Kunitz work flow

1. **PDB Search**

1.1. pfam: pf00014
1.2. resolution 2.0 at least
1.3. length of chain 50 and 70
1.4. wild type

2. **PDB report**

2.1. PDB ID
2.2. Chain length
2.3. eperimental method
2.4. resolution
2.5. entity ID
2.6. Chain ID
2.7. tabularResults.csv

3. **PDBefold**

3.1. select highest resolution from pdb report and undergo pairwise alignment with the whole database
3.2. take the pdb ids and chains from the file
3.3. reslist.dat

4. **merge PDBefold and PDB results**

4.1. take the ids and chains from both the files

4.2. command for pdb file preparation:

4.2.1. `cat tabularResults.csv |tail -n +2 | sed 's/"//g' | cut -d ',' -f 1,2 | sed 's/,/:/g' > pdbresult.id.chain`

- pdberesult.id.chain

4.3. command for pdbe file preparation

4.3.1. `cat reslist.dat | tail -n +6 | tr -s '\t' ' ' | cut -d ' ' -f 19 > pdberesult.id.chain`

- pdberesult.id.chain
  - move to upper case
    - `cat pdberesult.id.chain | tr '[:lower:]' '[:upper:]' > pdberesult.id.chain.mod`
      - pdberesult.id.chain.mod

4.4. common ids between two

4.4.1. `comm -12 <(sort pdbresult.id.chain) <(sort pdberesult.id.chain.mod) > common.pdb.pdbe.id.chain`

- common.pdb.pdbe.id.chain
  - remove redundancy

5. **common file is reduntant -> reduce it by clustering**

5.1. download pdb sequences database

5.2. print the ids in the right format

5.2.1. `cat common.pdb.pdbe.id.chain | awk -F ':' '{print tolower($1)"_"$2}' > common.pdb.pdbe.idlow.chain.pbdseqrescomp`

- common.pdb.pdbe.idlow.chain.pbdseqrescomp

5.3. use python script to extract fasta

5.3.1. sel_seq_dictionary.py

- use the command

- python ../sel_seq_dictionary.py common.pdb.pdbe.idlow.chain.pbdseqrescomp pdb_seqres.txt > common.pdb.pdbe.fasta
    - common.pdb.pdbe.fasta

5.4. clustering command with coverageand identity options

5.4.1. ./../blast-2.2.26/bin/blastclust -L 0.9 -S 90 -i common.pdb.pdbe.fasta -o common.pdb.pdbe.clust

- common.pdb.pdbe.clust

5.5. add resolution to each pdb id using python script

5.5.1. clustsort_res.py

- use command
    - python ../clustsort_res.py common.pdb.pdbe.clust pdbtabsep.table common.pdb.pdbe.clust.res

5.6. seed ids are chosen by highest resolution

5.6.1. command

- cat common.pdb.pdbe.clust.res | cut -d ' ' -f 1 | cut -d ':' -f 2 > hmm.seed.ids
    - hmm.seed.ids

6. **hmmbuild**

6.1. use the seed ids and do multiple structural alignment using PDBefold

6.1.1. fasta.seq

6.2. use the msa file to produce hmm

6.2.1. command

- hmmbuild hmm.kunitz fasta.seq
    - Screenshot from 2019-05-04 17-44-13.png
        - hmm.kunitz

7. after building the HMM, testing is mandatory

8. Positive Set

8.1. In uniprot enter the website and search for kunitz using Pfam 00014 and reviewed

8.1.1. make the file of seed ids is suitable for comparison

-
  ```
  cat hmm.seed.ids | awk -F '_' '{print toupper($1)}' > hmm.seed.ids.uniprot.filter
  ```
  - hmm.seed.ids.uniprot.filter

8.1.2. download tab format and filter the seed ids using special command

-
  ```
  cat uniprot-pf00014+reviewed%3Ayes.tab | grep -v -f hmm.seed.ids.uniprot.filter | cut -f 1 > positive.filteredfromseeds.ids
  ```
  - positive.filteredfromseeds.ids

8.2. download uniprot swissprot database

8.3. get the sequence of the filtered positive

8.3.1. command

-
  ```
  python ../sel_seq_dictionary-foreuniprot.py positive.filteredfromseeds.ids uniprot_sprot.fasta > positive.fasta
  ```
  - positive.fasta

8.4. undergo hmmsearch

8.4.1. command

-
  ```
  hmmsearch --tblout positive.search --noali --max hmm.kunitz positive.fasta
  ```
  - positive.search

8.5. normalize

8.5.1. command

- cat positive.search | tail -n +4 | tr -s ' ' ' ' | cut -d ' ' -f 1,8 | sed 's/|/ /g' | cut -d ' ' -f 2,4 | head -n 343 | awk -F ' ' '{i=$2/343; print $0" "i" "0}' > positive.search.set
    - positive.search.set

8.6. Do we need to do a blastall after eliminating the possibility by removing the Uniprot entry that correspond to uniprot IDs?

8.6.1. the answer is no need thanks to roberto

9. **negative set**

9.1. In uniprot enter the website and search for everything else! and reviewed

9.1.1. uniprot-NOT+pf00014+reviewed%3Ayes+length%3A %5B45+TO+_%5D.list

9.2. take randomly 500 sequences

9.2.1. command to take ids

- cat uniprot-NOT+pf00014+reviewed%3Ayes+length%3A %5B45+TO+_%5D.list | sort -R | head -n 500 > negative.ids
    - command to get sequences
        - python ../sel_seq_dictionary-foreuniprot.py negative.ids uniprot_sprot.fasta > negative.fasta

9.3. do the hmmsearch

9.3.1. command

- hmmsearch --tblout negative.search --noali -E 1e+50 --domE 1e+50 --max hmm.kunitz negative.fasta
    - negative.search

9.4. normalize

9.4.1. command

- cat negative.search | tail -n +4 | head -n 201 | tr -s ' ' ' ' | cut -d ' ' -f 1,8 | sed 's/|/ /g' | cut -d ' ' -f 2,4 |awk -F ' ' '{i=$2/500; print $0" "i" "1}' > negative.search.set
    - negative.search.set

9.5. correcting small set

9.5.1. extract the no hit seqs

- define the hits
    - command
        - cat negative.search.set | cut -d ' ' -f 1 > negative.hits.ids
- then compare with original
    - command
        - comm -23 <(sort negative.ids) <(sort negative.hits.ids) > negative.nohits.ids
- add with E value '1'
    - python didn't work
    - bash is the answer
        - cat negative.nohits.ids | awk '{print $0" 5e+02 1 1"}' >> negative.search.set.mod
- the whole set

10. confusion matrix

10.1. this matrix is to test and evaluate the hmm

10.1.1. confusionM.py

10.2. run according to threshold

10.2.1. combine the positive and negative set files

- whole.set

10.2.2. command

- python confusionM.py whole.set 1e-05

11. testing the hmm with bigger data size

11.1. collect the rest of negative ids

11.2. extract fasta

11.3. do hmmsearch

11.3.1. command

- hmmsearch --tblout total.test.search --noali -E 1e+500 --domE 1e+500 --max hmm.kunitz total.test.fasta

11.4. normalize

11.4.1. command

- cat total.test.search | tail -n +4 | tr -s ' ' ' ' | cut -d ' ' -f 1,8 | sed 's/|/ /g' | cut -d ' ' -f 2,4 |awk -F ' ' '{i=$2/547488; print $0" "i}' > total.search.set

11.5. test according to previous threshold

11.5.1. improve threshold

- best is 1e-05

11.6. test and add positive

11.7. correcting the large set

11.7.1. same as small set

12. **ROC curve**

13. **awk NF remove empty lines**