Using experimental data with Rosetta

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`

If you want to try making files that already exist (e.g., input files), write them to a different directory! (mkdir my_dir)

Authors

- Tutorial and Program Author:
  - Clara T. Schoeder and Rocco Moretti (clara.t.schoeder@vanderbilt.edu,rmorettiase@gmail.com)
- Corresponding PI:
  - Jens Meiler (jens@meilerlab.org)
- Instructors:
  - Alican Gulsevin (alican.gulsevin@vanderbilt.edu)
  - Taylor L. Jones (taylor.l.jones.1@vanderbilt.edu)
  - Clara T. Schoeder (clara.t.schoeder@vanderbilt.edu)

Citation

Important citations for modeling in Rosetta with experimental restraints

There are a large number of publications where Rosetta has been used together with experimental data. The papers depicted here are just a small collection for reading.

Del Alamo D, Tessmer MH, Stein RA, Feix JB, Mchaourab HS, Meiler J. Rapid Simulation of Unprocessed DEER Decay Data for Protein Fold Prediction. Biophys J. 2020, PMID: 31892409.

Bender BJ, Vortmeier G, Ernicke S, Bosse M, Kaiser A, Els-Heindl S, Krug U, Beck-Sickinger A, Meiler J, Huster D. Structural model of ghrelin bound to its G protein-coupled receptor. Structure. 2019. PMID: 30686667.

Seacrist CD, Kuenze G, Hoffmann RM, Moeller BE, Burke JE, Meiler J, Blind RD. Integrated Structural Modeling of Full-Length LRH-1 Reveals Inter-domain Interactions Contribute to Receptor Structure and Function Structure, 2020, PMID: 32433991

Kuenze G, Meiler J. Protein structure prediction using sparse NOE and RDC restraints with Rosetta in CASP13. Proteins. 2019, PMID: 312929

Chou HT, Apelt L, Farrell D, White SR, Woodsmith J, Svetlov V, Goldstein JS, Nager AN, Li Z, Muller J, Dollfus H, Nulder E, DiMaio F, Nachury MV, Walz T. The Molecular Architecture of Native BBSome Obtained by an Integrated Structural Approach. Structure. 2019. PMID: 31303482

More information on using restraints in Rosetta can be found here: https://www.rosettacommons.org/docs/latest/rosetta_basics/file_types/constraint-file

You find throughout the literature both restraint and constraint. Restraint is the commonly used term in molecular modeling. In Rosetta restraints are called 'constraints' to differentiate them from retsraint in molecular modeling approaches. Therefore, when we generally talk about restraining sampling space, we use restraint, but if specifically talking about a protocol in Rosetta, we use 'constraints'.

Overview

In this tutorial we want to assess how using restraints can change the outcome of our modeling approach. Throughout the tutorial we will apply more and more restraints to a docking between an antibody variable fragment and its epitope. We will use the antibody STAU-281, which binds to the NEAT2 domain of the bacterial outer-membrane protein IsdB. IsdB is part of the iron capture system of the bacterial outer membrane and a target of the human adaptive immune response (Bennett MR, et al. mBio, 2019).

The purpose of the tutorial is to train using restraints in Rosetta. Therefore, you are highly encouraged to change the constraints and monitor the outcome.

We will use the case of the co-crystal structure of STAU-281 with NEAT2 as a benchmark case, meaning, the co-crystal structure is solved, so we can compare our modeling to the known crystal structure. This is a good way to test a method for its performance prior to applying it to research question with no crystal structure to compare to. Naturally, low resolution experimental data are incoporated in modeling when no structural solution is possible yet.

Software packages needed for this tutorial

- Rosetta
- PyMOL

Tutorial

In every section of the tutorial, we will use the same procedure, but apply an increasing amount of constraints. Our general workflow will consist of the following steps:

- create a starting model using the edit function in PyMOL
- use the protein-protein docking protocol without and with constraints
- score the docked structures for their RMSD to the co-crystal structure using SimpleMetrics

Generally, there is a reference model to score against in your output folder. Prepared inputs will be in the input folder. We use the crystal structure PDB:6p9h - so, don't check, don't cheat. It's gonna only be have the fun when you know the solution. Therefore: Also don't check the reference model before we start anaylzing our own generated models. You can influence the outcome of your docking runs when you position the model in its known binding position (the lowest energy conformation, as this position scores well in Rosetta, it will continue to except it as the best solution). In a real world project, you would not know the solution, so your task is to sample the conformational space as thoroughly as you can to find the lowest energy conformation.

1. Docking of STAU-281 to NEAT2 without constraints:

   This is a so-called global docking. We do not have any information where on NEAT2 the antibody binds. The only information we can use is that an antibody always uses its complementary determining region (CDR) to engage an epitope. Normally, the major interaction are made through the HCDR3, whichis the third loop of the antibody heavy chain, a point of major diverity. In the provided models, reisdues 214-219 correspond to the HCDR3 region.

We start our modeling by making a working directory and copying the necessary files in there (starting models and rosetta_scripts files):

```
mkdir my_files
cd my_files
cp ../input_files/NEAT2_STAU-281_000* .

cp ../input_files/docking* .
```

In order to get an idea, how the input models look like open them in PyMOL by typing:

```
pymol NEAT2_STAU*pdb
```

The NEAT2 domains (chain A) are aligned and should not be moved, as we calculate the overall RMSD later-on against the complex co-crystal structure. The antibody fragment (chain H+L) are at different places. This is a random position and can be changed. If you want to modify the position of the antibody frgament in resect to the NEAT2 domain, you can either use the PyMOL editing mode, seelct the respective chain and move it while pressing Shift on your Keyboard. Make sure you do not dirupt the overall structure. Another option to move the antibody in respect to the NEAT2 domain can be achieved by using the commands *rotate* and *translate* while applying to the antibody frgament chain.

For example:

```
rotate x,60,(NEAT2_STAU-281_0001 and chain H+L)
```

will rotate the antibody fragment around the x-axis by 60 degree. The command:

```
translate [0,-20,0], (NEAT2_STAU-281_0001 and chain H+L)
```

will move the antibody fragment along the y-axis.

If you use these options you will have to save the structure from PyMOL by choosing Export molecule from the Start menu. Export your molecule as PDB and overwrite your current version of the structure. If you choose a new name for your structure apply these through-out the tutorial.

We will now run a small protein-protein docking and calculate the RMSD against the complex structure.

```
/path_to_rosetta/main/source/bin/rosetta_scripts.default.linuxgccrelease \
  -s NEAT2_STAU-281_0001.pdb NEAT2_STAU-281_0002.pdb NEAT2_STAU-281_0003.pdb \
  -parser:protocol docking_full.xml -out:prefix no_cst_ @docking.options
```

The docking will take a little bit of time, you can continue reading through this protocol and set-up more runs in the meantime.

In this tutorial, we will only compare our outputs against the provided co-crystal structure using the CA-RMSD. Normally docking would we evaluated as explained in the protein-protein docking protocol by comparing score versus RMSD. In a real-world experiment, we would definitly do that. for teaching purposes, we will only focus on the RMSD here.

We will do so by running a small SimpleMetrics rosetta_scripts protocol.

To use the SimpleMetrics and also the comparison model type the following command:

```
cp ../input_files/RMSD_metric.xml .
cp ../output_files/complex.pdb .
```

Now execute the following Rosetta_scripts protocol:

```
/path_to_rosetta/main/source/bin/rosetta_scripts.default.linuxgccrelease -s no_cst_*.pdb \
  -parser:protocol RMSD_metric.xml -in:file:native complex.pdb -out:file:scorefile no_cst_RMSD.sc
```

You can either open the scorefile by typing

```
cat no_cst_RMSD.sc
```

Which values do you get for your RMSD? How do your models look like when you open them in PyMOL?

2. Using a constraint while docking

As touched on earlier in our tutorial, antibodies do only engage their antigen with a certain interface, the paratope, specified through the CDRs. Most important for these interactions is the HCDR3, which corresponds to residues 214-219 in our models.

In this part of the tutorial we want to use this knowledge to constrict our sampling space during docking.

In order to do so we have to construct a constraint file. Go to the Rosetta constraint documentation and read-up on the possible constraints: https://www.rosettacommons.org/docs/latest/rosetta_basics/file_types/constraint-file. Which constraint might be useful in this case?

In the input folder an empty constraint file has been depsoited. Copy it do your working directory and use the information on the Rosetta constraint documentation to make your own constraint file. Use a text editor of your liking and save your constraint file. In case, you struggle, or your constraint files fails to run, an examplary constraint file with constraint information as been deposited in the output_files folder. You can also look at it to compare your chosen constraint to the example constraint.

```
cp ../input_files/constraint_part_2.cst .
```

To allow Rosetta faster to find a good starting spot, you also might want to use the tools we used before in PyMOL and move the variable Fragment in respect to the NEAT2 domain. *Do not move the NEAT2 domain!* Again, we run a short docking, this time though, we will parse our constraint file:

```
/path_to_rosetta/main/source/bin/rosetta_scripts.default.linuxgccrelease \
  -s NEAT2_STAU-281_0001.pdb NEAT2_STAU-281_0002.pdb NEAT2_STAU-281_0003.pdb \
  -parser:protocol docking_full.xml -out:prefix cst_2_ \
  -constraints:cst_file constraint_part_2.cst @docking.options
```

Analyze the output just as we did above by calculating the RMSD using

```
/path_to_rosetta/main/source/bin/rosetta_scripts.default.linuxgccrelease -s cst_2_*.pdb \
-parser:protocol RMSD_metric.xml -in:file:native complex.pdb -out:file:scorefile cst_2_RMSD.sc
```

Does the RMSD decrease?

3. Using mutagenesis data

A collaborator that has been investigating the interactions of STAU-281 with the NEAT2 domain analyzed the effect of mutations in the NEAT2 domain on antibody binding. The collaborator shares the data with you and asks whether you can create a model using the experimental data. The following mutation abrogates STAU-281 binding: T98A

Note, that this does not exclude other residues from binding, or might be the only interaction partner. Also, take into consideration that you might want to combine the information we have from part 2 (HCDR3 residues) with this new information.

But, we can use this information in our docking protocol. It is useful in docking simulations to move your proteins in a starting position close to the expected interface, as this also minimizes the sampling space till Rosetta starts finding preferable binding poses. Especially, as we are only running 3 models, it might take time and multiple rounds of docking till Rosetta identifies the correct binding pose (in case you want to try that you would take the output models from your runs and input them as starting models - over time you would see how the constraint forces Rosetta to except more and more poses that honor your constraint). You can use the tools in PyMol as discussed above to move the antibody to starting position closer to our experimental restraints. *Do not move the NEAT2 domain!*

Copy the empty constraint file from the input folder:

```
cp ../input_files/constraint_part_3.cst .
```

Go to the RosettaCommons documentation and create a constraint file. In case you struggle to find a good constraint file, you find a pre-generated constraint file in the output folder.

Again, we will execute docking and RMSD calculations using:

```
/path_to_rosetta/main/source/bin/rosetta_scripts.default.linuxgccrelease \
  -s NEAT2_STAU-281_0001.pdb NEAT2_STAU-281_0002.pdb NEAT2_STAU-281_0003.pdb \
  -parser:protocol docking_full.xml -out:prefix cst_3_ \
  -constraints:cst_file constraint_part_3.cst @docking.options
```

and

```
/path_to_rosetta/main/source/bin/rosetta_scripts.default.linuxgccrelease -s cst_3_.*pdb \
  -parser:protocol RMSD_metric.xml -in:file:native complex.pdb -out:file:scorefile cst_3_RMSD.sc
```

Look at your output and decide whether you are getting closer in sampling the right conformational space?

4. Using even more data

Your collaborator further investigates the binding of STAU-281 to NEAT2 and finds out that first STAU-281 occupies the heme binding pocket in NEAT2 and that one of the most crucial interactions parter is F169 in the heavy chain of the antibody. You can find a NEAT2 protein with a heme bound from another crystal structure in your input_files folder. It is named 3rur.pdb. You can look at it and identify residues that seem to be crucial for an interaction. You can use them as a constraint. You should further incoporate residue F169 in you constraint file.

Copy the constraint file from you inout folder to your working directory and start creating some new restraints. Maybe you also want to move the antibody fragment again to bring it in a better starting position. Do so my executing the following commands:

```
cp ../input_files/constraint_part_4.cst .

/path_to_rosetta/main/source/bin/rosetta_scripts.default.linuxgccrelease \
  -s NEAT2_STAU-281_0001.pdb NEAT2_STAU-281_0002.pdb NEAT2_STAU-281_0003.pdb \
  -parser:protocol docking_full.xml -out:prefix cst_4_ \
  -constraints:cst_file constraint_part_4.cst @docking.options

/path_to_rosetta/main/source/bin/rosetta_scripts.default.linuxgccrelease -s cst_4_*.pdb \
  -parser:protocol RMSD_metric.xml -in:file:native complex.pdb -out:file:scorefile cst_4_RMSD.sc
```

We have no provided an example for this constraint. You should be able to use the constraint documentation by now to customize your constraint files.

Again, check your RMSD and whether it improved over the docking runs we executed before. At this point you can also check the complex.pdb structure that serves as control. As you see here, the antibody fragment actually binds NEAT2 heme binding pocket mostly through its HCDR2. Antibodies as STAU-281 are potential new therapeutics against bacterial diseases as infections from *Staphylococcus aureus*, which is a hospitalization germ that has acquired a number of resistances against common antibiotics.

A note of caution for the usage of constraints: Using too many constraints might restrict your sampling space too much and prevent you from sampling the right solution. Constraints should therefore always be applied with caution and only when experimental data warrant a constraint for your modeling.

Thank you for doing this tutorial! I hope you learned a lot and are ready to work with constraints and your own experimental data! Cheers!