EvoClustRNA

Marcin Magnus

Oct 30, 2019

Contents

Marcin Magnus & Rhiju Das

The code of the project can be found at GitHub (https://github.com/mmagnus/EvoClustRNA)

A clustering routines of evolutionary conserved regions (helical regions) for RNA fold prediction.

At the moment we are testing the apprach using models from Rosetta FARFAR (https://www.rosettacommons.org/) and SimRNAweb (http://genesilico.pl/SimRNAweb/).

The documentation can be found here http://evoclustrna.readthedocs.io/en/latest/

Contents:

CHAPTER 1

Get Started

(All the code can be executed in the folder evoClustRNA/test_data/rp13 of this repository)

1.1 Prepare a multiple sequence alignment (MSA)

For the target sequence, the user needs to prepare an alignment or download it from the from the Rfam database. The sequence similarity should be reduced, using JalView to keep only diverse representatives. In theory, all sequences could be folded but because of the computational costs of simulations (6-10h per sequence for 80 CPUs, using either SimRNAweb or Rosetta FARFAR), we decided to fold only 4 the shortest sequences from the MSA. Once the final set of homologs to be folded was selected, the positions common to all sequences selected were determined.

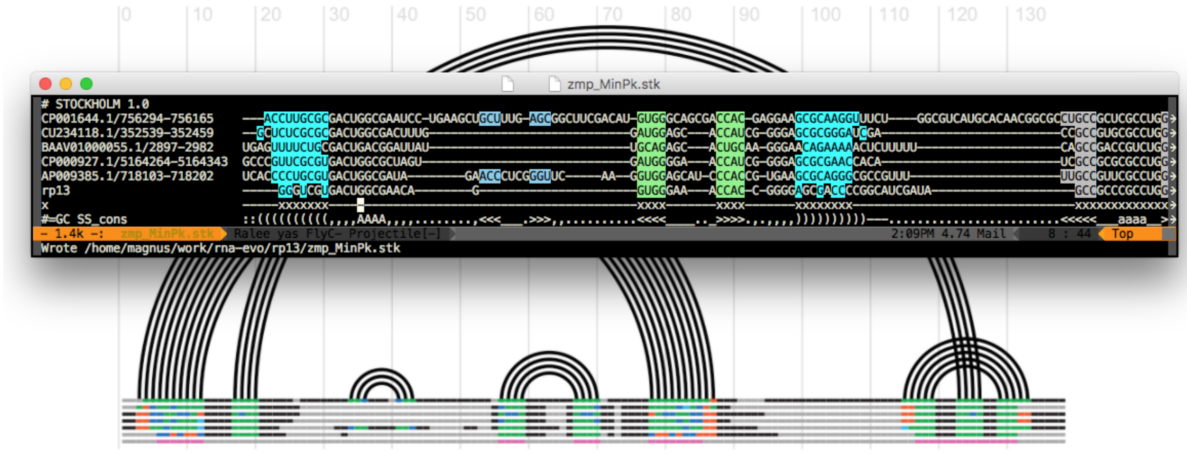

Figure 1. **The alignment preparation.** The conserved residues are marked with "x" in the pseudo-sequence "x". The marked as the conserved residues columns can be inspected in an arc diagrams of RNA secondary structures as the pink line (at the very bottom)

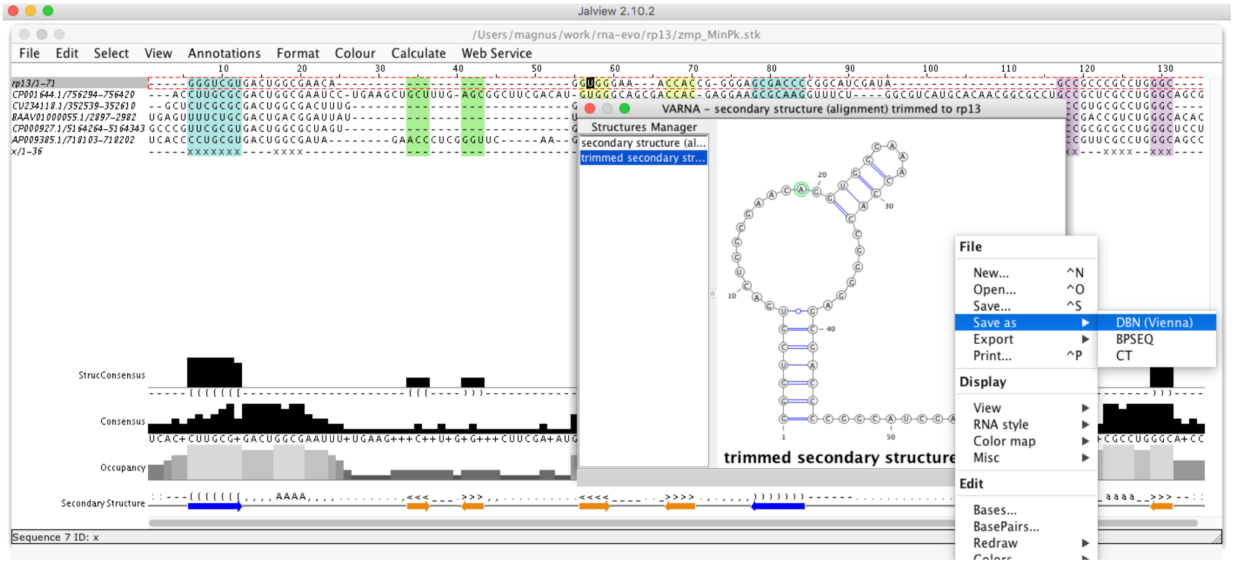

Figure 2. **Generation sequence and secondary structure**. Each sequence and associated secondary structure was "Saved as" to a Fasta file and used at the next stage of modeling with the use of the Jalview program.

1.2 RNA 3D structure prediction to generate initial models

For each sequence chosen for folding, the user must prepare an input for RNA 3D structure prediction method. Two methods were used in this study: SimRNA and Rosetta. For Rosetta, a total of 10,000 decoys were generated for the target sequence and each homologous sequence using the Rosetta FARFAR protocol. For SimRNA prediction, SimRNAweb (https://genesilico.pl/SimRNAweb/) server was used using the default parameters.

To start with the EvoClustRNA protocol, we suggest using SimRNAweb first. The results are comparable to Rosetta, but the server is much easier to use for beginners.

At the end of modeling, top100 (or top200) models have to be extracted and copied into the `structures` folder.

1.3 Run EvoClustRNA clustering procedure (including extraction of conserved motifs)

Run `evoClustRNA.py` on an alignment (-a) and a folder with structures (-i) using mapping (-m) and flat directory structure:

```
[mm] evox$ git:(master)  evoClustRNA.py -a ../rp13finalX_noSSperSeq_ref.sto -i␣
↪structures -m ../mapping_ref.txt -f
 \_ evoClustRNA  Namespace(flat_dir=True, inf=False, input_dir='structures', mapping_
↪fn='../mapping_ref.txt', matrix_fn='', output_dir='out', rna_alignment_fn='../
↪rp13finalX_noSSperSeq_ref.sto', save=False, verbose=False)
rp13finalX_noSSperSeq_ref_mapping_refX.matrix
 # of rnastruc: 6
 rnastruc: ['rp13:tar_', 'rp13:solution', 'cp0016:zcp', 'nc9445:znc', 'nc3295:zc3',
↪'nzaaox:zza']
  rp13 <-> tar_
   cutting out fragments ...
   analyzing... structures/*tar_*.pdb
```

(continues on next page)

(continued from previous page)

```
   # of structures 200
  rp13 <-> solution
   cutting out fragments ...
   analyzing... structures/*solution*.pdb
   # of structures 1
  cp0016 <-> zcp
   cutting out fragments ...
   analyzing... structures/*zcp*.pdb
   # of structures 200
  nc9445 <-> znc
   cutting out fragments ...
   analyzing... structures/*znc*.pdb
   # of structures 200
  nc3295 <-> zc3
   cutting out fragments ...
   analyzing... structures/*zc3*.pdb
   # of structures 200
  nzaaox <-> zza
   cutting out fragments ...
   analyzing... structures/*zza*.pdb
   # of structures 200
 # of models: 1001
matrix was created!  rp13finalX_noSSperSeq_ref_mapping_refX.matrix
evoClustRNA.py -a ../rp13finalX_noSSperSeq_ref.sto -i structures -m ../mapping_ref.
↪txt -f
```

`rp13finalX_noSSperSeq_ref_mapping_refX.matrix` is the matrix with all-vs-all RMSDs for all conserved motifs.

Now it's time to cluster the matrix:

```
evoClust_autoclustix.py  rp13finalX_noSSperSeq_ref_mapping_refX.matrix
# of struc 1001
evoClust_clustix.py rp13finalX_noSSperSeq_ref_mapping_refX.matrix -c 0
n:  0
rm rp13finalX_noSSperSeq_ref_mapping_refX*cf0*.out # auto-removal
evoClust_clustix.py rp13finalX_noSSperSeq_ref_mapping_refX.matrix -c 0.5
n:  1
(....)
rm rp13finalX_noSSperSeq_ref_mapping_refX*cf8.5*.out # auto-removal
evoClust_clustix.py rp13finalX_noSSperSeq_ref_mapping_refX.matrix -c 9.0
n:  166
```

When the clustering is done, the best clusters can be obtained and copied to two folders for futher analysis: `reps` and `reps_ns`.

Copy the best cluster medoids from `structures` to reps

```
[mm] evox$ git:(master)  evoClust_get_models.py -i structures/ *.out -u
evoClust_get_models.py
--------------------------------------------------------------------------------
1_tar_min.out.1.pdb
2_zcp_min.out.8.pdb
3_tar_min.out.66.pdb
4_tar_min.out.98.pdb
5_tar_min.out.25.pdb
= structures == out/structures/<files>==================
```

(continues on next page)

(continued from previous page)

```
cp -v structures//tar_min.out.1.pdb reps/c1_tar_min.out.1.pdb
structures//tar_min.out.1.pdb -> reps/c1_tar_min.out.1.pdb
cp -v structures//zcp_min.out.8.pdb reps/c2_zcp_min.out.8.pdb
structures//zcp_min.out.8.pdb -> reps/c2_zcp_min.out.8.pdb
cp -v structures//tar_min.out.66.pdb reps/c3_tar_min.out.66.pdb
structures//tar_min.out.66.pdb -> reps/c3_tar_min.out.66.pdb
cp -v structures//tar_min.out.98.pdb reps/c4_tar_min.out.98.pdb
structures//tar_min.out.98.pdb -> reps/c4_tar_min.out.98.pdb
cp -v structures//tar_min.out.25.pdb reps/c5_tar_min.out.25.pdb
structures//tar_min.out.25.pdb -> reps/c5_tar_min.out.25.pdb
```

Copy the best cluster medoids from `structures` to `reps_ns` (this is where only models for the target sequences are stored, so no models of homologs):

```
[mm] evox$ git:(master)  evoClust_get_models.py -i structures/ *.out -n tar -u
evoClust_get_models.py
--------------------------------------------------------------------------------
['tar_min.out.1.pdb', '', 'tar_min.out.66.pdb', 'tar_min.out.98.pdb', 'tar_min.out.25.
↪pdb']
1_tar_min.out.1.pdb
2_
3_tar_min.out.66.pdb
4_tar_min.out.98.pdb
5_tar_min.out.25.pdb
= structures == out/structures/<files>==================
cp -v structures//tar_min.out.1.pdb reps_ns/c1_tar_min.out.1.pdb
structures//tar_min.out.1.pdb -> reps_ns/c1_tar_min.out.1.pdb
cp -v structures// reps_ns/c2_
cp: structures// is a directory (not copied).
cp -v structures//tar_min.out.66.pdb reps_ns/c3_tar_min.out.66.pdb
structures//tar_min.out.66.pdb -> reps_ns/c3_tar_min.out.66.pdb
cp -v structures//tar_min.out.98.pdb reps_ns/c4_tar_min.out.98.pdb
structures//tar_min.out.98.pdb -> reps_ns/c4_tar_min.out.98.pdb
cp -v structures//tar_min.out.25.pdb reps_ns/c5_tar_min.out.25.pdb
structures//tar_min.out.25.pdb -> reps_ns/c5_tar_min.out.25.pdb
```

1.4 Compare to the reference structure

OK, so now we have two folders with models that we can compare to the reference structure.

Various methods can be used to do that. For `reps_ns` (so the models for the reference sequence) you can use full atom RMSD:

```
[mm] evox$ git:(master)  rna_calc_rmsd.py -t ..//*ref.pdb reps_ns/*.pdb
method: all-atom-built-in
# of models: 4
c1_tar_min.out.1.pdb 6.34 1295
c3_tar_min.out.66.pdb 11.6 1295
c4_tar_min.out.98.pdb 15.1 1295
c5_tar_min.out.25.pdb 14.34 1295
# of atoms used: 1295
csv was created!  rmsds.csv
```

core RMSDs based on the alignment:

```
evoClust_calc_rmsd.py -a ..//*ref.sto -t ..//*ref.pdb -n rp13 -m ..//*mapping*ref.txt␣
↪-o rmsd_motif.csv reps/*.pdb
Fri Jun 21 17:01:33 2019
Namespace(debug=False, dont_ignore_clusters=False, files=['reps/c1_tar_min.out.1.pdb',
↪ 'reps/c2_zcp_min.out.8.pdb', 'reps/c3_tar_min.out.66.pdb', 'reps/c4_tar_min.out.98.
↪pdb', 'reps/c5_tar_min.out.25.pdb'], group_name='', mapping_fn='..//mapping_ref.txt
↪', output_fn='rmsd_motif.csv', rna_alignment_fn='..//rp13finalX_noSSperSeq_ref.sto',
↪ target='..//target_13_solution_0_renumber_puzzle_ref.pdb', target_name='rp13')
target: ..//target_13_solution_0_renumber_puzzle_ref.pdb
 # of rnastruc : 6
 rnastruc: ['rp13:tar_', 'rp13:solution', 'cp0016:zcp', 'nc9445:znc', 'nc3295:zc3',
↪'nzaaox:zza']
 WARNING: if any of your PDB file is missing, check mapping!
                                          target                  model   rmsd group_
↪name
0  target_13_solution_0_renumber_puzzle_ref.pdb   c1_tar_min.out.1.pdb   4.41
1  target_13_solution_0_renumber_puzzle_ref.pdb   c2_zcp_min.out.8.pdb  16.08
2  target_13_solution_0_renumber_puzzle_ref.pdb  c3_tar_min.out.66.pdb  10.32
3  target_13_solution_0_renumber_puzzle_ref.pdb  c4_tar_min.out.98.pdb  15.50
4  target_13_solution_0_renumber_puzzle_ref.pdb  c5_tar_min.out.25.pdb  15.24
```

or INFs:

```
[mm] evox$ git:(master)  rna_calc_inf.py -f -t ..//*ref.pdb reps_ns/*.pdb
100% (4 of 4) |#########################################################################
↪#########################################################################
↪################| Elapsed Time: 0:00:16 ETA:  00:00:00csv was created!  inf.csv
[mm] evox$ git:(master)  csv inf.csv
target                                                fn                          inf_
↪all  inf_stack  inf_WC  inf_nWC  sns_WC  ppv_WC  sns_nWC  ppv_nWC
target_13_solution_0_renumber_puzzle_ref.pdb.outCR  c4_tar_min.out.98.pdb.outCR  0.
↪453    0.000      0.923   0.507    0.947   0.900   0.429    0.600
target_13_solution_0_renumber_puzzle_ref.pdb.outCR  c3_tar_min.out.66.pdb.outCR  0.
↪437    0.000      0.947   0.218    0.947   0.947   0.143    0.333
target_13_solution_0_renumber_puzzle_ref.pdb.outCR  c1_tar_min.out.1.pdb.outCR   0.
↪431    0.000      0.973   0.286    0.947   1.000   0.286    0.286
target_13_solution_0_renumber_puzzle_ref.pdb.outCR  c5_tar_min.out.25.pdb.outCR  0.
↪483    0.129      0.947   0.535    0.947   0.947   0.286    1.000
```

1.5 [Example of post-EvoClustRNA analysis]

The results can be also viewed with Clans, shown in the Figure 3 & 4.

In this clustering visualization, 100 models of five homologs are shown (each homolog uniquely colored, models of the target sequence are colored in lime). Models with a pairwise distance in terms of RMSDs lower than 6 Å are connected. The native structure was added to this clustering to see where it would be mapped. Interestingly, the native structure was mapped to the small cluster. In this cluster, there are three models for the target sequence. The model the closest to this the cluster center (Fig. 3B) achieved an RMSD of 6.98 Å to the native structure. This clustering visualization showed that there were models generated with the correct fold, but none of them were selected as the final prediction. The final prediction was the center of the biggest cluster (Fig. 3C).

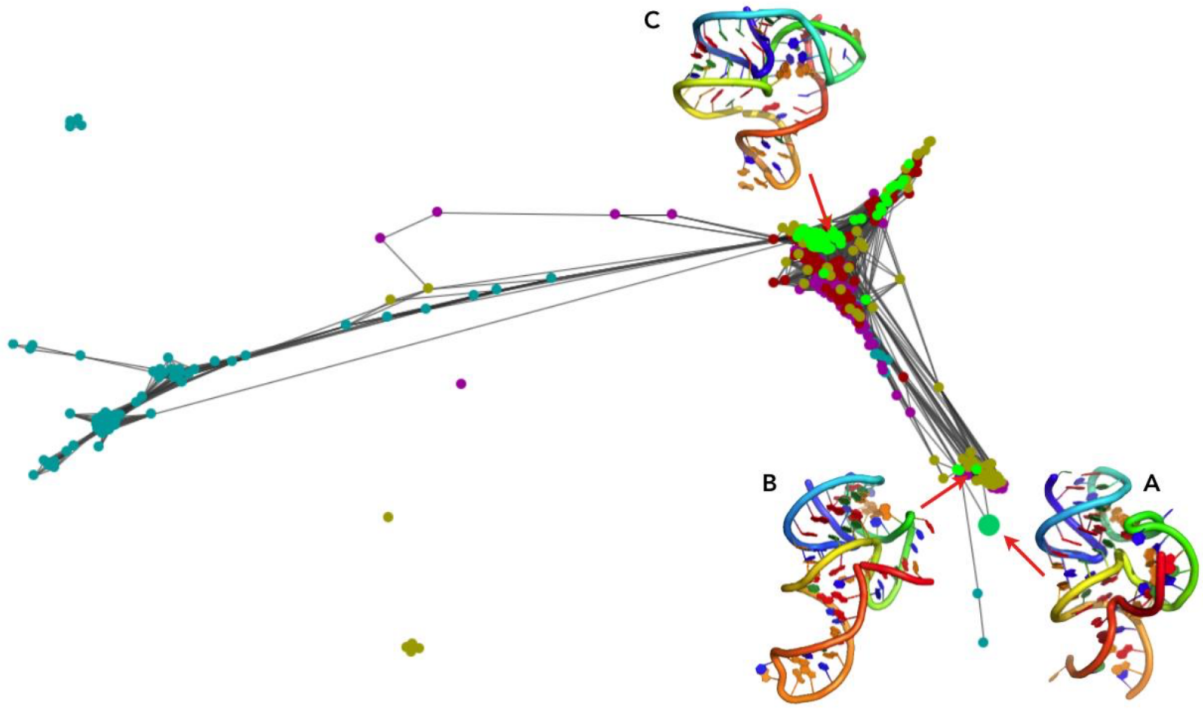

Figure 3. **Clustering visualized with Clans for Pistol Ribozyme (RNA-Puzzle 17)** (A) the native structure, (B) the model with the close fold to the native, detected in a small cluster, (C) the biggest cluster with the model that was returned as the final prediction.

An analogous analysis was performed the results of clustering of EvoClustRNA|SimRNAweb run for the TPP riboswitch. Models with a pairwise distance in terms of RMSDs lower than 9 Å are connected. Interestingly, the native structure (Fig. 4A, big dot) was mapped to a cluster of models of one of the homologs (Fig. 4, blue). The center of this cluster (Fig. 4B) achieved an RMSD (of helical, shared fragments) of 9 Å to the native structure. In this cluster, there were not models for the target sequence. Since SimRNAweb was not able to detect non-canonical interactions, most of the structures were in "open" conformation and clustered far from the native structure. The final prediction was (Fig. 4C) achieved an RMSD of 24.08 Å with respect to the native.

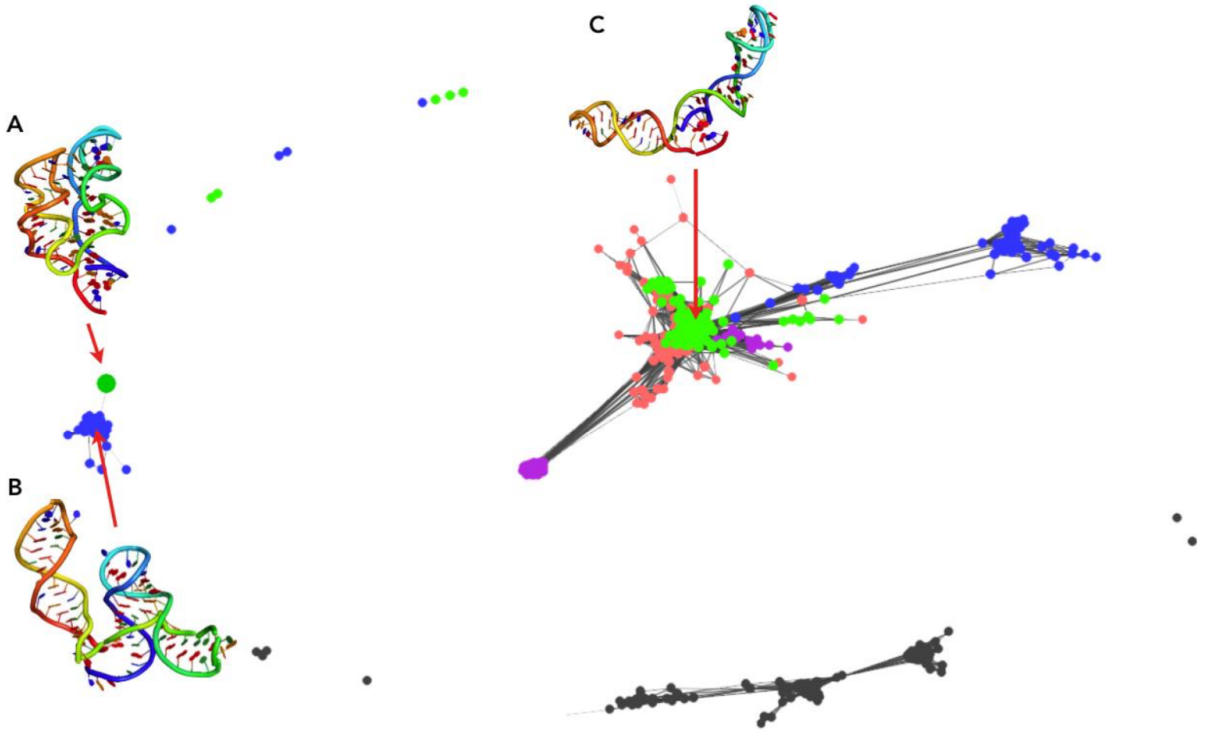

Figure 4: **Clustering visualized with Clans for TPP riboswitch** (A) the native structure, (B) the model with the close fold to the native (C) the biggest cluster with the model that was returned as the final prediction.

Learn more about Clanstix https://rna-tools.readthedocs.io/en/latest/tools.html#module-rna_tools.tools.clanstix.rna_clanstix

Figure 4: **Clustering visualized with Clans for TPP riboswitch** (A) the native structure, (B) the model with the close fold to the native (C) the biggest cluster with the model that was returned as the final prediction.

CHAPTER 2

Adv

2.1 RNA 3D structure prediction

For each sequence chosen for folding, secondary structure predictions were generated based on the MSA. Two methods were used in this study: SimRNA and Rosetta. For Rosetta, a total of 10,000 decoys were generated for the target sequence and each homologous sequence using the Rosetta FARFAR protocol. For SimRNA prediction, SimRNAweb server was used using the default parameters.

Both modeling steps can be performed in a semi-automated way with rna-tools (M.M. et al., unpublished, software available for download at https://github.com/mmagnus/rna-tools) as well as the pipeline of tools facilitating modeling with Rosetta (https://rna-tools.readthedocs.io/en/latest/tools.html#rosetta) and SimRNA/SimRNAweb (https://rna-tools.readthedocs.io/en/latest/tools.html#simrnaweb).

2.2 evoClustRNA

```
usage: evoClustRNA.py [-h] [-a RNA_ALIGNMENT_FN] [-o OUTPUT_DIR]
                      [-i INPUT_DIR] [-m MAPPING_FN] [-x MATRIX_FN] [--inf]
                      [-v] [-s] [-f]
```

2.2.1 Named Arguments

-a, --rna_alignment_fn rna alignemnt with the extra guidance line, e.g. test_data/rp14sub.stk

-o, --output_dir output folder where motifs and structures will be saved, e.g. test_out/rp14 (default: out -> out/structures and out/motifs will be created

Default: “out”

-i, --input_dir input folder with structures, .e.g. test_data

Default: “.”

-m, --mapping_fn	a file with mapping folders on the drive with sequence names in the alignment (<name in the alignment>:<folder name>), use multiple lines for multiple seqs
-x, --matrix_fn	output matrix with rmsds all-vs-all Default: ""
--inf	Use INFs instead of RMSD Default: False
-v, --verbose	be verbose Default: False
-s, --save	save motifs and structures to output_dir, this slows down the program Default: False
-f, --flat-dir	use flat directory structure, structures/<all pdbs here>, fetch pdbs based on leading part of names Default: False

When RNA models are loaded, models ending with 'template.pdb' are ignore.

evoClustRNA.**get_rna_models_from_dir**(*directory*, *residues*, *save*, *output_dir*, *flat_dir*)

@todo

This function goes folder by folder.

Ugly hack: it removes clust01-05X from the list.

Parameters

- **directory** –
- **residues** –
- **save** –
- **output_dir** –

Returns

Return type

evoClustRNA.**parse_num_list**(*s*)

http://stackoverflow.com/questions/6512280/accept-a-range-of-numbers-in-the-form-of-0-5-using-pythons-argparse

evoClustRNA.**sort_nicely**(*l*)

Sort the given list in the way that humans expect.

http://blog.codinghorror.com/sorting-for-humans-natural-sort-order/

2.3 evoClust_autoclustix.py

```
usage: evoClust_autoclustix.py [-h] [--half] [-v] matrix
```

2.3.1 Positional Arguments

matrix	A txt file with a similarity matrix with column headers, See test_data/matrix.txt for more . ! .txt is need to auto-removal system to work

2.3.2 Named Arguments

--half 50% in 3 the biggest clusters

Default: False

-v, --verbose Default: False

evoClust_autoclustix.py implements a simple interactive clustering. Technically, this script is a simple wrapper for evoClust_clustix.py.

```
usage: evoClust_clustix.py [-h] [-o OUTPUT] [-c CUT_OFF] [-v] matrix
```

2.3.3 Positional Arguments

matrix A txt file with a similarity matrix with column headers, See test_data/matrix.txt for more

2.3.4 Named Arguments

-o See test_data/output.txt for more, don't type extension of the file

-c Cut_off of RMSD for the formation of a cluster

Default: 5.0

-v, --verbose be verbose

Default: False

2.4 evoClust_get_models.py

evoClust_get_models.py

Uses find in curr directory to find needed file.

This script creates: - reps for top 5 clusters representative structures - resp_motifs for top 5 clusters representative motifs

Add cutoff the name of reps, e.g. reps_c2.5

The script has the second mode right now:

```
[mm] rosetta-5x$ evoClust_get_models.py -i structures/ ade_plus_ade_cleanup_mapping_
↪pkX_*.out -n adepk
evoClust_get_models.py
--------------------------------------------------------------------------------
['adepk_min.out.10.pdb', 'adepk_min.out.5.pdb', '', 'adepk_min.out.1.pdb', '']
1_adepk_min.out.10.pdb
2_adepk_min.out.5.pdb
3_
4_adepk_min.out.1.pdb
5_
= structures == out/structures/<files>===================
cp -v structures//adepk_min.out.10.pdb reps_ns/c1_adepk_min.out.10.pdb
structures//adepk_min.out.10.pdb -> reps_ns/c1_adepk_min.out.10.pdb
```

(continues on next page)

(continued from previous page)

```
cp -v structures//adepk_min.out.5.pdb reps_ns/c2_adepk_min.out.5.pdb
structures//adepk_min.out.5.pdb -> reps_ns/c2_adepk_min.out.5.pdb
cp -v structures// reps_ns/c3_
cp: structures// is a directory (not copied).
cp -v structures//adepk_min.out.1.pdb reps_ns/c4_adepk_min.out.1.pdb
structures//adepk_min.out.1.pdb -> reps_ns/c4_adepk_min.out.1.pdb
cp -v structures// reps_ns/c5_
cp: structures// is a directory (not copied).

# evoClust_get_models.py -i structures/ ade_plus_ade_cleanup_mapping_pkX_*.out -n␣
↪adepk
```

first, the input is parsed to get borders of lines of clusters. These borders are used to select structures that come to a given cluster. For each cluster, there is a search if within it there is a structure that starts with a given name - defined with –NATIVE_SEQ_ONLY. If there is none, then to the reps list ‘’ is appended.

OLD: It reads *out* folder created by evoclustRNA.py in structure such as: - out/structures/<homologs>

```
usage: evoClust_get_models.py [-h] [-i INPUT_DIR] [-o OUTPUT_PREFIX] [-c] [-s]
                              [-u] [-n NATIVE_SEQ_ONLY]
                              clustix_results_fn
```

2.4.1 Positional Arguments

clustix_results_fn

2.4.2 Named Arguments

-i, --input_dir input folder with structures, .e.g. test_data

Default: “out”

-o, --output_prefix output folder where motifs and structures will be saved, e.g. test_out/rp14

Default: “”

-c, --use-cutoff-for-names Default: False

-s, --skip_motifs Default: False

-u, --skip_structures Default: False

-n, --native-seq-only

2.5 Python Classes used in the scripts

2.5.1 RNAmodel

class RNAmodel.**RNAmodel**(*fpath*, *residues*, *save=False*, *output_dir=”*)

Example

```
>>> rna = RNAmodel("test_data/rp14/rp14_5ddp_bound_clean_ligand.pdb",␣
↪[1], False, None)
>>> rna.get_report()
"File:  rp14_5ddp_bound_clean_ligand.pdb  # of atoms: 1 \nresi:  1 ␣
↪atom:  <Atom C3'> \n"
```

Parameters

- **fpath** – file path, string
- **residues** – list of residues to use (and since we take only 1 atom, C3', this equals to number of atoms.
- **save** – boolean, save to output_dir or not
- **output_dir** – string, if save, save segments to this folder

get_report()
Str a short report about rna model

get_rmsd_to(*other_rnamodel*, *output=''*, *dont_move=False*)
Calc rmsd P-atom based rmsd to other rna model

save(*output_dir*, *verbose=True*)
Save structures and motifs

2.5.2 RNAalignment

RNAalignment

Example:

```
    # STOCKHOLM 1.0

AACY023581040                --CGUUGGACU------AAA--------AGUCGGAAGUAAGC-----AAU-C-----
↪-GCUGAAGCAACGC---
    AJ630128                    AUCGUUCAUUCGCUAUUCGCA-AAUAGCGAACGCAA--AAG------CCG-A-
↪------CUGAAGGAACGGGAC
    target                      --CGUUGACCCAG----GAAA-----
↪CUGGGCGGAAGUAAGGCCCAUUGCACUCCGGGCCUGAAGCAACGCG--
    #=GC SS_cons                ::(((((,<<<<<<<..______..>>>>>>>,,,,,,,,<<<<....______.
↪....>>>>,,,,)))))::::
    x                           --xxxxxxxxx-----------------xxxxxxxx--xxx------------
↪------xxxxxxxxxx----
    #=GC RF                     AUCGUUCAuCucccc..uuuuu..ggggaGaCGGAAGUAGGca....auaaa.
↪....ugCCGAAGGAACGCguu
//
```

x line is used to pick resides to calculate RMSD.

x line could be renamed to EvoClust

class RNAalignment.**RNAalignment**(*fn*)
RNAalignemnt

get_range(*seqid*, *offset=0*, *verbose=True*)
Get a list of positions for selected residues based on the last line of the alignment!

If seqis not found in the alignment, raise an exception, like

```
Exception: Seq not found in the alignment: 'CP000879.1/21644622164546
```

Warning: EvoClust lines has to be -1 in the alignemnt.

CHAPTER 3

Installation

Get it the project:

```
git clone https://github.com/mmagnus/EvoClustRNA.git
```

or download https://github.com/mmagnus/EvoClustRNA/archive/master.zip and unzip the file.

Enter the folder `evoClustRNA` and to install, type::

```
pip install --user -r docs/requirements.txt
```

Check if all requirements have been installed correctly. If yes, you can go to Get Started! :-)

CHAPTER 4

Indices and tables

- genindex
- modindex
- search

Python Module Index

e

r

Index