**G Protein-Coupled Receptor Modeling Tutorial (as of 2/27/20)**

**Author: Greg Szwabowski**

The following table outlines the steps required to create homology models based on a GPCR target sequence in MOE that are then loop modeled in Rosetta. The necessary files can be found in the .zip file at the following link:

<https://github.com/gszwabowski/guides/blob/master/HM_tutorial_files.zip>

|  |  |  |
| --- | --- | --- |
| **Step** | **Description** | **Figure** |
| 1 | Download your target sequence from the Protein Data Bank (<https://www.rcsb.org/>) or Uniprot (<https://www.uniprot.org/>) as a .fasta file. |  |
| 2 | Open the excel file “Ngo\_supplementData3.xlsx”. Click on the GPCR1 tab in the first column and then type the name of your receptor and then click *OK*. |  |
| 3 | The database will now show pairwise comparisons of various GPCR your target receptor. The “GPCR-CoINPocket Score” column in the table should be sorted automatically, this metric will be used to select a template structure to model the target receptor. Starting from the first entry in the table, verify that the receptor in the “GPCR2” column:   * Is not the same receptor as your target receptor * Is not within the same subfamily as your target receptor * Has a crystallized structure within the Protein Data Bank   Once a table entry satisfying all 3 criteria is found, the receptor in the “GPCR2” column can be used as a template structure from which to model the target receptor. Take note of the PDB code of the target receptor, as it will be used in the next step. |  |
| 4 | Open MOE and click *File* 🡪 *Open*. Open the previously downloaded FASTA sequence of your target receptor. Click *File* 🡪 *Open* once more. In the dialog box, click the box with the RCSB PDB logo on the left-hand side. In the “Codes” text box, type the 4 letter PDB code of your target receptor and then click *OK*. Close the dialog box. |  |
| 5 | Both the target sequence (Chain 1) and template structure (Chain 2) should be loaded into MOE. Most template structures will have a fusion partner that needs to be deleted, typically numbered starting at residue 100X. Any waters (denoted “W”) within the template structure can be deleted, as well as ligands (denoted “L”) that are *not* located within the binding pocket of the template structure. The binding pocket can be found using the “Site Finder” function in MOE if you are unsure. Only one ligand should remain in the system after this step, this ligand will be used during the homology modeling process. |  |
| 6 | Within the sequence editor, click *Alignment* 🡪 *Align/Superpose*. In the new panel that appears within the sequence editor, click *Sequence and Structural* and change this option to *Sequence Only* then click *Align*. |  |
| 7 | Turn on secondary structure annotation using the  button in the sequence editor and check the alignment to ensure that any gaps in alignment are shifted into loop regions and conserved TM.50 residues remain aligned. An alignment should be performed at <https://gpcrdb.org/> to assist with this process. The second figure displays the GPCRdb suggested alignment, the third figure displays a portion of the original MOE alignment, and the fourth figure displays the same portion of the alignment that has had gaps shifted into loop regions. In addition, the highlighted green residues are conserved TM.50 residues for TM helix 4 and have remained aligned throughout adjustment of the alignment. |  |
| 8 | Once the alignment is complete, homology models can be built. Click *Protein* 🡪 *Homology Model*. In the dialog box that appears, there are a few settings that must be changed:   * In the “Options:” section, the Environment for Induced Fit setting must be enabled using the “Ligand Atoms” option. If you are curious as to which atoms MOE is using, clicking the question mark in this section will make the ligand atoms blink. * Model scoring should be changed from “GB/VI” to “Contact Energy”.   Once these settings are changed, change the “Output Database” name to reflect the target:template pairing that is being modeled. Lastly, click *OK* to generate your homology models. |  |
| 9 | Once the homology models are generated, a database will open containing 11 homology models. In order to select a model to use moving forward, they must be sorted. Since the homology models were created using Contact Energy as the scoring metric, select the Contact Energy column and right-click it. Click *Sort* 🡪 *Ascending*. With the entries in the database sorted, the entry with the *lowest* contact energy should be imported into MOE by double-clicking the first entry’s “mol” entry. |  |
| 10 | Once the lowest contact energy homology model is loaded into MOE, it must be saved as a .pdb file to use with Rosetta. To do this, click *File* 🡪 *Save* and then type the desired filename ending in “.pdb” and click *OK*. MOE will automatically save the model as a .pdb file. |  |
| 11 | Before loop modeling is performed, Rosetta fragment files must be generated based on your saved homology model based on the nine residues prior to the first loop anchor, the ECL2 sequence and the nine residues after the second loop anchor. With the homology model open, go to the sequence editor and look at the model’s sequence for TM4, ECL2, and TM5. If secondary structure annotation is enabled, this will be the 4th and 5th long stretch of helical residues denoted by the red bars above the model’s sequence. The second figure to the right denotes the selected sequence to be submitted to the Robetta fragment server (<http://old.robetta.org/>) Once the sequence is selected, click *Edit* 🡪 *Copy As* 🡪 *Raw Sequence (.seq)*, which will copy the sequence to your clipboard. |  |
| 12 | Head to the Robetta webpage (<http://old.robetta.org/>) and register for an account if you do not already have one. One logged in, click *Submit* under the “Fragment Libraries” section. On this new page, paste your copied sequence into the “Paste Fasta” box. Next, enter your username as well as the target receptor’s name in the respective boxes and then click *Submit*. Fragment generation takes some time and your results will be emailed to the email address you registered with. Once the results have been emailed, click the first link under the “Download Fragment Files” section. Two fragments will be generated, known as a “3mer” and “9mer”. The 3mer will have the string “03” present in its URL, while the 9mer will have “09” present. Once on the page for the URL, right-click anywhere on the page and click *Save as*, ensuring that the page is saved as a .txt file. Take note of the names used to save both fragments as they will be used later. |  |
| 13 | Back in the MOE window, open your template structure and delete the chain containing the sequence, leaving only the template ligand used in the “Environment for Induced Fit” portion of the homology modeling process. Click *File* 🡪 *Save* and save the ligand as a .sdf file. |  |
| 14 | Prior to performing any loop modeling, the necessary loop modeling files must be present within a folder on the HPC. Use Filezilla (or an alternate SFTP client) to log in to the HPC and transfer job files. Here is how to login:   * Host: type “sftp://username@hpclogin.memphis.edu”. The SFTP string is important since it’s the HPC’s preferred method of file transfer. * Username: can be left blank * Password: Your account password * Port: blank |  |
| 15 | The files needed to perform loop modeling are contained within the “loop\_model” folder in the .zip file detailed in the introduction. These files can be moved once your SFTP client (Filezilla in this case) is connected to the HPC. In the left pane (local directory), navigate to the directory containing the loop modeling files after extracting them from the .zip file. In the right pane (HPC directory), create a folder you wish to run your jobs in or navigate to a pre-existing folder. Drag both the loop modeling files from the left pane into the right pane and wait for the upload to complete. In addition, the previously created homology model .pdb, template ligand .sdf , and fragment .txt files must be moved into the same folder. |  |
| 16 | Once the loop modeling files are present on the HPC, a command line must be opened using an SSH client. In this case, PuTTY will be used (<https://www.putty.org/>). Once installed, PuTTY can be configured as follows:   * Hostname:   <username>@hpclogin.memphis.edu, where username is your HPC account username.  Click *Open* and type your password into the window that is created once prompted for it. |  |
| 17 | Once logged into your HPC directory, use the command cd to navigate to the folder containing the files that were moved using Filezilla. (ex. cd loop\_model). Additionally, the ls command can be used to list the files in the folder you are currently located in. |  |
| 18 | Since loop modeling is being performed with a ligand present in the binding pocket, ligand parameters must be generated so the loop modeling process won’t present any errors. To do this, use the following command:  /public/apps/rosetta/2017.29.59598/main/source/scripts/python/public/molfile\_to\_params.py -n <ligand abbreviation> -p <ligand abbreviation> <ligand>.sdf  Example command using the ligand D7V:  /public/apps/rosetta/2017.29.59598/main/source/scripts/python/public/molfile\_to\_params.py -n D7V -p D7V D7V.sdf  This will output a .params file, which is used by Rosetta to interpret the ligand residue. | …  …  … |
| 19 | Once in your working directory, files can be edited using the emacs <filename> command. emacs can only be navigated using the keyboard. To save an edited file, press “Ctrl + X” and then “Ctrl + S”. To close emacs, press “Ctrl + X” and then “Ctrl + C”. Multiple files must be edited, including:   * disulf.cst: this file contains the two residue numbers that detail the 3.25 and 45.50 Cys residues in GPCR. Loop modeling incorporates a distance constraint between these residues that has shown to improve model quality. To find these numbers, go to your previously generated GPCRdb alignment and match the 3.25 and 45.50 Cys residues to your homology model sequence within MOE. The numbers above the Cys residues within the MOE sequence editor will be used * hm.loops: this file details the loop anchor residue numbers, the last residue in TM4 and the first residue in TM5. Again, MOE’s sequence editor can be used to verify the numbers to be used. ALWAYS double check! * kic\_with\_frags.flags: This file details many of the input/output commands for Rosetta, many of which are useful with loop modeling. in:file:s should be your homology model .pdb, as it is the input structure for loop modeling. The in:file:extra\_res\_fa argument calls for the .params file generated in the previous step. loops:frag\_files should be changed to reflect the names of the fragment .txt generated earlier. Note that the 9mer fragment *must* be listed first, then the 3mer fragment. out:nstruct details how many loop conformations are going to be sampled. By default, 250 are generated, though this number can be changed if jobs are being split. * KICfragsub.sh: this file is the “job” file for loop modeling that will be run. The job name (--job-name) can be changed to more easily recognize a job when looking through the job queue. Additionally, the timeout limit of the job can be adjusted with the --time line. This value is in minutes, by default it is set to 14400 (10 days). If your job will not run, delete this line for easier job queueing. | disulf.cst:    hm.loops:    kic\_with\_frags.flags:    Example kic\_with\_frags.flags file:    KICfragsub.sh: |
| 20 | Once the files detailed in the previous step are edited and saved, the job can be run using the following command:  sbatch -D . KICfragsub.sh  Once finished, there should be 250 numbered .pdb files within your working directory, each with its own ECL2 conformation. The progress of jobs can be checked using the command squeue -u <username>. This command will tell display the status of a user’s current jobs and is helpful for determining which jobs are done and which are still pending/running. |  |
| 21 | Once the job is complete, use the command grep final\_looprelax\_score \*.pdb > scores.txt to aggregate the loop scores into one .txt file. Download this .txt file along with the 250 generated .pdb files into one folder on your local machine using Filezilla. |  |
| 22 | Open MOE and click *File* 🡪 *New* 🡪 *Text Editor*. In the text editor window, click *File 🡪 Open* and navigate to where you have extracted the contents of the HM\_tutorial\_files.zip file. Click the “pdb2mdb\_b.svl” file and then click *Open*. Lastly, click *SVL* 🡪 *Save and Load* to load the script into your MOE environment. |  |
| 23 | Once the script is loaded, click *File 🡪* Open and navigate to the directory *above* the folder containing the loop models and click . Then, click the  button in MOE to open the SVL command prompt. Use the following command to generate a .mdb database of your loop models:  pdb2mdb ‘<dirname>’  where <dirname> is the name of the folder containing the 250 loop models. |  |
| 24 | Open the scores.txt file in a text editor and use find and replace to correctly format the scores so they can be imported to a MOE database as well as possess the same names as the receptors in the database created in the previous step. The find string should be “:final\_looprelax\_score” and the replace string should be left blank. Replace all and then save the .txt file. |  |
| 25 | In addition to creating a database with the models, a database with the scores must be created as well. In MOE, click *File* 🡪 *New* 🡪 *Database*. In the dialog box that pops up, name the file “scores.mdb” and delete the name from the “mol” option. Click *OK* to create the database. In the database window, click *File* 🡪 *Import*. In the Database Import window, click *Add*. Select the scores.txt file and then click *OK*. Under the “Import Fields” section in the Database Import window, click the first entry and change the Field Name (below the entries) to “filename” and press the enter key. Click the second entry and change the Field Name to “scores” and press the enter key. Lastly, click *OK* and entries in the text file will be imported into the database. Within the database, ensure that entries in the “filename” field end in “.pdb” so that they will be correctly matched when merging databases in the next step. |  |
| 26 | Open the database containing your loop model .pdb files and click *File* 🡪 *Merge*. In the Merge Databases window, click *Browse* for Input Database 2. Select the scores database created in the prior section, then click *OK*. Name the output database “models\_scored.mdb”, ensure the “New Database” option is checked, and click *Next*. In the next section, click *Undefined* after “key1”, select “filename” for both Database 1 and Database 2 and click *Next*. In the next section, click all 3 entries in the “Field Name” box to select them and then click *Next*. In the next section, click *Merge* to merge the databases and then click *Close*. |  |
| 27 | Now that a database containing the loop models and their respective scores has been created, the Cys 3.25-45.50 distance must be measured. Open the db\_measure\_disulfide\_lb.svl file in the MOE text editor and then click *SVL* 🡪 *Save and Load*. Click the  button in MOE to open the SVL command prompt. Use the following command to measure the disulfide distance of each model in the database:  cys\_measure\_LB [database\_file, database\_field, residue1, residue2]  where:   * database\_file is the name of the database file containing the scored loop models (e.g. ‘models\_scored.mdb’). * database\_field is the database field containing the model structures, ‘mol’ in this case. * residue1 is the number of the 3.25 Cys residue. This number will be the same as the first number used in the disulf.cst file. * residue2 is the number of the 45.50 Cys residue. This number will be the same as the second number used in the disulf.cst file.   There should now be a new database field titled “S-S Dist” within the database. Next, click the “scores” field and sort by ascending, i.e. the lowest scoring structures. Hold the shift key and select the top 10 scoring entries with disulfide distances that are between 1.0 and 5.1 Å, as these reflect typical disulfide distances found in GPCR. Hold the shift key and click the 5 field names as well, then click *File* 🡪 *Save as* and save the selected entries as a new databased suffixed “\_top10.mdb”. |  |
| 28 | The final step in the receptor modeling process involves minimization of the top 10 scoring structures and creation of the conserved disulfide bond between Cys residues 3.25 and 45.50. To do this, open the make\_disulfide\_min\_loop.svl script in the MOE text editor and click *SVL* 🡪 Save and Load. Once the script is loaded, use the following command in the SVL command prompt:  SSbond [database\_file, database\_field, start, end, cys1, cys2]  Where:   * database\_file is the database containing the top 10 scoring loop modeled structures. * database\_field is the database field containing the model structures, ‘mol’ in this case. * start is the first loop anchor residue number, this will be the same as the first number used in the hm.loops file. * end is the first loop anchor residue number, this will be the same as the second number used in the hm.loops file. * cys1 is the number of the 3.25 Cys residue. This number will be the same as the first number used in the disulf.cst file. * cys2 is the number of the 45.50 Cys residue. This number will be the same as the second number used in the disulf.cst file.   In the database containing the top 10 scoring loop modeled structures, a field titled “S-S bonded” will be created and each entry will fill as it is completed. Once this is done, receptor modeling is complete and the 10 loop modeled receptor models can be used for docking. |  |

Troubleshooting

Q: Why is my loop modeling job not running?

A: There are multiple answers to this question depending on the circumstances. When using the squeue -u <username> command, a job labeled P under the ST (status) column is queued but not currently running. This means that the HPC is trying to allocate resources to schedule the job. If you have patiently waited and the job still isn’t running, use scancel <jobID> to cancel the job (where jobid is the job number listed when using the squeue command) and remove the #SBATCH --time=14400 line from the job’s KICfragsub.sh file. After this, resubmit the job. If your job is not listed immediately after running the sbatch command, this typically means that something is not configured right within the job’s files. Using the commands more slurm-XXXXXX.out or more loops.log is a great way to troubleshoot what might be wrong with your job. Common mistakes include incorrectly numbered disulfide residues, typos within file names, and incorrect listing of the fragment files within the kic\_with\_frags.flags file.

Q: Why am I getting a “residue SG XXX not found” error?

A: You may be modeling an atypical GPCR without the conserved 3.25-45.50 disulfide bond. If cysteines are not present at these positions in the GPCRdb alignment, then the disulfide constraint can’t be used. In this case, the -cst\_fa\_file disulf.cst and -cst\_fa\_weight 1000 lines should be deleted from the kic\_with\_frags.flags file. If your target GPCR *does* have Cys residues at this position, use the command grep SG <homology model>.pdb to ensure that the numbering in your disulf.cst file matches with the Cys numbers within the .pdb file. If all else fails, open the .pdb file in MOE, renumber the receptor chain starting from 1, then save and reupload the renumbered .pdb file.

Q: What does “Unrecognized residue: XXX” mean?

A: This error usually means that a .params file for the template ligand is not present, check to ensure that one was created when the command in step 18 was run.

Q: What does “Residue outside res\_map range” mean?

A: While it is difficult to ascertain the exact cause of this error, it typically occurs when something is amiss with the template ligand files. For example, this error occurs when the ligand used during loop modeling is located outside the binding pocket of the receptor (e.g. a ligand that binds a helical domain on the exterior of the receptor). In this case, check to see that your ligand is located inside the binding pocket of the target receptor. In addition, including multiple conformers of the ligand may alleviate this error. Using MOE, a conformation search can be performed once the ligand .sdf file is loaded into the system. Once the conformation search is complete, load *every* conformation into MOE and then save the system as a .sdf file named “ligand\_conformations.sdf”. Upload this .sdf file to your working directory and return to this directory in PuTTY. A new .params file must be generated using a slightly different command:

/public/apps/rosetta/2017.29.59598/main/source/scripts/python/public/molfile\_to\_params.py -n <ligand abbreviation> -p <ligand abbreviation> --conformers-in-one-file ligand\_conformations.sdf

Then, the job can be run using the sbatch command. In addition, adding the line

-load\_PDB\_components false to your kic\_with\_frags.flags file may allow the job to run.

Q: My job stopped and is displaying the error “slurmstepd: error: \*\*\* JOB XXXX ON cXXX CANCELLED AT XXXX-XXX-XXTXX:XX:XX \*\*\*”

A: This error means that the job has timed out. To remedy this, create a new folder in the working directory using the command mkdir. Using Filezilla, copy every file *except* the numbered .pdb files generated by Rosetta, the .out file and the scores.sc file to the newly created directory. Navigate to the newly created directory and edit the kic\_with\_frags.flags file using emacs. The #-out:suffix line should be changed to -out:suffix \_A, which instructs Rosetta to name the new .pdb files differently than the ones generated in the canceled job. In addition, the -out:nstruct line should be changed to make sure that 250 structures are produced between the canceled job and the job in the new folder. For example, if 107 .pdb files were generated in the canceled job, 143 more are required to complete the set of 250. Thus, the -out:nstruct line should be updated to read -out:nstruct 143. Once the file is saved, the new job can then be run with sbatch.