Evan Edelstein

Summer 2019

Lab Notebook

Overview:

Four scripts were implemented to collect organize and sort conservation score data for proteins. Conservation scores were collecting from Predus and Consurf. Predus showed a global F-score of 0.340 while Consurf had a global F-score of 0.102.

Key Terms and Methods:

Residue:

Residue is the unit of protein used to measure accessibility, energy and binding. Specifically, interface residues are the parts of the protein that take part in binding and protein function. Our job is to identify which residues are interface residues.

Conservation Score:

Conservation score is a measurement of relative residue importance, as determined by the likelihood that a specific residue will stay unchanged as the protein evolves over time. Three main factors are taken into account to determine a proteins evolution: alignment/structure, homology and phylogeny. Predus and Consurf differ in the weight they give to each factor, which explains the divergence in score. Predus relies heavily on structure and comparing residue position with homologous proteins, a sort of “horizontal evolution, while Consurf relies on its machine learning mechanics to simulate phylogenetic trees, a “vertical evolution”.

Predus uses homolog proteins and structural information to examine evolutionary differences horizontally

Consurf uses machine learning phylogenetic trees to determine

conservation vertically.

Scripts:

\*\*\*all scripts used can be find under the E\_Edelstein branch in the MAIN folder at https://github.com/eved1018/Raji\_Summer2019.git

Script 1: PDB\_parser.R

R script that takes in proteins PDB IDs and returns a directory with PDB files by chain.

Script 2: Predus\_Score\_unix.sh

Shell/Unix script that takes in PDB files from Predus, appended with a conservation score, and returns a file with only relevant data, residue number and conservations score. The files are sorted by conservation score, then compared with the annotated list of interface residues. Currently it counts how many annotated residues are listed and takes that number residues from the Predus file and combines them into aa single file with the annotated residues. The script then counts the duplicate lines in the file, or true positives. It then computes the F score, (true positives / number of annotated residues), and the false positives, (annotated – true positive). A global and average F score is the computed.

This is not a great method because residues sharing the same conservation score but lower on the list (because of a higher residue id, a factor unrelated to interface) are not included in the comparison. This leads to a deflated F score, as residues that Predus predicted as interface are not compared to the annotated list. Another factor effecting the efficacy of the method is the inability to automatically get the Predus files. The thirty test proteins were inputted mainly.

Input:

Predus files

Parsed for Protein Id and conservation score

Sorts by conservation score and counts until amount of lines is equal to annotated file

Compares annotated file with Predus file returning True positives, F score, and False positives. Outputs table with data for each protein

Script 3: Consurf.sh

Same as script 1 but for Consurf. Conservation score is found using Consurf database, by calling the script. F score, true positives, and false positives are computed. Major problems with the script include the small number of proteins in the Consurf database, 10 protein scores were not calculated by the script.

Script 4: Predus\_ROC.sh

To get around the problem of cutting the list of interface residues from Predus to short, ROC curves were constructed. A threshold level was used to calculate the cutoff conservation score instead of using the same number as annotated residues. The script iterated over an increasing threshold value, calculating the true and false positive rate at each threshold. A global false and true positive rate for each protein were used as axis for the ROC curve. The output of the script is an CSV file, compatible with Microsoft Excel. In the future, a direct graphing script will be used to output the ROC curve directly.

Script 5: fscore\_predus\_and\_ispred.sh

This script takes the residue ID, and F score from the result files from Predus and Ispred, to compare the two. Also takes in vorffip no clue why it isn’t in the name.

Results: F scores:

protein Predus Ispred Vorffip

1ACB.E .384 .307 .153

1ACB.I .636 .818 .272

1AHW.A .666 .150 .300

1AK4.A .222 .333 .111

1AVX.A .684 .473 .052

1AY7.A .222 .555 .111

1AY7.B .444 .333 .111

1BGX.H .470 .225 .064

1DE4.E .714 .380 .285

1DFJ.E 0 .111 .055

1DFJ.I .611 .388 0

1DQJ.A .250 .222 .055

1E4K.C .666 .166 .166

1E6J.H .200 .142 0

1EXB.E .333 .333 .333

1F34.A .043 .434 .130

protein Predus Ispred Vorffip

1FC2.D .200 0 .200

1FFW.A 0 .250 0

1FFW.B .400 .400 0

1FLE.E .181 0 0

1GHQ.A 0 0 .200

1GXD.A 0 .095 0

1GXD.C .444 .277 .111

1IRA.X .521 .565 .043

1JIW.I .266 .400 .066

1JIW.P .250 .350 0

1JPS.H .125 .187 0

1KAC.A 0 0 0

1KAC.B .500 .250 .083

1KLU.D .090 .181 0

G.F.S .340 .290 .090

A.F.S 0.307 0.260 0.090

Future Goals:

1. Currently the scripts are all “localized” in that they directly depend on a specific directory setup, my computer. Generalizing the scripts to work on any system is a top priority.
2. Automating the web input of Predus is essential to efficiency, having to manually input each protein takes time.
3. A series-scaled ranking system that finds the average rank per property and then scales by “average” F score (f score/sum of F scores).
4. Cleaning up my project folder and renaming scripts and files to increase readability. As well as combining multiple scripts into one script with multiple parts to increase efficiency.
5. Learning Perl