BIOINFORMATIC RESOURCES PROJECT

The project, provided as an R script, aims at using various concepts learnt throughout the Bioinformatics Resources course in order to obtain different results starting from a data set.

Each of the project tasks has one or more results which, depending on the steps performed, are provided as plots, saved in files or variables.

The data set selected in order to test the data was the Lung Cancer (LungCancer.R) data set. Despite that, the script should work given any data set, if both the Rdata and TSV files used as input are correct.

The code is structured in order to separate each task, in ascending order, using comments.

Task n.1: Load the Rdata in RStudio

To perform this task, the Rdata file can easily be loaded in RStudio using the function load(). I have used an absolute path but, depending on the context, a relative path could be given as the input of the function if a working directory is set, using setwd().

Task n.2: Update raw_count_df and r_anno_df extracting only protein coding genes

In this step, using biomaRt and the Ensembl database, the script downloads a list of all the protein coding genes having the ID in the given Rdata file. Then it removes all the non-protein coding genes from the list, updating raw_count_df and r_anno_df with these values.

Since the number of selected genes is quite high, a query selecting only genes on the 21st chromosome was used to test the code in order to spend less time computing the results.

The query is now commented and the query selecting all the genes is provided in the code. Using a different query might break some functions, for example in the 4th task.

Task n.3: Perform differential expression analysis

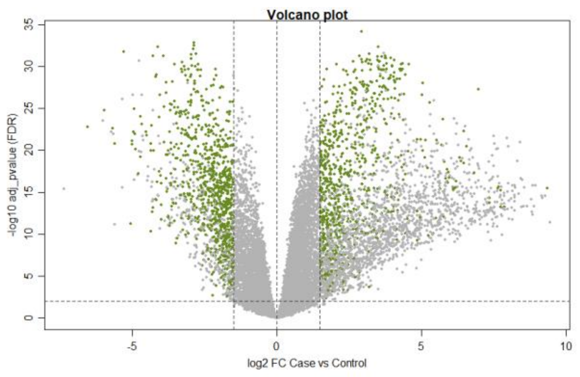
Using the library edgeR, differential expression analysis is performed.

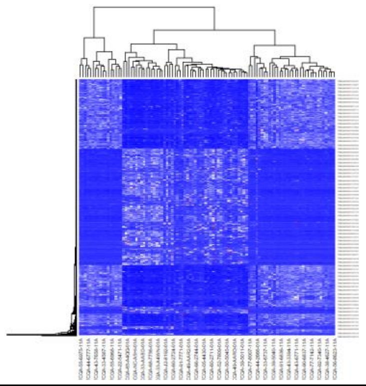
At first, the data set is filtered according to a count threshold and a minimum number of replicates (3b).

Then, using the function seen during the course, the analysis is performed setting variables for a p-value cutoff of 0.01 (pval_thrs), a log fold change ratio >1.5 for up-regulated genes and < (-1.5) for down-regulated genes (fc_thrs) and a log2 CPM >1 (CPM_thrs).

The list of differentially expressed genes is saved in DEGs, which is then used to create a Volcano plot and an Annotated Heatmap. The Volcano plot indicates with vertical lines the fold change thresholds, while the horizontal line shows the p value threshold.

The resulting plots are shown below.



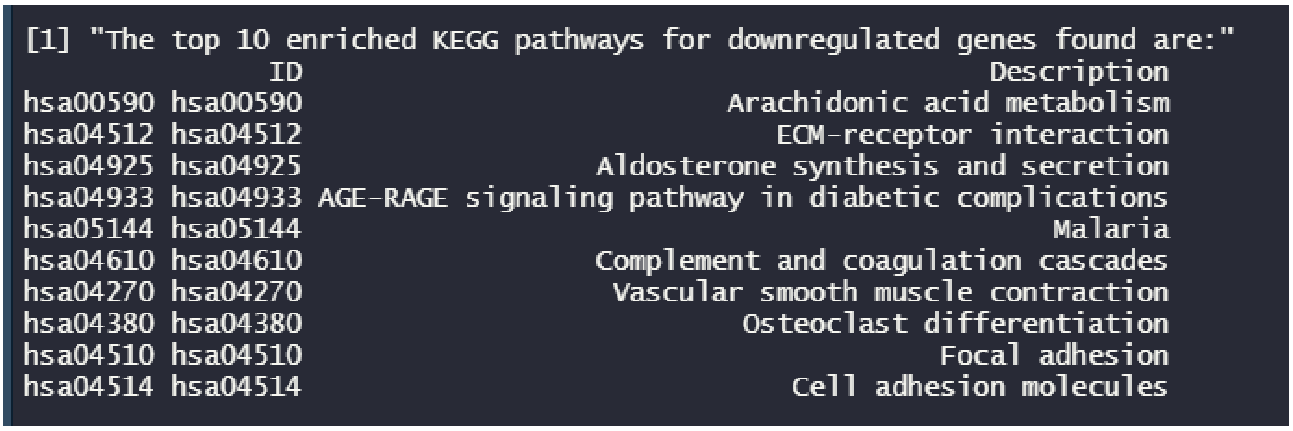


Task n.4: Perform gene set enrichment analysis

The Ensembl database is used in order to retrieve additional information for the list of DEGs, such as the values entrezgene_id and external_gene_name.

Two different datasets are then created, selecting from DEGs only genes that are respectively upregulated (upDEGs) and downregulated (downDEGs).

Enrichment Analysis is then performed on both upregulated and downregulated genes, using both GO (BP and MF) and KEGG analysis. The top 10 enriched GO terms are printed, for example:

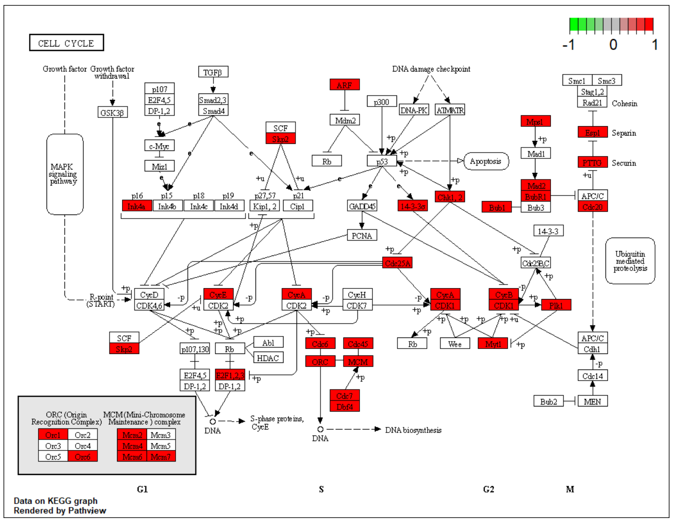


```
[1] "The top 10 enriched KEGG pathways for downregulated genes found are:"
                ID                                          Description
hsa00590 hsa00590                           Arachidonic acid metabolism
hsa04512 hsa04512                              ECM-receptor interaction
hsa04925 hsa04925                      Aldosterone synthesis and secretion
hsa04933 hsa04933 AGE-RAGE signaling pathway in diabetic complications
hsa05144 hsa05144                                                Malaria
hsa04610 hsa04610                       Complement and coagulation cascades
hsa04270 hsa04270                        Vascular smooth muscle contraction
hsa04380 hsa04380                             Osteoclast differentiation
hsa04510 hsa04510                                         Focal adhesion
hsa04514 hsa04514                                Cell adhesion molecules
```

Task n.5: Visualize one pathway you find enriched using the up-regulated gene list

Using the function pathview() and the data frame upDEGs, containing the list of upregulated genes, an image explaining one enriched pathway is stored in the current working directory. The chosen pathway is the one having contained in the 1st index of the data frame. Changing the index of the input data frame changes the corresponding pathway.

The resulting figure is shown below, representing the graph that describes one pathway enriched according to the starting gene list.



Task n.6: Identify TFs having enriched scores in the promoters of all up-regulated genes

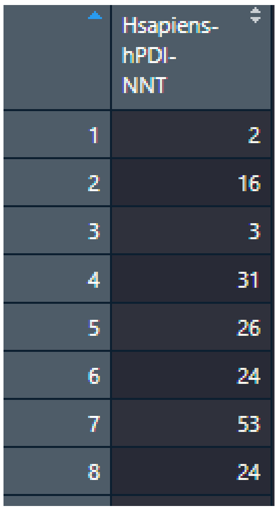
Using again the Ensembl database, all the sequences related to upregulated genes are retrieved, with a window of 500 nucleotides upstream for each gene. These sequences are then converted to a DNAString object and then the scores are computed, using in this case the affinity score.

Tasks n.7/8: Select one among the top enriched TFs and calculate the empirical distribution of scores. Identify which genes have a region in their promoter with binding scores above the computed thresholds.

From the TFs obtained in the previous step, a target is chosen in order to perform the calculations.

The PWMs for the selected TF are downloaded from MotifDB and then the thresholds and the scores are computed using a threshold cutoff at 99.5%.

The result is stored in the scores data frame. Since every row in the data frame for this specific data set has a value of 0, no specific genes having a region in the selected promoter. Changing the threshold to 95% gives instead more results, for example:



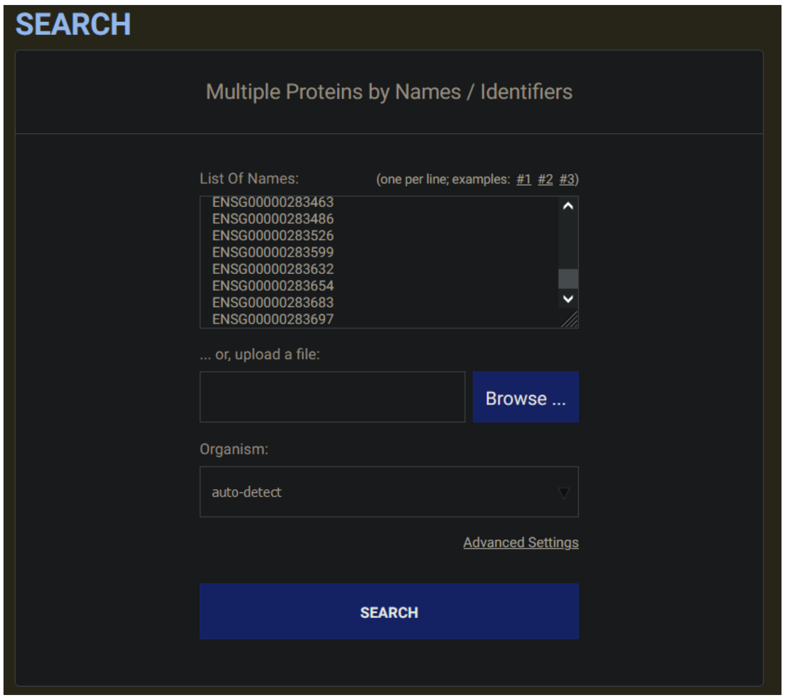
	Hsapiens-hPDI-NNT
1	2
2	16
3	3
4	31
5	26
6	24
7	53
8	24

Task n.9: Use STRING database to find PPI interactions

Using the website https://string-db.org/ and the list of differentially expressed genes found in the task n.3, a list of protein-protein interactions can be downloaded.

The list of differentially expressed genes was saved in a txt file to make it easier to copy that on the website.

Since the website has a maximum of 2000 nodes in a network, only upregulated genes were used.


SEARCH

Multiple Proteins by Names / Identifiers

List Of Names: (one per line; examples: #1 #2 #3)

```
ENSG00000283463
ENSG00000283486
ENSG00000283526
ENSG00000283599
ENSG00000283632
ENSG00000283654
ENSG00000283683
ENSG00000283697
```

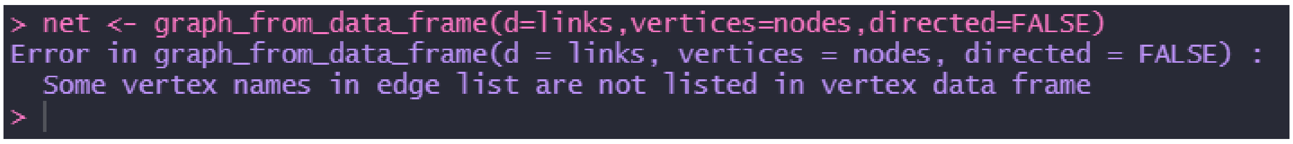
... or, upload a file:

Organism: auto-detect

The result is exported in a TSV file and then loaded in R.

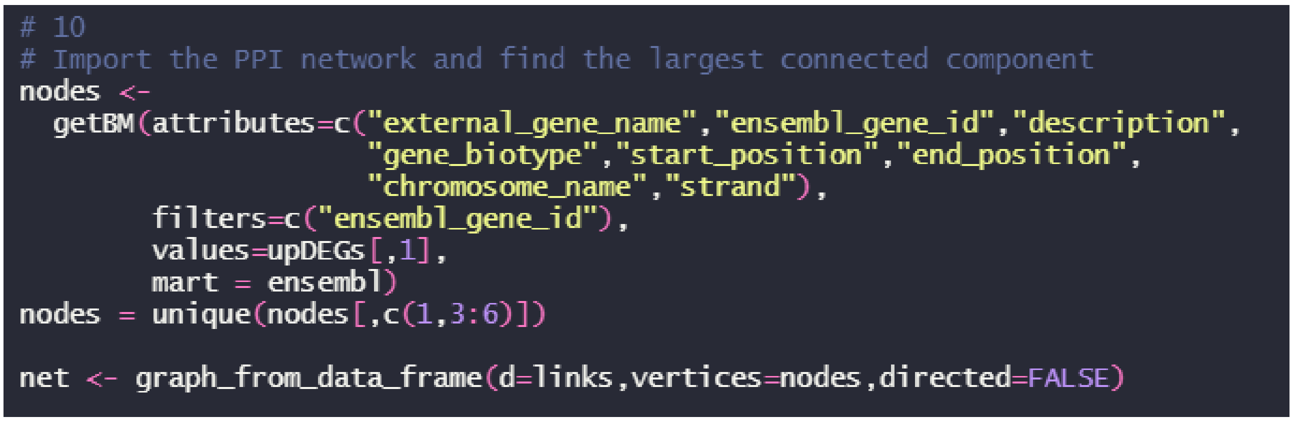
Task n.10: Import the network and determine which is the largest connected component

I was not able to complete this task specifically, as Rstudio kept giving me an error which I was not able to fix. This happened both with the code I have tried to write and the one used during the class.



```
> net <- graph_from_data_frame(d=links,vertices=nodes,directed=FALSE)
Error in graph_from_data_frame(d = links, vertices = nodes, directed = FALSE) :
  Some vertex names in edge list are not listed in vertex data frame
>
```

Here is the code which should, given the list of connections between the edges, create a network.



```
# 10
# Import the PPI network and find the largest connected component
nodes <-
  getBM(attributes=c("external_gene_name","ensembl_gene_id","description",
                     "gene_biotype","start_position","end_position",
                     "chromosome_name","strand"),
        filters=c("ensembl_gene_id"),
        values=upDEGs[,1],
        mart = ensembl)
nodes = unique(nodes[,c(1,3:6)])

net <- graph_from_data_frame(d=links,vertices=nodes,directed=FALSE)
```