Rosetta Documentation main site: <https://www.rosettacommons.org/docs/latest/Home>

[RosettaCM Documentation](https://www.rosettacommons.org/docs/latest/application_documentation/structure_prediction/RosettaCM)

* [Hybridize mover documentation](https://www.rosettacommons.org/docs/latest/scripting_documentation/RosettaScripts/Movers/movers_pages/HybridizeMover)
* [RosettaScripts documentation](https://www.rosettacommons.org/docs/latest/scripting_documentation/RosettaScripts/RosettaScripts)

Additional Material:

* [High-Resolution Comparative Modeling with RosettaCM](https://www.sciencedirect.com/science/article/pii/S0969212613002979)
  + Original paper describing method
* [Protocols for Molecular Modeling with Rosetta3 and RosettaScripts](https://pubs.acs.org/doi/10.1021/acs.biochem.6b00444)
  + Tutorial paper giving overview of Rosetta, primary methods, and associated tutorials
* [MeilerLab Tutorials (comparative modeling and more)](http://meilerlab.org/index.php/rosetta-tutorials)

Notes: Paths are relative. You will need two directories

* Rosetta: all source code necessary to run Rosetta
  + /PATH/TO/ROSETTA/
* rosettagpcr: directory containing all files neccesary for this tutorial
  + /PATH/TO/ROSETTAGPCR/

Necessary Inputs:

* Target fasta (/**PATH/TO/ROSETTAGPCR/**rosettagpcr/class\_a/{TARGET}/target.fasta)
* Template pdbs (find new or pull from /PATH/TO/ROSETTAGPCR/rosettagpcr/class\_a/templates/)
* Alignment file (make or lives in /PATH/TO/ROSETTAGPCR/rosettagpcr/class\_a/{TARGET}/alignment.fasta)

Step 1: Pick templates

* The templates that are present in /PATH/TO/ROSETTAGPCR/rosettagpcr/class\_a/{TARGET}/alignment.fasta are the best templates by identity as of fall 2018, if better templates have since been released or you want to change templates for a reason specific to your target do the following
  + Search template structures by identity to target
  + Also consider whether template is in the same receptor family (i.e. don’t use a class C gpcr to model a Class A receptor)
  + Consider ligand similarity (i.e. use peptide-binding receptors to model target peptide-binding receptor)
  + Want up to 5 templates
    - If available templates are >50% ID, need one template
    - If less than 50% ID, consider multiple templates
    - Below 20% ID is not advisable

Step 2: Prepare templates

* First start in /PATH/TO/ROSETTAGPCR/rosettagpcr/class\_a/templates/ to look for already prepared and aligned template pdbs, otherwise do the following
* Remove fusion proteins and non-ligand HETATMs (i.e. cholesterols/pegs/etc.)
* Align templates in space
  + Hint: select the TM helices of one template and align other templates to just this region
* Save aligned template pdbs

Step 3: Prepare alignment file

* Start with /PATH/TO/ROSETTAGPCR/rosettagpcr/class\_a/{TARGET}/alignment.fasta if you are happy with the templates that were preselected, otherwise:
* Go to /PATH/TO/ROSETTAGPCR/rosettagpcr/class\_a/ and find alignment.fasta
  + This is a master alignment of all GPCRs
* Pull out the fastas for your target and templates
  + There are a number of template pdbs already included in this alignment file (i.e. 6ndd\_) corresponding to the templates that live in /PATH/TO/ROSETTAGPCR/rosettagpcr/class\_a/templates/
  + Use these template alignments since they will already have accounted for non-resolved residues, truncated termini, and thermostabilizing mutations
  + **If your template does not have a provided crystallographic alignment**, the template alignment will need to be checked for non-resolved residues, truncated termini, and thermostabilizing mutations
* Adjust alignment according to the structural alignment of the templates
  + For most receptors this is already done but it always pertinent to check (garbage in equals garbage out and the alignment is the most important part of this process)
* When alignment is good, save alignment file such that the target sequence is first and then template sequences are ordered from best template to worst

Step 4: Setup RosettaCM

* With target.fasta, alignment.fasta, and (aligned) template pdbs use the following command to prepare a directory for modeling
  + python /PATH/TO/ROSETTAGPCR/rosettagpcr/scripts/setup\_RosettaCM.py –bin **/PATH/TO/ROSETTA/**Rosetta/main/source/bin--fasta target.fasta --alignment alignment.fasta --templates {template pdbs}
* This script creates a new directory rosetta\_cm, threads your target sequence onto the template pdbs, generates the rosetta\_cm.xml and flags files that are needed to run RosettaCM
* **For all files that were generated, check the paths (particularly in the rosetta\_cm.xml and flags files**

Step 5: Prepare span and disulfide files

* Change into the newly made rosetta\_cm directory
* Make disulf.txt
  + Identify the sequence position of cysteines involved in disulfide formation (i.e. TM3 and ECL2)
  + Make a file “disulf.txt” that contains these each pair of residue positions separated by whitespace, i.e.
    - 93 151
* Make span file
  + The Rosetta score function needs to be told which residues live in the membrane, this is the purpose of the span file
  + Go to <http://octopus.cbr.su.se/>
  + Copy and paste your target sequence into the text box and hit submit
  + On the results page, click on link to “OCTOPUS topology file (txt)”
  + Copy the entire page into a text file called octopus.txt
  + Run script to convert octopus format to Rosetta format
    - /PATH/TO/ROSETTAGPCR/Rosetta/tools/membrane\_tools/octopus2span.pl octopus.txt > span.txt
  + Format should look like:
    - TM region….
    - 7 325 #### number of TM spans and number of residues
    - antiparallel
    - n2c
    - 7 lines with 4 numbers each corresponding to start and end position of each span
  + **IF ADDING A LIGAND**, add 1 residue to the second number in line 2
    - So in the above example 325 becomes 326

Step 6: Check that all inputs are present and run homology modeling

* In the rosetta\_cm directory there should be
  + rosetta\_cm.xml
  + flags
  + span.txt
  + disulf.txt
  + threaded template pdbs
  + output/ directory
* If everything is present run this command to generate a single model
  + /PATH/TO/ROSETTA/Rosetta/main/source/bin/rosetta\_scripts.default.linuxgccrelease @flags -database /PATH/TO/ROSETTA/Rosetta/main/database -nstruct 1
* Output models live in the output directory

Optional: Modeling with a Ligand Present

If you want to add a ligand in your model, a few extra steps are needed

* Using a crystallized ligand from a template
  + If the template is aligned with the other templates, save the ligand coordinates as its own pdb file
  + Convert the ligand pdb to mol2 file and add hydrogen atoms
    - /nfs/soft/openbabel/current/bin/obabel -ipdb xtal-lig.pdb -omol2 -O xtal-lig.mol2 -h -p 7.4
    - Alternative for “correct hydrogen-bonding” add hydrogen in chimera within the protein-ligand-complex. Add charges to the ligand and save as mol2.
  + Convert the mol2 file into a Rosetta readable file
    - /PATH/TO/ROSETTA/Rosetta/main/source/scripts/python/public/molfile\_to\_params.py xtal-lig.mol2 --keep-names --clobber --extra\_torsion\_output --centroid -p lig -n lig
    - Outputs should be
      * lig\_0001.fa.pdb
      * lig\_0001.cen.pdb
      * lig.fa.params
      * lig.cen.params
      * lig.fa.tors
      * lig.cen.tors
  + Cat the lig.pdb coordinates to the **threaded** pdbs in the rosetta\_cm directory
    - cat lig\_0001.fa.pdb >> template\_thread.pdb
  + In the flags file add:
    - -extra\_res\_cen lig.cen.params
    - -extra\_res\_fa lig.fa.params
    - -extra\_improper\_file lig.cen.tors
  + In the span file:
    - Update the total length of the protein to have one additional residue
    - I.e: Original
      * 7 278
    - Updated
      * 7 279
  + In the rosetta\_cm.xml file
    - Find the line that starts “Hybridize”, the end of this line should be the disulfide file path
    - Between the end of the disulfide file path and before the “>” symbol add the following:
      * add\_hetatm=”1”
  + Run modeling using the above provided command line
* Using an analog a crystallized ligand from a template
  + Align the analog to the crystallized ligand
  + Save the coordinates of the analog as xtal-lig.pdb
  + Use above pipeline
* Using a fully unrelated ligand to any of the templates
  + Consider docking against a threaded template
  + Use the coordinates of the docked pose in the above pipeline