

Comparative Modeling: Loop Building and Clustering

Rosetta Comparative Modeling Tutorial

Bold text means that these files and/or this information is provided.

Italicized text means that this material will NOT be conducted during the workshop

fixed width text means you should type the command into your terminal

Note: If you would like to manually create files instead of using those that are provided, (e.g., input files), write them to a new directory! (`mkdir ~/my_model`)

1. Prepare the input files

Prepare FASTA file of the target sequence:

The **1u19A.fasta** file is already provided in the
~/rosetta_workshop/tutorials/protein_modeling/input_model/
directory.

i. Get the sequence in FASTA format from NCBI.

1. Go to www.ncbi.nlm.nih.gov/protein/.
2. Type “1u19A” in the search bar.
3. Click the “FASTA” link to see the protein sequence in FASTA format.
4. Copy all the sequence information, including the line beginning with “>”, into a file called 1u19A.fasta.
5. Replace the first line of the file with “>1u19A”
6. Remove the N- and C-terminal regions (delete from the beginning of the XMNGTEGPNFYVPFSNKTGVVRSPFEAPQYYLAE and from the end of the sequence KNPLGDDEASTTVSKTETSQVAPA, so you are left with the sequence as shown below. We will only be constructing a comparative model of the transmembrane region of the protein.

```
>1u19A
PWQFSMLAAYMFLIMLGFPINFLTLYVTVQHKKLRTPLNYYILLNLAVADLFMVFGGFTTLYTSLHGYFVF
GPTGCNLEGFFATLGGEIALWSLVLAIERVYVVCKPMSNFRFGENHAIMGVAFTVMALACAAPPLVGWS
RYIPEGMQCSCGIDYYTPHEETNNESFVIYMFVVHFIPLIVIFFCYGQLVFTVKEAAQQQESATTQKAEKE
VTRMVIIMVIAFLICWLPYAGVAFYIFTHQGSDFGPIFMTIPAFFAKTSAVYNPVIYIMMNKQFRNCMVTTLCCG
```

ii. Move the FASTA file to your input directory.

1. `mv 1u19A.fasta my_model/1u19A.fasta`
2. Prepare PDB file of the template structure.

The **2rh1A.pdb** file is already provided in the
~/rosetta_workshop/tutorials/protein_modeling/input_model/
directory.

3. Download the 2RH1 PDB file from www.rcsb.org. Search for 2RH1, click Download Files, select PDB File (text).

4. Remove the lines corresponding to the T4-lysozyme inserted into the protein for crystallization purposes (residue number 1002 to 1161) and save as 2RH1_noT4L.pdb.

Note that these residues all begin with the letter A (A1002, A1003, etc).

5. Often, structures downloaded from the PDB contain information and characters that Rosetta is incapable of processing. Before using a PDB file with Rosetta, it is best practice to clean it with a script to avoid encountering errors. To clean this PDB file, use the script “clean_pdb.py”:

```
~/rosetta_workshop/Rosetta/tools/protein_tools/scripts/clean_pdb.py 2RH1_noT4L.pdb A
```

Note: The ‘A’ tells the script that you are interested in chain A and two files will be generated: 2RH1_NOT4L_A.pdb and 2RH1_A.fasta.

6. Replace the first line of the FASTA file 2RH1_A.fasta with: >2rh1A
7. Move the cleaned PDB file and FASTA file to your input directory as 2rh1A:

```
mv 2RH1_noT4L_A.pdb my_model/2rh1A.pdb
```

Align the target and template sequences:

The 1u19A.2rh1A.aln file is already provided in the
~/rosetta_workshop/tutorials/protein_modeling/input_model
directory.

1. Align sequences using ClustalW:
2. Copy/paste both sequences for the target (1u19A.fasta) and template (2rh1A.fasta) proteins, including headers starting with “>”, to the ClustalW server at www.genome.jp/tools/clustalw/.
3. Choose “Slow/Accurate” for Pairwise Alignment and “Protein” for sequences.
Leave Output Format: CLUSTAL.
4. Download the clustalw.aln alignment to a file called 1u19A.2rh1A.aln.
5. Move the alignment file to your input directory:

```
mv 1u19A.2rh1A.aln my_model/1u19A.2rh1A.aln
```

Thread the sequence of the target protein onto the backbone of the template protein:

The 1u19A_on_2rh1A.pdb file is already provided in the
~/rosetta_workshop/tutorials/protein_modeling/input_model
directory.

1. Thread the target sequence over the template PDB using the included script:

```
python2.7 ~/rosetta_workshop/Rosetta/tools/protein_tools/scripts/thread_pdb_from_alignment.py \
--template=2rh1A --target=1u19A --chain=A --align_format=clustal 1u19A.2rh1A.aln 2rh1A.pdb \
1u19A_on_2rh1A.pdb
```

Note: This script will output to the terminal all gaps in the sequence alignment by sets of residue numbers. This output is only reference information and may be ignored.

2. Verify that the sequence of the threaded PDB matches the target primary sequence.
Pull the sequence from the threaded PDB with this script and align it to 1u19A.fasta using ClustalW the same way you aligned 1u19 and 2rh1.

```
python2.7 ~/rosetta_workshop/Rosetta/tools/protein_tools/scripts/get_fasta_from_pdb.py \
1u19A_on_2rh1A.pdb A 1u19A_on_2rh1A.fasta
```

Create 3mer and 9mer fragment libraries:

The 1u19A fragment files (aa1u19A03_05.200_v1_3 and aa1u19A09_05.200_v1_3) are already provided in the ~/rosetta_workshop/tutorials/protein_modeling/input_model directory.

1. Robetta will be used to generate fragment libraries for the purposes of this workshop.
2. If you are an academic or non-profit user of Rosetta, make sure you are registered at rosetta.bakerlab.org.
3. Under “Services,” click “Submit” under “Fragment Libraries.”
4. Fill in the form; copy/paste all the text in 1u19A.fasta into the provided field.
5. Under “Target name” put “1u19A”
6. Click “Submit.” See your position in the queue by clicking “Queue” under “Fragment Libraries.” Click on the job ID link for your job and refresh to monitor it until completed.
7. Once complete, fragment files can be downloaded and should be saved as

aa1u19A03_05.200_v1_3

and

aa1u19A09_05.200_v1_3

8. Save all files to my_model/

Create a Rosetta Loops File:

The 1u19A.loops file are already provided in the ~/rosetta_workshop/tutorials/protein_modeling/input_model directory.

1. Create a file called 1u19A.loops in my_model
2. Use a visualization tool to help you determine loop start and end residue numbers. For example, you can use “pymol 1u19A_on_2rh1A.pdb” and show-as ribbon to visualize what residues connect loop regions to helix.
3. Create one line per loop to be built:

```
LOOP 31 38
LOOP 67 74
LOOP 106 118
LOOP 139 169
LOOP 200 216
LOOP 244 252
LOOP 275 279
LOOP 286 290
```

Create other input files (disulfide and span files):

**The 1u19A.disulfide and 1u19A.span files are already provided in the
~/rosetta_workshop/tutorials/protein_modeling/input_model
directory.**

1. Disulfide Bond Constraints

- The .disulfide input file is not necessary for all protein modeling applications.
- This input file informs Rosetta to preserve one or more disulfide bonds during loop building.
- This information is usually gathered from experimental data and literature searches. If this data is unavailable or if your protein of interest does not contain disulfide bonds, this input file can be ignored.
- The format of this file is one disulfide bond per line and each line only contains the two residue numbers involved in that particular bond.

2. Span File

- Span files are necessary when modeling membrane proteins.
- They inform Rosetta which portions of the protein exist within the membrane. This is important because specific scoring terms are different for soluble residues versus those within the membrane.
- Commonly, this information is predicted directly from the protein sequence using algorithms such as OCTOPUS topology prediction.
- Submit the sequence from 1u19A.fasta to <http://octopus.cbr.su.se>
- Save the OCTOPUS topology file as 1u19A.octopus
- Convert the OCTOPUS file to a span file using the following script:

```
~/rosetta_workshop/Rosetta/main/source/src/apps/public/membrane_abinitio/octopus2span.pl \
1u19A.octopus
```

OCTOPUS makes predictions based on artificial neural networks trained with many protein sequences and structures to identify residues at the sites of membrane entry, reentry, membrane dip, TM hairpin regions, and membrane exit. Since this is a computational prediction, it may be important to visualize the predicted span regions on the threaded structure and to compare it with membrane region information from homologous proteins (if available). We will be using the predictions directly from octopus for this example as they have been shown to be effective ~96% of the time. However, these predictions should be adjusted to reflect any experimental evidence pertinent to your protein of interest.

Create the Options file:

**The ccd.options file are already provided in the
~/rosetta_workshop/tutorials/protein_modeling/input_model
directory.**

When creating an options file, remember these important tips:

- Rosetta ignores comment lines beginning with #.
- Avoid mixing tabs and spaces. Be consistent in your formatting (tab-delimited or colon-separated) .
- Much of what goes into an options file can be provided at the command line. Options file allow for shorter command lines that are simpler to organize and follow. In this case, we are using a single template so we have only one starting structure (1u19A_on_2rh1A.pdb). However, you may be modeling with multiple templates all of which have a unique starting structure. Therefore, the line `-in:file:s 1u19A_on_2rh1A.pdb` can be placed on the command line rather than in the options file and the same options file can be used for more than one starting structure (ex: `-in:file:s 1u19_on_2rh1A.pdb` for the first starting structure and `-in:file:s 1u19_on_2vt4A.pdb` for the second). This allows you to avoid creating different options files unnecessarily.

2. Run the Rosetta Loop Modeling application

The following command line runs the loop modeling application in Rosetta. Before launching, ensure that the following files are in the directory you are launching from:

- 1u19A.loops
- 1u19A.disulfide
- 1u19A.span
- 1u19A_on_2rh1A.pdb
- aa1u19A09_05.200_v1_3
- aa1u19A03_05.200_v1_3

This command line is also found in the command_lines.txt file in protein_modeling/ :

```
~/rosetta_workshop/Rosetta/main/source/bin/loopmodel.default.linuxgccrelease @ccd.options \
-database ~/rosetta_workshop/Rosetta/main/database >& ccd.log &
```

If you want to monitor the progress of the the application you may watch the output messages as they enter ccd.log by typing “tailf ccd.log” To get out of this use ctrl-c.

Note: You will see several “[WARNING] missing heavyatom” messages at the beginning. This is normal and can be ignored.

3. Extract PDB files from the silent file

- When the job is finished running, extract the PDB files from the silent files with the following command line.

This command line is also found in the command_lines.txt file in protein_modeling/.

```
~/rosetta_workshop/Rosetta/main/source/bin/score_jd2.default.linuxgccrelease -database \
~/rosetta_workshop/Rosetta/main/database -in:file:silent 1u19A_on_2rh1A_ccd_test_01.out \
-score:weights membrane_highres_Menv_smooth.wts -in:file:fullatom -in:file:spanfile 1u19A.span \
-out:pdb
```

4. Analyze your data

See tutorial from De Novo Folding on “Score and extract PDBs” and “Score vs. RMSD plots” for further instructions on analysis.

To Cluster large sets of models, see the Rosetta Clustering tutorial below.

Rosetta Clustering Tutorial

Note: If you would like to manually create files instead of using those that are provided, (e.g., input files), write them to a new directory! (mkdir ~/my_cluster)

1. Prepare your input files

- Prepare Silent Files or list of PDBs:

The silent file 1u19A_on_2rh1A_ccd_production_01.out file is already provided for you in the ~/rosetta_workshop/tutorials/protein_modeling/input_cluster/ directory.

- Prepare Options file:
The cluster.options file is already provided in the
~/rosetta_workshop/tutorials/protein_modeling/input_cluster/
directory
- Rosetta ignores comment lines beginning with #.
- Avoid mixing tabs and spaces. Be consistent in your formatting (tab-delimited or colon-separated)

2. Run the Rosetta Clustering application

- Run the clustering.py script, which will execute the Rosetta cluster application and output a series of summary files:

```
~/rosetta_workshop/Rosetta/tools/protein_tools/scripts/clustering.py \
--silent=1u19A_on_2rh1A_ccd_production_01.out \
--rosetta ~/rosetta_workshop/Rosetta/main/source/bin/cluster.default.linuxgccrelease \
--database ~/rosetta_workshop/Rosetta/main/database \
--options=cluster.options cluster_summary.txt cluster_histogram.txt
```

3. Analyze your data

The cluster_summary.txt and associated files are provided in the
~/rosetta_workshop/tutorials/protein_modeling/input_cluster/
directory.

- Sort the cluster_summary.txt file by the score column from lowest to highest:

```
sort -rnk4 cluster_summary.txt > cluster_summary_sorted.txt
```

- Look at the top 5 clusters by size:

```
head -n 5 cluster_summary_sorted.txt
```

- Extract the models you are interested in viewing from the binary silent file (make sure the silent file 1u19A_on_2rh1A_ccd_production_01.out and 1u19A.span are in the directory you are running from. If not, copy them to the present directory):

```
~/rosetta_workshop/Rosetta/main/source/bin/score_jd2.default.linuxgccrelease\
-database ~/rosetta_workshop/Rosetta/main/database \
-in:file:silent 1u19A_on_2rh1A_ccd_production_01.out \
-in:file:silent_struct_type binary -in:file:fullatom \
-score:weights membrane_highres_Menv_smooth.wts \
-in:file:spanfile 1u19A.span -out:output -out:pdb \
-out:file:fullatom \
-in:file:tags 1u19A_on_2rh1A_ccdS_0241 1u19A_on_2rh1A_ccdS_0310 \
1u19A_on_2rh1A_ccdS_0394 1u19A_on_2rh1A_ccdS_0028 1u19A_on_2rh1A_ccdS_0286
```

- View the models using protein visualization tool of your choice.
- Examine the PDB files you extracted using any text editor. The overall Rosetta scores for each scoring term as well as the Rosetta scores for each individual residue can be found at the bottom of the model PDB files.