Project 3. Functional Genomics Analysis for Gene Expression Arrays

Table of Contents

## The Outline for Project 3

In this project, we will perform classical functional genomic analysis on a gene expression microarray dataset. We will first conduct hierarchical clustering on samples, and then determine the proper feature selection method that can improve the prediction of the tissue labeling. Next, we will perform clustering on genes, during which we will determine the number of gene modules using the silhouette analysis, followed by interpreting the gene clusters by gene set enrichment analysis in DAVID.

## 1. Construct SummarizedExperiment Object using Expression Matrix and Experimental Design

The data set containing the expression levels obtained from a set of microarray experiments is stored in a matrix; while the rows of the matrix are genes / probes, and the columns of the matrix are samples. The expression matrix is saved in a tabular file named expression.csv under the project 3 directory. Please use the function read.csv () to load the CSV file into R. Remember to set the parameter row.names =" probeID " so that the first column becomes the row name of the imported data.frame object.

Then, we need to read in another important file called tissue.csv. This file is a table with rows corresponding to the tissue information of each sample. Please read the file into R with read.csv().

Next, store the expression matrix and the tissue information into a single summarizedExperiment object named by SE. The expression matrix should be entered in the assays= argument, and the tissue information should be entered in the colData= argument.

Also, please notice that the colData slot in the summarizedExperiment object must be a DataFrame object (not data.frame), so make sure to convert the tissue into DataFrame with the function DataFrame().

The reason we want to construct a summarizedExperiment object is to tidy things up. Compared to simultaneously managing multiple variables (expression assays, column designs, row’s metadata, and row’s GRanges), it is much easier to code with only one variable that includes all of the data in a genomic experiment.

## === Hint code, fill the "\_\_\_" ============= ##  
#tissue <- read.csv(\_\_\_)  
#e <- read.csv(\_\_\_, row.names = "probeID")  
#library(SummarizedExperiment)  
#SE <- SummarizedExperiment(assays = \_\_\_, colData = DataFrame(\_\_\_)) #Represent the data into a summarizedExperiment  
## ===== Enter your code below =============== ##  
tissue <- read.csv("/home/yuxuan/BIO214/Project3/tissue.csv")  
e <- read.csv("/home/yuxuan/BIO214/Project3/expression.csv", row.names = "probeID")  
library(SummarizedExperiment)  
SE <- SummarizedExperiment(assays = e, colData = DataFrame(tissue)) #Represent the data into a summarizedExperiment

Examine the summarizedExperiment object as the following:

SE

## class: SummarizedExperiment   
## dim: 22215 189   
## metadata(0):  
## assays(1): ''  
## rownames(22215): 1007\_s\_at 1053\_at ... 91920\_at 91952\_at  
## rowData names(0):  
## colnames(189): GSM11805.CEL.gz GSM11814.CEL.gz ... GSM307640.CEL.gz  
## GSM307641.CEL.gz  
## colData names(1): Tissue

table(SE$Tissue)

##   
## cerebellum colon endometrium hippocampus kidney liver   
## 38 34 15 31 39 26   
## placenta   
## 6

* SAQ1: How many genes or probes are included in this set of the experiments? How many samples are there? How many unique tissues are there? What are the contents of rownames and colnames of the SummarizedExperiment object?
* Answer: 22215 genes or probes are included in this set of the experiment 189 samples 7 unique tissues rownames genes or probes colnames are the sample IDs

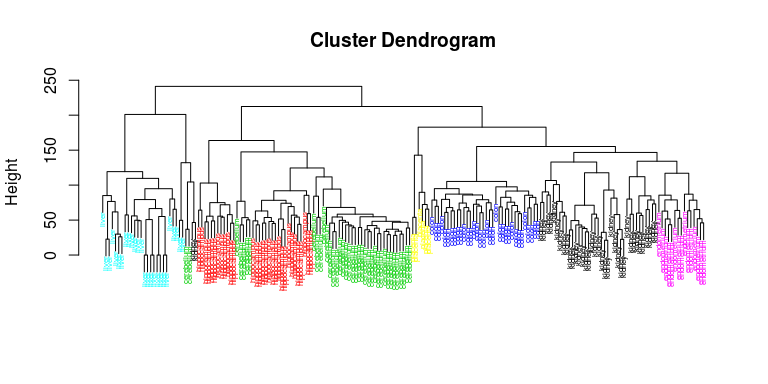
## 2. Hierarchical Clustering by Samples

Next, we will perform hierarchical clustering through the columns of the expression array. As introduced in lecture 8, to perform hierarchical clustering, we must calculate the distance metric between the objects in advance.

The distance metric can be calculated in R using the function dist(). Please note that by default, dist() is used to calculate the distances between rows of the matrix input. Therefore, if you want to cluster by columns, you need to to use the function t() to transform the row and columns of the matrix.

Then, you can then use the hclust() function to build a hierarchical clustering dendrogram with distance metrics. In this problem, please perform a hierarchical clustering on Euclidean distance between the columns of the expression assay, and store the result hclust object in the variable hc.

## === Hint code, fill the "\_\_\_" ============= ##  
#d <- dist( t(\_\_\_) ) #Use assay() to extract the expression matrix in SE  
#hc <- hclust(\_\_\_)  
## ===== Enter your code below =============== ##  
d <- dist( t(assay(SE)) ) #Use assay() to extract the expression matrix in SE  
hc <- hclust(d)  
## ===== Your code is finished =============== ##  
library(rafalib)  
myplclust(hc, labels=SE$Tissue, lab.col=as.fumeric(as.character(SE$Tissue)),cex=0.5)



After drawing the dendrogram, you should able to see a tree structure representing the arrangement of sample clusters. We could observe that the division of the branches has a significant correlation with the tissue label.

Next, we will retrieve the “partition” based on the hierachical clustering dendrogram. To achieve this, we need to apply the function cutree(). Set the number of clusters (also called cardinality or K) to the number of tissues.

Then, tabulate the two factors of cluster partition and the true tissue labeling. Display the contingency table. Use chisq.test() to evaluate the contingency table.

Then, use DescTools::MutInf() to calculate the empirical mutual information between the 2 factors.

[Mutual information](https://en.wikipedia.org/wiki/Mutual_information) is a measure of the dependence between two random variables. If the 2 variables are independent, their mutual information will be 0. MI is defined as the information gain when modeling the dependency between the 2 random variables (compared with the independence modeling).

## === Hint code, fill the "\_\_\_" ============= ##  
#hclusters <- cutree(\_\_\_, k=\_\_)  
#tb <- table(true = \_\_\_, cluster = \_\_\_)  
#tb  
#chisq.test(tb)  
#DescTools::MutInf(\_\_\_, \_\_\_)  
## ===== Enter your code below =============== ##  
hclusters <- cutree(hc, k=7)  
tb <- table(true = tissue$Tissue, cluster = hclusters)  
tb

## cluster  
## true 1 2 3 4 5 6 7  
## cerebellum 0 0 36 0 0 2 0  
## colon 0 0 0 34 0 0 0  
## endometrium 15 0 0 0 0 0 0  
## hippocampus 0 12 19 0 0 0 0  
## kidney 37 0 0 0 0 2 0  
## liver 0 0 0 0 24 2 0  
## placenta 0 0 0 0 0 0 6

chisq.test(tb)

##   
## Pearson's Chi-squared test  
##   
## data: tb  
## X-squared = 787.33, df = 36, p-value < 2.2e-16

DescTools::MutInf(tissue$Tissue, hclusters)

## [1] 2.090514

* SAQ2: What are the calculated Chi-square statistics and mutual information? Are the cluster partitions significantly associate with the tissue labels? Is the prediction of the tissues perfect? Please explain your reasons.
* Answer: The calculated Chi-square statistics 787.33, p value < 0.05, which rejects the null hypothesis that there is no significant difference between the cluster partitions and the tissue labels mutal information: 2.09 The cluster partitions significantly associate with the tissue labels. No, the prediction is not perfect since the method we used is based on complete data set by default, which might lead some bias or deviation.

Change the clustering method to “ward.D2” by setting the method parameter of the hclust() function. Repeat the above analysis to see if the mutual information between the prediction and the label has increased.

## === Hint code, fill the "\_\_\_" ============= ##  
#hc <- hclust(\_\_\_, method = \_\_\_)  
#hclusters <- cutree(\_\_\_, k=\_\_)  
#tb <- table(true = \_\_\_, cluster = \_\_\_)  
#tb  
#DescTools::MutInf(\_\_\_, \_\_\_)  
## ===== Enter your code below =============== ##  
hc <- hclust(d, method = "ward.D2")  
hclusters <- cutree(hc, k=7)  
tb <- table(true = tissue$Tissue, cluster = hclusters)  
tb

## cluster  
## true 1 2 3 4 5 6 7  
## cerebellum 0 5 31 0 0 2 0  
## colon 0 0 0 34 0 0 0  
## endometrium 15 0 0 0 0 0 0  
## hippocampus 0 31 0 0 0 0 0  
## kidney 37 0 0 0 0 2 0  
## liver 0 0 0 0 24 2 0  
## placenta 0 0 0 0 0 0 6

chisq.test(tb)

##   
## Pearson's Chi-squared test  
##   
## data: tb  
## X-squared = 877.35, df = 36, p-value < 2.2e-16

DescTools::MutInf(tissue$Tissue, hclusters)

## [1] 2.250404

The mutal information between the prediction and the samples are increased to 2.25.

## 3. Feature Selection and Heat Map

Then, we will use a simple but most widely applied feature extraction technique: keeping only the genes that have the highest variance. This time, we will keep the 85 rows with the highest (row) variances.

You could use the function rowVar() defined in the geneFilter package to calculate the variance of the gene. Save the index of the selected rows into a variable named idx. Using the ward.D2 method, conduct hierarchical clustering of columns with the rows subsetted by idx. Please draw a dendrogram and then evaluate the mutual information again.

Then, run the code prepared below to draw the heat map.

## === Hint code, fill the "\_\_\_" ============= ##  
#library(genefilter)  
#rv <- rowVars(\_\_\_)  
#d <- dist(\_\_\_)  
#idx <- order(-\_\_\_)[\_\_\_]  
#hc <- hclust(\_\_\_, method = \_\_\_)  
#hclusters <- cutree(\_\_\_, k=\_\_)  
#tb <- table(true = \_\_\_, cluster = \_\_\_)  
#tb  
#DescTools::MutInf(\_\_\_, \_\_\_)  
## ===== Enter your code below =============== ##  
library(genefilter)  
rv <- rowVars(assay(SE))  
idx <- order(-rv)[1:85]  
d <- dist(t(e[idx,]))   
hc <- hclust(d, method = "ward.D2")   
hclusters <- cutree(hc, k=7)  
tb <- table(true = tissue$Tissue, cluster = hclusters)  
tb

## cluster  
## true 1 2 3 4 5 6 7  
## cerebellum 0 0 5 33 0 0 0  
## colon 0 0 0 0 34 0 0  
## endometrium 0 0 0 0 0 0 15  
## hippocampus 0 0 31 0 0 0 0  
## kidney 22 17 0 0 0 0 0  
## liver 0 0 0 0 0 26 0  
## placenta 0 0 0 0 0 0 6

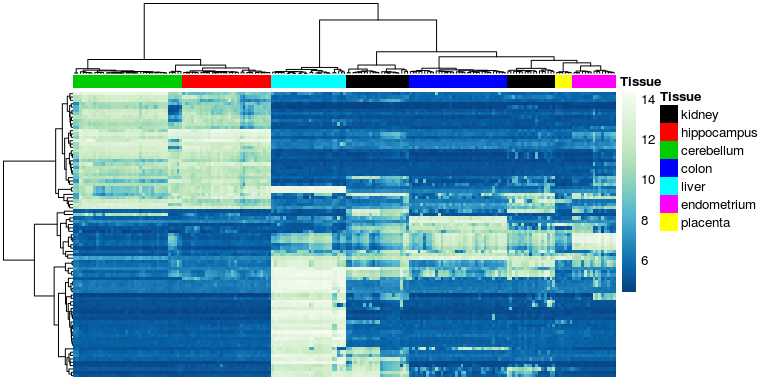
chisq.test(tb)

##   
## Pearson's Chi-squared test  
##   
## data: tb  
## X-squared = 897.34, df = 36, p-value < 2.2e-16

DescTools::MutInf(tissue$Tissue, hclusters)

## [1] 2.443282

## ===== Your code is finished =============== ##  
library(pheatmap)  
library(RColorBrewer)  
  
annotation\_col = data.frame(  
 Tissue = SE$Tissue  
 )  
rownames(annotation\_col) = colnames(SE)  
  
hmcol <- colorRampPalette(rev(brewer.pal(n = 9, name = "GnBu")))(100)  
cols <- palette(brewer.pal(8, "Dark2"))[as.fumeric(as.character(unique(SE$Tissue)))]  
names(cols) <- unique(SE$Tissue)  
  
pheatmap(assay(SE)[idx,],  
 color = hmcol,  
 show\_rownames=FALSE,  
 show\_colnames=FALSE,  
 annotation\_col=annotation\_col,  
 annotation\_colors = list(Tissue = cols),  
 scale = "none",  
 clustering\_method="ward.D2",  
 clustering\_distance\_cols="euclidean")



* SAQ3: Based on the mutual information and heat map, can we use only 85 genes to better classify tissues? Try to explain why we can obtain the observed performance using less than 0.5% of the total genes?
* Answer:

Yes, the mutal information changed from 2.25 to 2.44, and the MI doesn’t deviate a lot with the original one. The reason is that we first order the variance of gene, and the most similar gene are clustered. Meanwhile, the hclust() function conducts the same procedure, they first categorize the most similar gene, then iteratively identify the most similar gene. The essence of selecting less than 0.5% of the gene is similar to the hcluster of the whole complete data

* SAQ4: If we search exhaustively for all possible gene subsets, will we overestimate the above performance? Please provide your reasons. If the answer is yes, then what strategies can be used to solve this problem?
* Answer: We will not overestimate the performance. Since the results obtained from previous question, there is no significant deviation between the exhaustive search and 0.5% of the genes

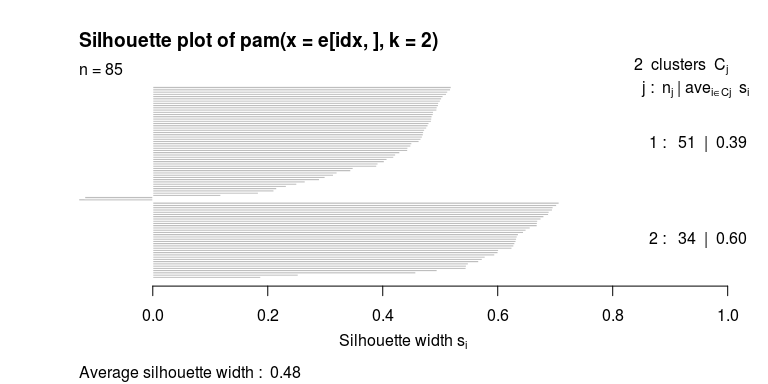
## 4. Determine the Number of Gene Clusters

The quality of the cluster partitions can also be determined by the average [Silhouette values](https://en.wikipedia.org/wiki/Silhouette_(clustering)) of each cluster.

The silhouette values quantifies how similar the object is to its own cluster (cohesion) when compared to other clusters (separation). The range of the contour is -1 to +1, where a high value indicates that the object matches well with its own cluster, and poorly matches with neighboring clusters.

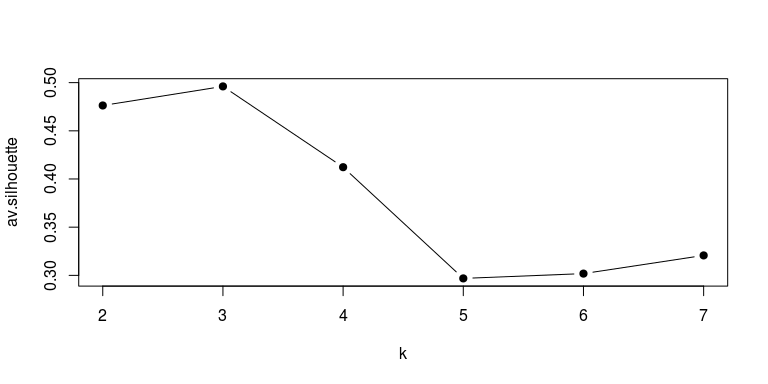
In the following steps, we need to calculate the Silhouette values of gene clustering. We will use the pam() function defined in the cluster package to perform this analysis. The function will run K-medoids clustering on the rows of the input matrix. Firstly, Set k = 2 so that we will have 2 gene clusterst. Please use the same 85 genes selected in the previous question. Save the output of pam() in the variable named pamclu.

set.seed(101)  
## === Hint code, fill the "\_\_\_" ============= ##  
#library(cluster)  
#pamclu = pam(\_\_\_, \_\_\_) #Set k = 2, make sure to subset with idx first!  
## ===== Enter your code below =============== ##  
library(cluster)  
pamclu = pam(e[idx,], k=2) #Set k = 2, make sure to subset with idx first!  
## ===== Your code is finished =============== ##  
plot(silhouette(pamclu), main=NULL)



Then, we will try to calculate the average silhouette values using different ks. Fill in the middle of the following code chunk block to calculate silhouette statistics for each k. Remember, we need a FOR loop to perform this process.

## === Hint code, fill the "\_\_\_" ============= ##  
#Ks = sapply(2:7, function(i) summary(silhouette(pam(\_\_\_, \_\_\_)))$avg.width)  
## ===== Enter your code below =============== ##  
Ks = sapply(2:7, function(i) summary(silhouette(pam(e[idx,],i)))$avg.width)  
## ===== Your code is finished =============== ##  
plot(2:7,Ks,xlab="k",ylab="av.silhouette",type="b",pch=19)



* SAQ5: When k = 2, is each gene well justified in its own cluster? In the ks we tested, what is the optimal number of gene modules? What is the most unlikely number? Please explain your reasons.
* Answer: K = 2, most gene is justified in its own cluster, except for two genes. In generael, the average silhouette width are 0.48, so well justified is a little bit inappropriate. Personally, I would say k = 2 and each gene is justified in its own cluster.  
  K = 3 is the optimal number of gene modules, since the average silhouette values is highest, which indicates that the object matches well with its own cluster.

K = 5 is the most unlikely number, since it has the lowest Silhouette value.

## 5. Functional Characterization of the Gene Modules.

Run hierachical clustering on the 85 genes using method ward.D2, cut the dendrogram using the optimized k obtained from the previous step. Store the output of cutree() in the variable named genecluster.

## === Hint code, fill the "\_\_\_" ============= ##  
#d <- dist(\_\_\_) #make sure to subset with idx first!  
#hc <- hclust(\_\_\_)  
#genecluster <- cutree(\_\_\_)  
## ===== Enter your code below =============== ##  
d <- dist((e[idx,])) #make sure to subset with idx first!  
hc <- hclust(d, method="ward.D2")  
genecluster <- cutree(hc,k=3)  
## ===== Your code is finished =============== ##  
sapply(unique(genecluster), function(i) writeLines(names(genecluster)[genecluster == i], paste0("List\_",i,".txt")) )

## [[1]]  
## NULL  
##   
## [[2]]  
## NULL  
##   
## [[3]]  
## NULL

writeLines(names(genecluster), "Background.txt")

After running the last 3 lines of code, you will get 3 sets of probe ids, which are saved under the project directory. Next, go to the [DAVID](https://david.ncifcrf.gov/) website.

Do the folowing steps:

1. Click Start Analysis >> Upload >> Copy & Paste genes in Background.txt >> select Background >> submit
2. Upload >> Copy & Paste genes in List\_i.txt >> select Gene List >> Submit list
3. click List >> select List\_i >> Use >> click Background >> select Background\_1 >> Use
4. Clear all >> Pathways >> KEGG\_PATHWAY
5. Functional annotation chart >> functional annotation chart
6. Repeat step 3, 4, and 5 for all the List\_is

You can find the description for the functional enrichment results at [here](https://david.ncifcrf.gov/helps/functional_annotation.html#summary)

* SAQ6: Report the top 5 KEGG terms enriched in each cluster(if any). Then, explain what the columns “P-value” and “Benjamini” mean? How do we determine whether each term is statistically significant or not? Combining the heat map and tissue information, please try to provide some biological explanations for the enriched terms.
* Answer: list1: 1 term, Complement and coagulation cascades list2: 0 term list3: 5 terme, protein digestion and absorption, amoebiasis, ECM-receptor interaction, focal adhesion and PI3K-Akt signaling pathway The column“p-value” means a modified Fisher Exact P-Value which ranges from 0 to 1 for gene-enrichment analysis and the smaller p-value reflects the more enriched.(p-value < 0.05 can be considered statistically significant) The column “Benjamini” means Benjamini-Hochberg Procedure, a powerful tool that decreases the false discovery rate. Usually P-Value is equal or smaller than 0.05 to be considered strongly enriched in the annotation categories

Based on the heat map and tissue information, we could obtain that we could divide the genes into three clusters.

## Session Info

sessionInfo()

## R version 3.6.3 (2020-02-29)  
## Platform: x86\_64-pc-linux-gnu (64-bit)  
## Running under: Ubuntu 16.04.6 LTS  
##   
## Matrix products: default  
## BLAS: /usr/lib/libblas/libblas.so.3.6.0  
## LAPACK: /usr/lib/lapack/liblapack.so.3.6.0  
##   
## locale:  
## [1] LC\_CTYPE=en\_US.UTF-8 LC\_NUMERIC=C   
## [3] LC\_TIME=en\_US.UTF-8 LC\_COLLATE=en\_US.UTF-8   
## [5] LC\_MONETARY=en\_US.UTF-8 LC\_MESSAGES=en\_US.UTF-8   
## [7] LC\_PAPER=en\_US.UTF-8 LC\_NAME=C   
## [9] LC\_ADDRESS=C LC\_TELEPHONE=C   
## [11] LC\_MEASUREMENT=en\_US.UTF-8 LC\_IDENTIFICATION=C   
##   
## attached base packages:  
## [1] parallel stats4 stats graphics grDevices utils datasets   
## [8] methods base   
##   
## other attached packages:  
## [1] cluster\_2.1.0 RColorBrewer\_1.1-2   
## [3] pheatmap\_1.0.12 genefilter\_1.68.0   
## [5] rafalib\_1.0.0 SummarizedExperiment\_1.16.1  
## [7] DelayedArray\_0.12.2 BiocParallel\_1.20.1   
## [9] matrixStats\_0.55.0 Biobase\_2.46.0   
## [11] GenomicRanges\_1.38.0 GenomeInfoDb\_1.22.0   
## [13] IRanges\_2.20.2 S4Vectors\_0.24.3   
## [15] BiocGenerics\_0.32.0   
##   
## loaded via a namespace (and not attached):  
## [1] xfun\_0.12 splines\_3.6.3 lattice\_0.20-41   
## [4] colorspace\_1.4-1 vctrs\_0.2.2 expm\_0.999-4   
## [7] htmltools\_0.4.0 yaml\_2.2.1 blob\_1.2.1   
## [10] XML\_3.99-0.3 survival\_3.1-8 rlang\_0.4.4   
## [13] DBI\_1.1.0 bit64\_0.9-7 GenomeInfoDbData\_1.2.2  
## [16] lifecycle\_0.2.0 stringr\_1.4.0 zlibbioc\_1.32.0   
## [19] munsell\_0.5.0 gtable\_0.3.0 mvtnorm\_1.0-12   
## [22] evaluate\_0.14 memoise\_1.1.0 knitr\_1.28   
## [25] AnnotationDbi\_1.48.0 Rcpp\_1.0.3 xtable\_1.8-4   
## [28] scales\_1.1.0 annotate\_1.64.0 XVector\_0.26.0   
## [31] farver\_2.0.3 bit\_1.1-15.1 digest\_0.6.25   
## [34] stringi\_1.4.5 grid\_3.6.3 tools\_3.6.3   
## [37] bitops\_1.0-6 magrittr\_1.5 DescTools\_0.99.34   
## [40] RCurl\_1.98-1.2 RSQLite\_2.2.0 MASS\_7.3-51.5   
## [43] Matrix\_1.2-18 rmarkdown\_2.1 R6\_2.4.1   
## [46] boot\_1.3-24 compiler\_3.6.3