In this case, **"NATAA"** allows side chains to be repacked, even if they are ineligible for redesign (**"NATRO"** - natural rotamers - would instruct Rosetta to leave the side chains untouched by default). The four residues we want to mutate, and the chains to which they belong, are explicitly defined, and **"PIKAA"** instructs Rosetta to replace them with one of the following residue identities. Since the only option is either alanine (A) or valine (V), depending on the residue, the mutagenesis achieves the desired result. Further documentation for resfile syntax is available at this link (https://www.rosettacommons.org/docs/latest/rosetta_basics/file_types/resfiles).

   Copy this file to your working directory:

   ```
   cp ../input_files/mutations_a.resfile .
   ```

2. Prepare an options file titled **introduce_mutations_a.options** with the following command line options:

   ```
   -parser:protocol introduce_mutations_a.xml
   -score:weights mpframework_smooth_fa_2012.wts
   -mp:setup:spanfiles LeuT.span
   -mp:scoring:hbond true
   ```

   This options file refers to the XML file **introduce_mutations_a.xml**, which is a script detailing the modeling steps and references the resfile above. It contains four "movers" that introduce specific changes to the input model:

   - **AddMembraneMover**, which defines the protein as a membrane protein,
   - **MembranePositionFromTopologyMover**, which defines the membrane position from the span file,
   - **PackRotamersMover**, which repacks side chains and introduces mutations of interest as dictated in the resfile, and
   - **FastRelax**, which minimizes the energy of the protein and repacks the side chains.

   NOTE: To run this protocol on a soluble protein, remove the first two Movers in the XML file and the bottom three options in the options file. You will also need to change the score function to **ref2015.wts**.

   Both files have already been provided:

   ```
   cp ../scripts/introduce_mutations_a.options .
   cp ../scripts/introduce_mutations_a.xml .
   ```

3. Run the protocol, which should take 10 to 15 minutes for both proteins. The output models will be named 2A65_in_mem_a_0001.pdb and 3TT3_in_mem_a_0001.pdb.

   ```
   ~/rosetta_workshop/rosetta/main/source/bin/rosetta_scripts.linuxgccrelease \
     @introduce_mutations_a.options -in:file:s 2A65_in_mem.pdb 3TT3_in_mem.pdb -out:suffix _a
   ```

4. Look at the output models in PyMol to compare the effect of introducing the mutation to the protein:

   ```
   pymol 2A65_in_mem_a_0001.pdb 2A65_in_mem.pdb &
   pymol 3TT3_in_mem_a_0001.pdb 3TT3_in_mem.pdb &
   ```

Part IV: Identify new thermostabilizing mutations

1. In this exercise we will be introducing new side chains at the same four sites. The protocol will be the same as the one used above, but the resfile will be slightly different, to allow the residues to be changed to whichever identity Rosetta determines is most energetically favorable:

   ```
   NATAA
   START
   268 A ALLAA
   288 A ALLAA
   354 A ALLAA
   355 A ALLAA
   ```

   NOTE: In the wet lab, we generally want to avoid introducing cysteines into our construct unless we are interested in deliberately introducing disulfide bonds. To avoid introducing cysteines, replace **"ALLAA"** with **"ALLAAxc"**; this limits design to the other nineteen residues.

   Copy the provided resfile to you working directory:

   ```
   cp ../input_files/mutations_b.resfile .
   ```

2. The XML and options files need to be changed to use this resfile; everything else is unchanged. You may copy and edit the existing XML file, or you may copy the XML file provided in the input directory:

   ```
   cp ../scripts/introduce_mutations_b.xml .
   cp ../scripts/introduce_mutations_b.options .
   ```

3. Run the protocol on the inward-facing model **only**. We will look at the mutations Rosetta inserts and introduce them into the outward-facing model.

   ```
   ~/rosetta_workshop/rosetta/main/source/bin/rosetta_scripts.linuxgccrelease \
     @introduce_mutations_b.options -in:file:s 3TT3_in_mem.pdb -out:suffix _b
   ```

   NOTE: To explore the full design space regarding just these four residues, which consists of 20^4 (160,000) possible sequences, you will want to make multiple models. It may also be helpful to iterate between designing the models and relaxing them to fully explore the design space. To generate N output models use the option "-nstruct N". Alternatively, we have provided 20 models in the directory below:

   ```
   cd ../output_files/mutants_b/
   ```

4. Compare the scores by opening score.sc. You may identify the lowest-energy structure using the sort command:

   ```
   sort -nk 2 score_b.sc
   ```

   Look at the PDB file by reading it directly or opening the file in PyMol. What are the residue identities Rosetta chose?

5. Write a new resfile to introduce those residue identities into the outward-facing model. Modify the resfile above (mutations_b.resfile) and change the amino acid identity of specific residues to those found in your protein. Save the file as mutations_3TT3_b.resfile. In our dataset of 20 models, we created a resfile based on the lowest scoring model. Copy the resfile and XML and options files provided:

   ```
   cp ../output_files/mutants_b/mutations_3TT3_b.resfile .
   cp ../output_files/mutants_b/introduce_mutations_3TT3_b.xml .
   cp ../output_files/mutants_b/introduce_mutations_3TT3_b.options .
   ```

6. With your modified resfile, run the protocol with the options file introduce_mutations_b.options:

   ```
   ~/rosetta_workshop/rosetta/main/source/bin/rosetta_scripts.linuxgccrelease \
     @introduce_mutations_3TT3_b.options -in:file:s 2A65_in_mem.pdb -out:suffix _b
   ```

Part V: Full redesign

1. Here we redesign the entire protein. Every residue will be open to redesign. As a result, identification of the lowest-energy mutant requires a much larger sample set (in practice, thousands or tens of thousands). The resfile, however, is simpler:

   ```
   ALLAA
   START
   ```

   Again, the default is specified by what comes above the **"START"** line. In this case, every residue is redesigned. You may copy this file to your working directory:

   ```
   cp ../input_files/mutations_c.resfile
   ```

2. Copy the XML and options file provided to your working directory:

   ```
   cp ../scripts/introduce_mutations_c.xml .
   cp ../scripts/introduce_mutations_c.options .
   ```

   Here we introduce a new flag in the options file, **-linmem_ig 10**, this flag limits the amount of memory used to store the interaction energy between rotamer pairs. If this flag is left off we may run out of memory performing a full redesign.

3. Run the protocol on the inward-facing state only, however this step will take ~2 hours, therefore it's recommended to use the files we've provided, skip to step 5:

   ```
   ~/rosetta_workshop/rosetta/main/source/bin/rosetta_scripts.linuxgccrelease \
     @introduce_mutations_c.options -in:file:s 3TT3_in_mem.pdb -out:suffix _c
   ```

4. How much of the native sequence is left in the redesigned model? Check the sequence recovery (to use this protocol, we must first generate a list of native structures and redesigned structures):

   ```
   echo 3TT3_in_mem.pdb > natives.list
   echo 3TT3_in_mem_c_0001.pdb > designs.list
   ~/rosetta_workshop/rosetta/main/source/bin/sequence_recovery.linuxgccrelease \
     -native_pdb_list natives.list -redesign_pdb_list designs.list
   ```

   Check the output files (**sequencerecovery.txt** and **submatrix.txt**). In previous benchmarking studies, we found that Rosetta favors mutating alpha-helical residues to leucine.

5. We have generated twenty models in the directory below:

   ```
   cd ../output_files/mutants_c/
   ```

6. Determine the best model from the score file in this directory using the **sort** command above. From the best-scoring model, obtain a FASTA file as we described above (the script will throw an error, even if it completes correctly).

   ```
   python ~/rosetta_workshop/rosetta/tools/protein_tools/scripts/get_fasta_from_pdb.py \
     3TT3_in_mem_c_0016.pdb A best_model.fasta
   ```

7. Run the following command to convert the FASTA into a resfile. Note that you will have to manually add **"START"** to the top of your resfile. (The command **tail -1** prints the last line of the file, in this case the entire amino acid sequence. This is then piped to **grep**, which splits the line into a list of characters. This list is then piped to **awk**, which prints the row number, followed by **"A PIKAA"**, followed by the character. Note that a typical fasta will have the sequence split across multiple lines. However, the protein sequence is on a single line in fasta files generated using get_fasta_from_pdb.py.)

```
tail -1 best_model.fasta | grep -o . | awk '$1=(FNR FS "A PIKAA " $1)' > mutations_c_0016.resfile
```

(You may also copy the file **mutations_c_0016.resfile** from the same directory)

8. Make a copy of the options file and XML file and change the resfile to the file above, or copy the appropriate files from the input files directory to your working directory:

```
cd ../../my_files/
cp ../output_files/mutants_c/introduce_mutations_c_0016.options .
cp ../output_files/mutants_c/introduce_mutations_c_0016.xml .
cp ../output_files/mutants_c/mutations_c_0016.resfile .
```

9. Run the protocol on the outward-facing structure:

```
~/rosetta_workshop/rosetta/main/source/bin/rosetta_scripts.linuxgccrelease \
  @introduce_mutations_c_0016.options -in:file:s 2A65_in_mem.pdb -out:suffix _d
```

10. Compare the score of the designed inward-facing and outward-facing structures. Which structure is more energetically favorable?

Part VI: Full redesign in an aqueous environment

1. If there is still time left, try redesigning LeuT without the membrane-specific options. The instructions are otherwise identical to those in part V.