MBioMed Bioinformatics: Assignments

Assignment #1: Analyzing Gene Expression

The goal of this assignment will to become better acquainted with R and to familiarize yourselves with its use for simple data handling and analysis. First of all you will need to:

1. Download and install R on your computers. I also recommend installing R-Studio, which provides a more user-friendly environment.
2. Once you do this, please take a look at this R-for-beginners-bundle (in Greek) that contains most of the things you need to know on installing R, data input and output and handling the basic data types and plots.
3. Next, you are asked to try to analyze a real gene expression dataset that you can download from here. The data we will study come from an expression experiment using DNA microarrays. mRNA expression has been measured for 18703 mouse genes in a wild-type strain (Wt-control) against 5 different conditions (A, B, C, D, E). Each experiment was performed in 4 biological replicates. The result is an array 18703x24 (four repetitions for 5 + 1 conditions). The file in the link above contains the **normalized expression values**. Our goals in this analysis are:

- Calculate gene expression fold change for each gene. In order to do this you will need to obtain the mean value for *each gene* and for *each condition* (Wt, A, B, C, D, E) and then calculate the difference $logFC(gene_{i,X/WT}) = mean(gene_{i,X}) - mean(gene_{i,Wt})$ for all conditions $X = A, B, C, D, E$ and for every gene i. The mean will be calculated over the 4 replicates.
- Calculate the p-value of a t-test for the same comparisons. This time you will use the 4 replicates instead of the mean. That is we need a p-value referring to $t.test(gene_{i,X,k=1-4}, gene_{i,Wt,k=1-4})$.
- Once you have done this, you will have successfully transformed the count data to relative expression data. You are now in position to apply the commands we discussed in part C3 (above) in order to extract differentially expressed genes (DEGs) for each of the conditions A, B, C, D and E. Use the following arbitrary thresholds (|logFC|>=1 & p.value<=0.05). Report back the numbers of DEGs for each condition.

You are free to work in groups of 2-3 people but please drop a comment on our Slack channel naming the members of your group.

Assignment #2: Sequence Similarity and Clustering

Introduction

The goal of this assignment will be to combine sequence similarity analysis with the clustering approaches we discussed last time.

Histone H4 is one of the most well-preserved proteins among eukaryotic organisms. In this assignment, you will have to assess **the degree of conservation** through **Needleman-Wunsch pair-wise alignment** in specific protein sequences derived from a broad group of eukaryotes.

The file you will find here contains the H4 protein sequences from the following organisms: (Homo sapiens: HUMAN, Mus musculus: MOUSE, Drosophila melanogaster: DROME, Arabidopsis thaliana: ARATH, Bos taurus: BOVIN, Caenorhabditis elegans: CAEEL : CHICK, Rattus norvegicus: RAT, Saccharomyces cerevisiae: YEAST, Xenopus laevis: XENLA)

Stage 1

You are asked to:

1. Align all sequences with each other in pairs using the BLOSUM50 replacement table.
2. Indicate the alignment scores on a square table of NXN dimensions

Auxiliary information: It is recommended that you use the the R::seqinr and BioConductor::Biostrings libraries. The installation of the former is done directly by R

```
install.packages("seqinr")
```

while for Biostrings you have to go through Bioconductor:

```
BiocManager::install("Biostrings")
```

The reading of the sequences can be done with the $read.fasta()$ function of seqinr. Alignment will be performed with the Biostrings $pairwiseAlignment()$ function using the substitutionMatrix = "BLOSUM50" parameter.

Stage 2

In this second stage you will use the data from the previous query. The purpose here is to estimate the evolutionary distances between the sequences based on pairwise alignment. You are required to:

1. Convert the similarity table from Stage 1 to a distance table. You can do this by taking the maximum similarity max(S) of the whole table and applying the formula $D_i = (1 - S_i)/Max(S)$ to all S_i values. 2.Use the distance D table as input to create a UPGMA tree. You can use the $hclust()$ function of the basic R.
2. Return the tree and comment if it has an image that reflects the complexity relationships between the organisms

Assignment #3: Motifs

For this assignment you are asked to perform a motif building and search analysis using a set of defined motif instances and custom-made R functions.

Your goal is to create a PWM with a set of instances of the GATA transcription factor, transform it into PSSM and then search it against a whole genome sequence.

1. You may start by downloading a set of GATA binding sequences that you will find here.
2. Once this is done, you can download the target sequence against which you will search the PWM here.

3. In order to build the PWM you need to read the GATA sequences using this custom function and then create the matrix with the PWM function. In order to use these functions you have to upload them to R using:

```
source("nameoffunction.R")
```

This means, that you may use $readfastafile()$ as:

```
source("readfastafile.R")
gataseqs<-readfastafile("gata.fa")
```

and then obtain the PWM with:

```
source("seqMotif.R")
gataPWM<-seqMotif(gataseqs, drawmotif=F)
```

4. The transformation of the PWM to PSSM requires an additional random sequence collection, which you can find here. You can then create the PSSM by taking the log2(ratio) of the two PWMs (one from the GATA sequences over the one from the random sequence).
5. In the next step you will use a PSSMSearch Function to search for the motif in the longer target sequence. Assuming you have created the PSSM into the table called *pssm*, the list of commands below should work:

```
source("pssmSearch.R")
targetset<-readfastafile("test1.fa")
pssmSearch(pssm, targetseq, threshold=X)
```

The parameter X in the threshold tells the search function to return a different number of successful hits for the search. Threshold can take values from 0 to 1, with 1 returning the instances of the binding sites that have score equal to the maximum (100%) of the PSSM. A value of 0.8 will return values with scores 80% or more of the maximum and thus a greater number of hits.

6. As a last step you are asked to do the following. Perform three different searches with threshold=0.5, 0.8 and 1.0 and obtain the results in a vector. Then use this vector of positions to extract the sequences from the target sequence (test1.fa). Then store these sequences in a table (you can use "write.table") and paste them to the WebLogo WebService to obtain three different sequence logos. Compare the three and discuss their differences.