PROTOCOL CAPTURE:

ROSETTATMH: AN INNOVATIVE METHOD FOR MEMBRANE PROTEIN STRUCTURE ELUCIDATION COMBINING EPR DISTANCE RESTRAINTS WITH ASSEMBLY OF TRANSMEMBRANE HELICES

This document contains the protocol capture for the modeling performed for the publication of the same title by Stephanie DeLuca, Sam DeLuca, and Jens Meiler. For this particular publication, input files for folding were generated in Rosetta version 3.4 (1). Rosetta revision numbers d592380 and d7b5a70 were used for RosettaTMH parameter optimization and benchmarking, respectively. For the purpose of this protocol capture, it is recommended that the user download weekly release 2014_28_57011, which is available at www.rosettacommons.org. Command lines and inputs use 1U19 as an example and are based on using Rosetta 3.4 or weekly release 2014_28_57011, which were the versions used for the protocol in the Stephanie DeLuca's 2015 dissertation (http://etd.library.vanderbilt.edu/available/etd-03102015-221044/). Examples of files are found in the listed directories in the Protocol Capture in an `examples` directory.

Information on how this input was generated can be found in the publication's Supplemental Information. Additionally, EPR restraint simulation, some structural analysis, etc., were performed using the BioChemical Library (BCL) version 3.1.0, revision 4753 or later. Please visit www.meilerlab.org/bclcommons for more information on licensing and installation.

Outline

I. Make or prepare input files

A. Generate a FASTA file for the target sequence.

A FASTA file is a text file that contains a header line, which consists of the name of the protein, followed by the amino acid sequence of the protein on a separate line; this is indicated below:

```
# protocol_capture/input/1U19A.fasta
1U19A
>BCL :A|PDBID|CHAIN|SEQUENCE
EPWQFSMLAAYMFLLIMLGFPINFLTLYVTVQHKKLRTPLNYILLNLAVA
DLFMVFGGFTTTLYTSLHGYFVFGPTGCNLEGFFATLGGEIALWSLVVLA
IERYVVVCKPMSNFRFGENHAIMGVAFTWVMALACAAPPLVGWSRYIPEG
MQCSCGIDYYTPHEETNNESFVIYMFVVHFIIPLIVIFFCYGQLVFTVKE
AAAQQQESATTQKAEKEVTRMVIIMVIAFLICWLPYAGVAFYIFTHQGSD
FGPIFMTIPAFFAKTSAVYNPVIYIMMN
```

B. Make fragment files

There are multiple ways of creating fragment files. Users from non-profit institutions can use the Robetta server (example below). Industry users will need to refer to the documentation at the RosettaCommons. For more information, please see Combs, et al., Nature Protocols, 2013 *(2)*.

1. **Register for a username and password at the Robetta web server**
http://robetta.bakerlab.org/fragmentsubmit.jsp

2. Input the sequence name, and load the target FASTA file from Step IA.

3. Submit the FASTA file.
The webpage will reload and state 'Successfully added your request to the queue.' The status of the fragment file generation can be checked at http://robetta.bakerlab.org/fragmentqueue.jsp.

4. Get files
Click the link to get a list of files generated by Robetta after the status has changed to 'Complete.' If you are following the example of 1U19, the fragment files should be called aa1U19A003_05.200_v1_3 for fragments of length 3 and aa1U19A009_05.200_v1_3 for fragments of length 9. Save all the files to the working directory by right-clicking and selecting 'Save as.' Files in `protocol_capture/input/`

C. Generate spanfile

The membrane span information can come from the source of your choosing but the format for use with Rosetta should look like the below example.

```
TM region prediction for protocol_capture/input/1U19A.octo_topo predicted using
OCTOPUS
7 278
antiparallel
n2c
   7    27     7    27
  43    63    43    63
  82   102    82   102
```

```
 120   140   120   140
 171   191   171   191
 221   241   221   241
 255   275   255   275
```

1. Get membrane spanning prediction from OCTOPUS.
Go to http://octopus.cbr.su.se/index.php. Paste the FASTA file from Step 1A into the server's text box and click "Submit OCTOPUS."

2. Convert the output from OCTOPUS to the Rosetta spanfile format

```
rosetta-3.4/rosetta_source/src/apps/public/membrane_abinitio/octopus2span.pl
1U19A.octo_topo > 1U19A.span
# File in protocol_capture/input/1U19A.span
```

C. Generate lipophilicity file

The spanfile (above) is used as used to create a lipophilicity file, which has the format seen below.

```
rosetta-3.4/rosetta_source/src/apps/public/membrane_abinitio/run_lips.pl 1U19A.fasta
1U19A.span my_dir/blastpgp rosetta-
3.4/rosetta_source/src/apps/public/membrane_abinitio/alignblast.pl >& log &

Lipid exposed data: resnum mean-lipo lipophil entropy
     12  -1.000   2.302   3.334
     15  -1.000   3.410   3.819
     16  -1.000   3.064   6.190
     19  -1.000   0.894   2.784
     22  -1.000   2.158   4.783
     23  -1.000   0.382   1.011
     26  -1.000   1.748   2.160
      8  -1.000   2.230   5.054
      9  -1.000   1.424   5.436
      ….
# File in protocol_capture/input/1U19A.lips4
```

D. Generate Topology Broker "rigid" files for computing $RMSD_{100}SSE$

Taken from secondary structure element definitions from DSSP and are similar to that used for evaluation of models in reference (1). The `RIGID` definitions indicate the beginning and end points of the region of the protein over which $RMSD_{100}$ will be computed. Residues over which $RMSD_{100}SSE$ was computed. Only applicable if there is a structure to which the model can be compared, and the structure has the same residue numbering. Assume the 1U19 sequence starts at residue 1, as are all models folded with Rosetta. In a text editor, create the following file:

```
# 1U19A
RIGID  2     32
RIGID  39    68
RIGID  75    108
RIGID  118   140
RIGID  168   193
RIGID  210   245
RIGID  253   277

# File in protocol_capture/input/1U19A_sse_defs.txt
```

E. Make Topology Broker setup files

In a text editor, create the following files.

1. Using 1U19A as an example

```
# folding from an extended chain
# protocol_capture/input/broker_setup/extended_epr/1U19A_s00_b000_0000.cst.tpb
CLAIMER MembraneTopologyClaimer
END_CLAIMER
```

2. Folding from an extended chain with using EPR restraints (extended chain + EPR)

```
# protocol_capture/input/broker_setup/extended_epr/1U19A_s01_b001_0000.cst.tpb
CLAIMER MembraneTopologyClaimer
END_CLAIMER
CLAIMER ConstraintClaimer
FILE protocol_capture/input/1U19A_s01_b001_0000.cst
END_CLAIMER
```

3. Folding with RosettaTMH

```
# protocol_capture/input/broker_setup/tmh_epr/1U19A_s00_b000_0000.cst.tpb
CLAIMER MembraneTopologyClaimer
END_CLAIMER
CLAIMER TMHTopologySamplerClaimer
END_CLAIMER
```

4. Folding with RosettaTMH with EPR restraints (RosettaTMH + EPR)

```
# protocol_capture/input/broker_setup/tmh_epr/1U19A_s01_b001_0000.cst.tpb
CLAIMER MembraneTopologyClaimer
END_CLAIMER
CLAIMER TMHTopologySamplerClaimer
END_CLAIMER
CLAIMER ConstraintClaimer
FILE protocol_capture/input/1U19A_s01_b001_0000.cst
END_CLAIMER
```

F. Simulate EPR distance restraints using the BCL

1. Convert Rosetta native PDB file to BCL format

```
# NOT NECESSARY if have PDB file already in BCL format (pdb_bcl.pdb)
# protocol_capture/input/simulate_cst/1U19A_bcl.pdb
bcl.exe protein:PDBConvert 1U19A.pdb -bcl_pdb -output_prefix 1U19A_ >& 1U19A_bcl.log
```

2. Make mutate.wts file in a text editor

```
# protocol_capture/input/simulate_cst/mutate.wts
# mutate.wts for generating EPR distance restraints using the BCL
bcl::storage::Table<double>    add_all add_single filter_aa_type_excl filter_sse_size
remove_single       swap distance_range_0 filter_exposure_0
weights      0      1      0      0      1      1      0      0
```

3. Make score.wts file in a text editor

```
# protocol_capture/input/simulate_cst/score.wts
# score.wts for generating EPR distance restraints using the BCL
```

```
bcl::storage::Table<double> data_density aa_type_excl    seq_sep data_set_size
sse_connection   sse_size   sse_term    bipolar sse_center triangulation_0
distance_range_0 exposure_0
weights 0    0     1     1     1     0     0     0     0     0     10000  10000
```

4. Make SSE pool, which is required for generating restraints in BCL.

Make file in a text editor.

```
# protocol_capture/input/simulate_cst/1U19A_native.pool
# SSE pool for 1U19A – required to run bcl.exe restraint:OptimizeDataSetPairwise
# All pool files based on spanfiles (TMH definitions)
bcl::assemble::SSEPool
HELIX    1   1 PRO A    2  GLN A   32  1                                  31
HELIX    2   2 PRO A   39  HIS A   68  1                                  30
HELIX    3   3 GLY A   74  VAL A  107  1                                  34
HELIX    4   4 GLU A  118  LEU A  140  1                                  23
HELIX    5   5 ASN A  168  GLN A  193  1                                  26
HELIX    6   6 THR A  210  THR A  245  1                                  36
HELIX    7   7 PRO A  253  MET A  276  1                                  24
END
```

5. Pick EPR distance restraints using the BCL

```
# command line for restraint picking
# BCL v3.1.0, r4753, compiled on Mon Feb 10 03:23:31 2014
bcl.exe restraint:OptimizeDataSetPairwise -fasta protocol_capture/input/1U19A.fasta -
pool_min_sse_lengths 3 0 -pool protocol_capture/input/simulate_cst/1U19A_native.pool
-distance_min_max 10 50 -nc_limit 10 -ensembles pdb.ls -mc_number_iterations 10000
10000 -prefix 1U19A -nmodels 10 -read_scores_optimization
protocol_capture/input/simulate_cst/score.wts -read_mutates_optimization
protocol_capture/input/simulate_cst/mutate.wts -read_mutates_start
protocol_capture/input/simulate_cst/mutate.wts -message_level Standard -pymol_output
-data_set_size_range 10 40 -data_set_size_fraction_of_sse_resis 0.2 >& log &
```

WARNING: scores.wts and mutate.wts in demo are different because some scores do not exist in more recent versions of BCL. Namely, filter_sse size (mutate.wts) and sse_connect, sse_size, sse_term, bipolar, sse_center (score.wts). If using a newer version of the BCL to simulate EPR distance restraints, contact Jens Meiler (jens.meiler@vanderbilt.edu) for best practices.

6. Add spin label uncertainty

```
# BCL v3.1.0, r4753, compiled on Mon Feb 10 03:23:31 2014
bcl.exe SimulateDistanceRestraints -pdb
protocol_capture/input/simulate_cst/1U19A_bcl.pdb -simulate_distance_restraints -
output_file protocol_capture/input/simulate_cst/1U19A_s01_b001_0000.cst -
add_distance_uncertainty protocol_capture/input/simulate_cst/sl-
cb_distances.histograms -restraint_list
protocol_capture/input/simulate_cst/1U19A_final_0000.data 0 1 5 6 -random_seed
0123456 -write_rosetta_mini_restraints
```

7. Convert restraint files to fold membrane proteins in Rosetta

```
# example using 1U19A from within protocol_capture/input/simulate_cst directory
# original cst file name:  1U19A_final_0000.cst; final name:  1U19A_s01_b001_0000.cst
protocol_capture/input/simulate_cst/1U19A_make_mp_cst_file.sh
```

8. Replace HA2 to 2HA

```
sed -i 's/HA2/2HA/g' 1U19A_s01_b001_0000.cst
```

```
# Rosetta restraint file format
# weighting EPR KBP by 10.0 and quadratic penalty by 1.0
# if have Gly in AtomPair, replace CB with 1HA or 2HA
AtomPair CB 67 CB 255 SCALARWEIGHTEDFUNC 1.0 SPLINE EPR_DISTANCE  28.9577 1.0 0.5
AtomPair CB 67 CB 255 SCALARWEIGHTEDFUNC 1.0 BOUNDED 16.9577 40.9577 1.0 NOE ;dist
```

G. Prepare files for comparing Rosetta and BCL models

Make loops file (for the purposes of measuring RMSD between BCL and Rosetta models)
For comparison with BCL-generated models, will need to build and close loops on Rosetta models.

1. Building loops on BCL and RosettaTMH files

```
# protocol_capture/input/1U19A.loops
# Loops file format – residues defined in loops are all residues not covered by spanfile
# EXAMPLE:  1U19A
LOOP   1      7
LOOP   27     43
LOOP   63     82
LOOP   102    120
LOOP   140    171
LOOP   191    221
LOOP   241    255
LOOP   275    278
```

2. Convert BCL files to Rosetta files and make loops files for Rosetta loop building

```
# BCL v3.1.0, r4753, compiled on Mon Feb 10 03:23:31 2014
# from the protocol_capture/build_loops directory:
bcl.exe protein:PDBConvert protocol_capture/input/1U19A.pdb -loop_file_rosetta CCD -write_zero_coordinates -bcl_pdb Split >& 1U19A_loops.log
```

3. Build loops in Rosetta (with options file from a PDB file

See `protocol_capture/input/fill_gaps.options`

```
# used to build loops on RosettaTMH models and BCL models
# weekly release 2014_28_57011
# example in protocol_capture/build_loops
Rosetta/main/source/bin/loopmodel.default.linuxgccrelease                    -database Rosetta/main/database/      @protocol_capture/input/fill_gaps.options      -in:file:l protocol_capture/build_loops/pdb.ls     -out:pdb_gz     -out:prefix     1U19A_     -out:no_nstruct_label -out:file:scorefile 1U19A_build_loops.sc >& log &
```

Restraint weights for folding in Rosetta (CST_WEIGHT in options file)

```
pdb    weight
1U19A  10.9355
```

H. Make score patch files

Make the following score patch files in a text editor.

1. score_membrane_s2.wts_patch

```
pair = 0.0
Mpair = 1.0
env = 0.0
```

```
Menv = 2.019
cbeta = 0.0
Mcbeta = 0.0
Menv_non_helix = 2.019
Menv_termini = 2.019
Menv_tm_proj = 2.019
Mlipo = 1.0
```

2. score_membrane_s3a.wts_patch

```
pair = 0.0
Mpair = 1.0
env = 0.0
Menv = 2.019
cbeta = 0.0
Mcbeta = 0.5
Menv_non_helix = 2.019
Menv_termini = 2.019
Menv_tm_proj = 2.019
Mlipo = 1.0
```

3. score_membrane_s3b.wts_patch

```
pair = 0.0
Mpair = 1.0
env = 0.0
Menv = 2.019
cbeta = 0.0
Mcbeta = 0.5
Menv_non_helix = 2.019
Menv_termini = 2.019
Menv_tm_proj = 2.019
Mlipo = 1.0
```

4. score_membrane_s4.wts_patch

```
pair = 0.0
Mpair = 1.0
env = 0.0
Menv = 2.019
cbeta = 0.0
Mcbeta = 2.5
Menv_non_helix = 2.019
Menv_termini = 2.019
Menv_tm_proj = 2.019
Mlipo = 1.0
```

I. Generate options files (in a text editor)

Generate Options files for *de novo* folding in Rosetta. Assume the options file is in `protocol_capture/input`

1. MembraneAbinitio

```
-in
      -file
            -native protocol_capture/input/1U19A.pdb
            -fasta protocol_capture/input/1U19A.fasta
            -frag3 protocol_capture/input/aa1U19A03_05.200_v1_3
```

```
            -frag9 protocol_capture/input/aa1U19A09_05.200_v1_3
            -spanfile protocol_capture/input/1U19A.span
            -lipofile protocol_capture/input/1U19A.lips4
-residues
      -patch_selectors CENTROID_HA
-score
      -find_neighbors_3dgrid
#     -use_membrane_rg  #### use this flag if using MP-specific RG score
-membrane
      -no_interpolate_Mpair
      -Menv_penalties
-abinitio
      -membrane
      -explicit_pdb_debug # if want to output at stages 0-4
      -rg_reweight 4.25
      -stage2_patch protocol_capture/input/score_membrane_s2.wts_patch
      -stage3a_patch protocol_capture/input/score_membrane_s3a.wts_patch
      -stage3b_patch protocol_capture/input/score_membrane_s3b.wts_patch
      -stage4_patch protocol_capture/input/score_membrane_s4.wts_patch
-evaluation
      -gdtmm
      -rmsd NATIVE _tm_sse protocol_capture/input/1U19A_sse_defs.txt
-out
      -output
      -file
            -output_virtual
            -silent_struct_type binary
-overwrite
```

2. Extended chain (and extended chain + EPR when CST_WEIGHT ≠ 0.0)

```
# assume the options file is in the protocol_capture/input directory
-in
      -file
            -native protocol_capture/input/1U19A.pdb
            -fasta protocol_capture/input/1U19A.fasta
            -frag3 protocol_capture/input/aa1U19A03_05.200_v1_3
            -frag9 protocol_capture/input/aa1U19A09_05.200_v1_3
            -spanfile protocol_capture/input/1U19A.span
            -lipofile protocol_capture/input/1U19A.lips4
-residues
      -patch_selectors CENTROID_HA
-broker
      -setup
protocol_capture/input/broker_setup/extended_epr/1U19A_s00_b000_0000.cst.tpb ####
will follow format of broker setup file above
# protocol_capture/input/broker_setup/extended_epr/1U19A_s01_b001_0000.cst.tpb if
using EPR restraints
-run
      -protocol broker
-score
      -find_neighbors_3dgrid
-membrane
      -no_interpolate_Mpair
      -Menv_penalties
-abinitio
```

```
	-membrane
	-explicit_pdb_debug # if want to output at stages 0-4
	-rg_reweight 4.25
	-stage2_patch protocol_capture/input/score_membrane_s2.wts_patch
	-stage3a_patch protocol_capture/input/score_membrane_s3a.wts_patch
	-stage3b_patch protocol_capture/input/score_membrane_s3b.wts_patch
	-stage4_patch protocol_capture/input/score_membrane_s4.wts_patch
-constraints
	-cst_file protocol_capture/input/1U19A_s00_b000_0000.cst # or
1U19A_s01_b001_0000.cst
	-cst_weight 0.0 # or 10.9355 if folding with restraints
	-epr_distance
-fold_cst
	-force_minimize
	-seq_sep_stages 1.0 1.0 1.2
-evaluation
	-gdtmm
	-rmsd NATIVE _tm_sse protocol_capture/input/1U19A_sse_defs.txt
-out
	-output
	-file
		-output_virtual
		-silent_struct_type binary
-overwrite
```

3. RosettaTMH (and RosettaTMH + EPR when cst_weight ≠ 0.0)

```
# assume the options file is in the protocol_capture/input directory
-in
	-file
		-native protocol_capture/input/1U19A.pdb
		-fasta protocol_capture/input/1U19A.fasta
		-frag3 protocol_capture/input/aa1U19A03_05.200_v1_3
		-frag9 protocol_capture/input/aa1U19A09_05.200_v1_3
		-spanfile protocol_capture/input/1U19A.span
		-lipofile protocol_capture/input/1U19A.lips4
-residues
	-patch_selectors CENTROID_HA
-broker
	-setup protocol_capture/input/broker_setup/tmh_epr/1U19A_s00_b000_0000.cst.tpb
#### will follow format of broker setup file above
# protocol_capture/input/broker_setup/tmh_epr/1U19A_s01_b001_0000.cst.tpb if using
EPR restraints
	-large_frag_mover_stage1_weight 0.0
	-small_frag_mover_stage1_weight 0.0
	-rb_mover_stage1_weight 5.0
-run
	-protocol broker
-score
	-find_neighbors_3dgrid
#	-use_membrane_rg #### use this flag if using MP-specific RG score
-membrane
	-fixed_membrane
	-no_interpolate_Mpair
	-Menv_penalties
-abinitio
```

```
	-membrane
	-explicit_pdb_debug # if want to output at stages 0-4
	-rg_reweight 4.25
	-stage2_patch protocol_capture/input/score_membrane_s2.wts_patch
	-stage3a_patch protocol_capture/input/score_membrane_s3a.wts_patch
	-stage3b_patch protocol_capture/input/score_membrane_s3b.wts_patch
	-stage4_patch protocol_capture/input/score_membrane_s4.wts_patch
-constraints
	-cst_file protocol_capture/input/1U19A_s00_b000_0000.cst # or
1U19A_s01_b001_0000.cst
	-cst_weight 0.0 # or 10.9355
	-epr_distance
-fold_cst
	-force_minimize
	-seq_sep_stages 1.0 1.0 1.2
-rigid
	-rotation 0.1
	-translation 0.5
-evaluation
	-gdtmm
	-rmsd NATIVE _tm_sse protocol_capture/input/1U19A_sse_defs.txt
-out
	-output
	-file
		-output_virtual
		-silent_struct_type binary
-overwrite
```

4. Lμμμoop building onto BCL and RosettaTMH models

```
# (from the protocol_capture/build_loops directory)
-in
	-file
		-native protocol_capture/input/1U19A.pdb
		-spanfile protocol_capture/input/1U19A.span
		-lipofile protocol_capture/input/1U19A.lips4
		-residue_type_set centroid
-chemical
	-patch_selectors CENTROID_HA
-score
	-find_neighbors_3dgrid
-evaluation
	-rmsd NATIVE _tm_sse protocol_capture/input/1U19A_sse_defs.txt
-membrane
	-no_interpolate_Mpair
	-Menv_penalties
-loops
	-loop_file protocol_capture/input/1U19A.loops
	-frag_sizes 9 3 1
	-frag_files  protocol_capture/input/aa1U19A09_05.200_v1_3
protocol_capture/input/aa1U19A03_05.200_v1_3 none
	-remodel quick_ccd
	-cen_weights score_membrane
	-cen_patch protocol_capture/input/score_membrane_s4.wts_patch
-out
	-output
```

```
	-no_nstruct_label
	-nstruct 1
	-file
		-silent_struct_type binary # only if outputting silent files, not pdb or
pdb_gz files
		-residue_type_set centroid
-overwrite
```

II. Command lines for *de novo* folding

The following command lines are used to de novo fold membrane proteins using Rosetta. The revision numbers are indicated.

A. MembraneAbinitio

```
# Rosetta revision numbers d592380 and d7b5a70.
# For 1U19A, takes approximately 15 minutes
Rosetta/main/source/bin/membrane_abinitio2.default.linuxgccrelease -database
Rosetta/main/database/ @protocol_capture/input/memabrlx_no_cst.options -out::nstruct
1 -out:file:silent protocol_capture/output/1U19A_memabrlx_no_cst.out -out:sf
protocol_capture/output/1U19A_memabrlx_no_cst.sc
```

B. Extended chain (and Extended chain + EPR if CST_WEIGHT ≠ 0.0)

```
# Rosetta revision numbers d592380 and d7b5a70.
# Note that for rmsd values, look in the silent file, not the score file
# For 1U19A with no restraints, takes approximately 3 minutes
# For 1U19A with restraints, takes approximately 9.5 minutes
Rosetta/main/source/bin/minirosetta.default.linuxgccrelease -database
Rosetta/main/database/ @protocol_capture/input/mp_extended.options -out::nstruct 1 -
out:file:silent protocol_capture/output/1U19A_extended_no_cst.out -out:file:scorefile
protocol_capture/output/1U19A_extended_no_cst.sc >&
protocol_capture/output/extended_no_cst.log &
```

C. RosettaTMH (and RosettaTMH + EPR if CST_WEIGHT != 0.0)

```
# Rosetta revision numbers d592380 and d7b5a70.
# Note that for rmsd values, look in the silent file, not the score file
# For 1U19A with no restraints, takes approximately 75 seconds
# For 1U19A with restraints, takes approximately 5 minutes
Rosetta/main/source/bin/minirosetta.default.linuxgccrelease -database
Rosetta/main/database/ @protocol_capture/input/mp_rosettatmh.options -out::nstruct 1
-out:file:silent protocol_capture/output/1U19A_rosettatmh_no_cst.out -
out:file:scorefile protocol_capture/output/1U19A_rosettatmh_no_cst.sc >&
protocol_capture/output/1U19A_rosettatmh_no_cst.log &
```

III. Analysis of Results

The following file formats are used for input. Command lines were used for analysis.

A. Generate of $RMSD_{100}$SSE histograms

1. File format for input

```
# format for files for input into rmsd_to_rmsd100.py, which is found in the
protocol_capture/scripts directory
# doesn't matter how many fields are between field 1 and description, but "SCORE:"
must be first, and "description" last, also total score is "score"
# example is protocol_capture/analysis/1U19A_s01_b001_0000_example.sc
SCORE:     score    atom_pair_constraint    rms_tm_sse       description
SCORE:  -124.890               -368.904        17.8586           S_0001
SCORE:    96.155               -329.637        16.3458           S_0002
SCORE:   -71.851               -372.237        13.7901           S_0003
SCORE:  -149.703               -360.364        15.2572           S_0004
SCORE:  -169.673               -364.787        17.3802           S_0005
```

2. convert Rosetta-computed RMSD values to $RMSD_{100}$

```
# type rmsd_to_rmsd100.py -h for commandline options
protocol_capture/scripts/rmsd_to_rmsd100.py --membrane --silent
protocol_capture/analysis/1U19A_s01_b001_0000_example.sc -n 278 --outfile
protocol_capture/analysis/1U19A_s01_b001_0000_example.sc.RMSDSSE100 --rms_tag=tm_sse
```

3. Generate histograms and summary

```
perl protocol_capture/scripts/Smbins_RMSD_dist_from_score.pl
protocol_capture/analysis/1U19A_s01_b001_0000_example.sc.RMSDSSE100 5 | awk
'{print($2"\t"$4)}' | head -n21 >
protocol_capture/analysis/1U19A_s01_b001_0000_example.sc.rmsdSSE100.txt
```

B. Convert Rosetta silent files to PDB files

```
Rosetta/main/source/bin/format_converter.default.linuxgccrelease            -database
Rosetta/main/database                                                  -in:file:silent
protocol_capture/output/1U19A_rosettatmh_cst.out -in:file:silent_struct_type binary -
in:file:residue_type_set centroid -out:pdb -out:file:residue_type_set centroid
```

C. Calculating contact order

1. Downloaded script

Get the script from the from Baker laboratory website (http://depts.washington.edu/bakerpg/contact_order/)

2. Run the script

```
# options:  -c = cutoff, default is 6; -a = absolute contact order
protocol_capture/scripts/contactOrder.pl –c 8 –a protocol_capture/output/test.pdb
```

References

1. Weiner BE, Woetzel N, Karakaş M, Alexander N, Meiler J (2013) BCL::MP-fold: folding membrane proteins through assembly of transmembrane helices. *Structure* 21:1107–1117.

2. Combs SA et al. (2013) Small-molecule ligand docking into comparative models with Rosetta. *Nature Protocols* 8:1277–1298.