Processing of Biological Data

Prof. Dr. Volkhard Helms
Tutor: Andreas Denger
Winter Semester 2021

Saarland University
Chair for Computational Biology

Exercise Sheet 4

Due: 07.01.2022 10:15 am

Submission

- You are advised to work in groups of two people.
- Submit your solutions as a single PDF attachment to andreas.denger@bioinformatik.uni-saarland.de. Hand in all source code via mail. Late submissions will not be considered.
- Do not forget to write your names/matriculation numbers on your submission.
- You are free to use any programming language to solve the problems.

CLI software and Gene annotation enrichment analysis

Exercise 4.1: Clustering with cd-hit (50 points)

In this exercise, we will perform clustering on a protein dataset. If the similarity between the sequences of two proteins is higher than a given threshold value, the two proteins will be part of the same cluster.
The clustering will be performed with the command line tool *cd-hit*. This program runs on Linux and MacOS. If you are using Windows, you can install an Ubuntu command line through the Windows Subsystem for Linux[1], or install Ubuntu in VirtualBox[2].

(a) Clone the cd-hit git repository from `https://github.com/weizhongli/cdhit`, and follow the instructions in the Readme file to compile the program. On Ubuntu, it could look something like this:

```
# make sure that c++ compiler and make are installed
sudo apt install g++ make
# clone repo
git clone https://github.com/weizhongli/cdhit
# enter directory
cd cdhit
# compile without zlib dependency
make zlib=no
```

(b) Call the cd-hit executable, and list the parameters with **./cd-hit −−help**. The repository also contains a PDF file with a users guide. What are the recommended word sizes for the thresholds 40%, 50%, 60% and 70%, when clustering protein sequences?

(c) Write a script, in a language of your choice, that calls cd-hit in order to cluster *ex1.fasta* at four different sequence similarity thresholds (40%, 50%, 60% and 70%), and creates the output files *ex1_cluster40.fasta*, ..., *ex1_cluster70.fasta*, accordingly.

(d) Cd-hit also creates a *.clstr file for every clustered fasta file it creates. Write a program that reads a .clstr file and creates a bar plot, where the x-axis corresponds to the cluster size, and the y-axis to the number of clusters with that size. Create a plot for all four files, and include them in your submission.

[1]https://docs.microsoft.com/en-us/windows/wsl/install
[2]https://ubuntu.com/tutorials/how-to-run-ubuntu-desktop-on-a-virtual-machine-using-virtualbox

Exercise 4.2: Gene Ontology enrichment analysis (50 points)

In this exercise, you will perform a basic GO term enrichment analysis for a set of proteins from *Arabidopsis Thaliana*. The goal is to find a set of protein annotations that are overrepresented in this subset of proteins, when compared to the background set of all proteins in *A. Thaliana*.

(a) Read the files *ex2.tsv* and *goa_arabidopsis.tsv* into appropriate data structures. The former contains the set of proteins, the latter is a table of GO annotations for all proteins in *A. Thaliana*.

(b) Now it is time to prepare the table of GO terms for the analysis.

   (1) The second column, called *qualifier*, describes the relation between the protein in the column *Uniprot*, and the GO term in the column *go_term*. According to the official documentation[3], it is possible for a GO term to be explicitly *not* associated with a certain protein. Remove any rows where this is the case.

   (2) Another column contains the *evidence code*. The guide to evidence codes[4] on the website explains these abbreviations. For this analysis, we want the annotations to be based on experimental evidence, and/or reviewed by a human. Remove any rows that were *inferred from electronic annotation*.

   (3) Finally, the GO annotations are divided into three parts: Biological **P**rocess, Cellular **C**omponent, and Molecular **F**unction. These are described on the website[5], and can be found in the *aspect* column in the file. Split your table into three tables, one for each aspect.

(c) Next, we will calculate the enrichment scores for each GO term. For each of the three tables from the previous part, perform the following steps:

   (1) Calculate the frequency $F_{\pi,background}$ of each GO term π as the number of proteins in the table with that GO term divided by the total number of proteins in the table.

   (2) Calculate the frequency $F_{\pi,proteinset}$ of each GO term π as the number of proteins with that GO term in the protein set, divided by the number of proteins in the intersection of the protein set and the table.

   (3) Calculate the enrichment factor for each GO term as $E_\pi = log2\left(\frac{F_{\pi,proteinset}}{F_{\pi,background}}\right)$

(d) Interpretation of results

   (1) Try to make sense of the enrichment score. What does a score of 5.0, -5.0 and 0.0 tell us about a GO term, respectively?

   (2) Retrieve the descriptions for some of the GO terms with the highest and lowest enrichment score in each table. You can find a description for each GO term by searching for it on QuickGO[6] or AmiGO[7].

   (3) Write a few sentences about your findings. What can you say about the set of proteins with regards to their molecular functions, biological processes and cellular components, based on this enrichment analysis?

   (4) The enrichment score that is calculated here is not a statistical test. Which statistical test could you use to estimate whether the enrichment of a GO term is statistically significant?

[3]http://geneontology.org/docs/go-annotations/
[4]http://geneontology.org/docs/guide-go-evidence-codes/
[5]http://geneontology.org/docs/ontology-documentation/
[6]https://www.ebi.ac.uk/QuickGO/
[7]http://amigo.geneontology.org/amigo