# Lab 4 - Gr. 14 - Bioinformatics (732A93)

*Julius Kittler (julki092), Stefano Toffol (steto820), Saewon Jun (saeju204), Maximilian Pfundstein (maxpf364)*

## Assignment 1

**Task:**

- Run all the R code and reproduce the graphics (https://www.bioconductor.org/help/course-materials/2015/Uruguay2015/day3-gene.expression.html)
- Go carefully through the R code and explain in your words what each step does.

**Data import**

First, we import the data corresponding to the GEO accession number GSE20986 using the `getGEOSuppFiles()` function. We obtain a .tar file ("GSE20986_RAW.tar"). After extracting its contents, we obtain 12 .CEL.gz files. Consequently, these 12 files are unzipped and the results are put into the `data` directory. The resulting 12 files are .CEL files (cell intensity files).

These files contain microarray data created by Affymetrix DNA microarray image analysis software. Recall that each set of microarray data contains light intensity values, where high intensity corresponds to expressed genes and small intensity to genes that were not expressed.

All the data comes from humans (Homo sapiens), from four different tissues: iris, retina, choroid and human umbilical vein. There are three .CEL files per tissue.

References:

- https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE20986
- https://www.ncbi.nlm.nih.gov/geo/geo2r/?acc=GSE20986
- https://fileinfo.com/extension/cel
- https://www.affymetrix.com/support/developer/powertools/changelog/gcos-agcc/cel.html
- https://bioconductor.org/help/course-materials/2009/SeattleApr09/AffyAtoZ/AffymetrixAtoZSlides.pdf

**Creating phenodata**

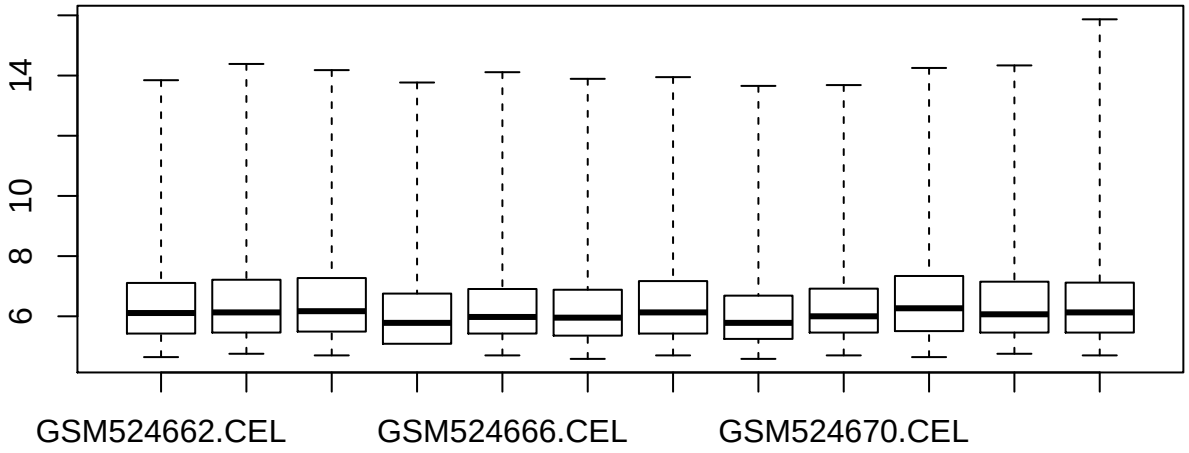
Before we are able to read in the data from all 12 files, we create a list with threee objects: The `Name` (for use in the resulting R object), the `FileName` (for finding the corresponding .CEL file) and `Targets` which seems to be some required meta data like the number of files. This list is then written as a .txt file. Finally, the function `read.affy()` from the `simpleaffy` package is called. to actually create the data to be used in R. This function takes phenodata.txt as input to be able to find the .CEL files. The resulting object `celfiles` is of class `AffyBatch` and contains the microarray data.

**Simple boxplot**

Now, we create a visualization of 12 boxplots, one for every .CEL file, using `celfiles` as input. For this, we use the `boxplot` function from the `BiocGenerics` package. Unfortunately, the x-axis labels are not readable. Therefore, we don't know which boxplots belong to which .CEL file. Therefore, we create an improved boxplot visualization in the next step.

Note: Although light intensity values do have a unit, it does not really matter for the interpretation. All we can do with the values is make relative comparisons within one experiment. Higher values correspond to expressed genes. It is not possible whatsoever, to make comparisons across experiments because the scale can differ. Furthermore, the scale is usually transformed (e.g. with logs), so that absolute interpretations make even less sense.

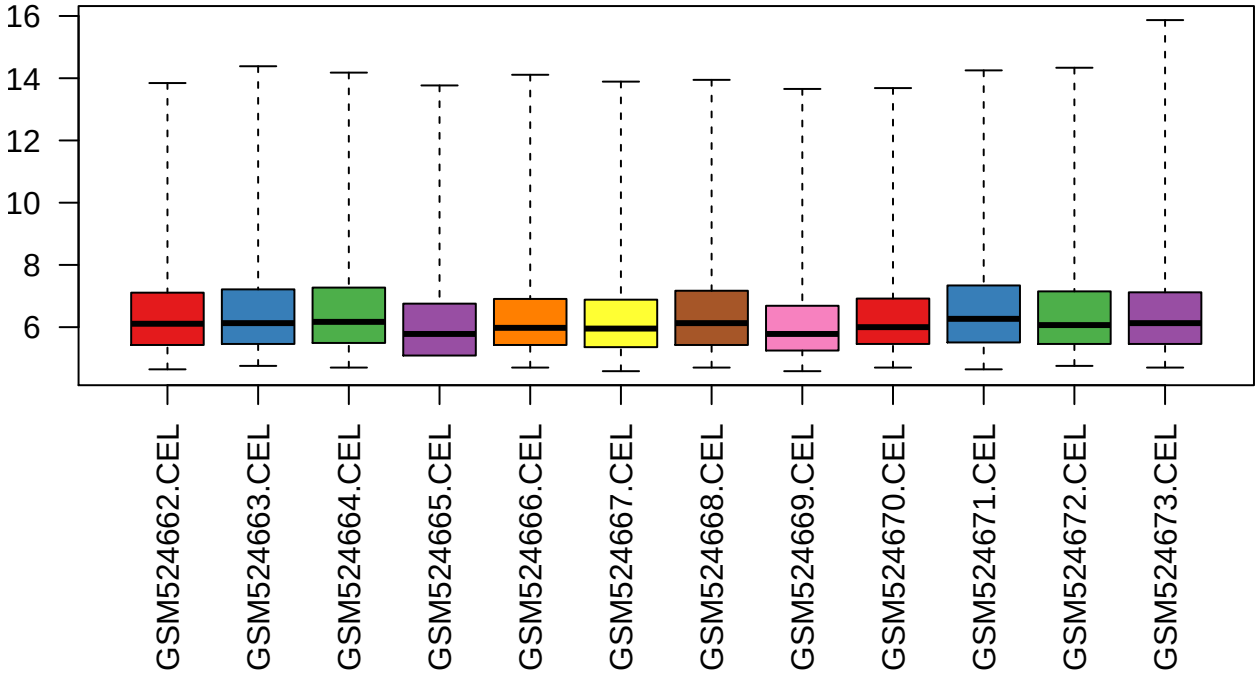
Interpretation: As we can see, the majority of the light intensity values are between 5 and 7. However, there are also some very large light intensity values which correspond to strongly expressed genes. The intensity values in the inter-quartile range (the rectangle boxes in the plots) may correspond to genes that are only expressed to a small extent or even not at all.




**Improved boxplot**

We now create an improved box plot, in which the x-axis labels are readable.

Note: The colors do not have a special meaning. They do not e.g. correspond to a specific tissue. Instead, there are 8 color values which are used from left to right. After all 8 values have been used, the 1st color is used, then the 2nd color etc.




**Cluster analysis (not normalized, tissue as labels)**

We have extracted the expression matrix (1354896 rows, 12 columns) from the `celfiles` object. Each row corresponds to a light intensity that tells us to what extend a gene is expressed (?). There are 12 columns for cell intensity file.

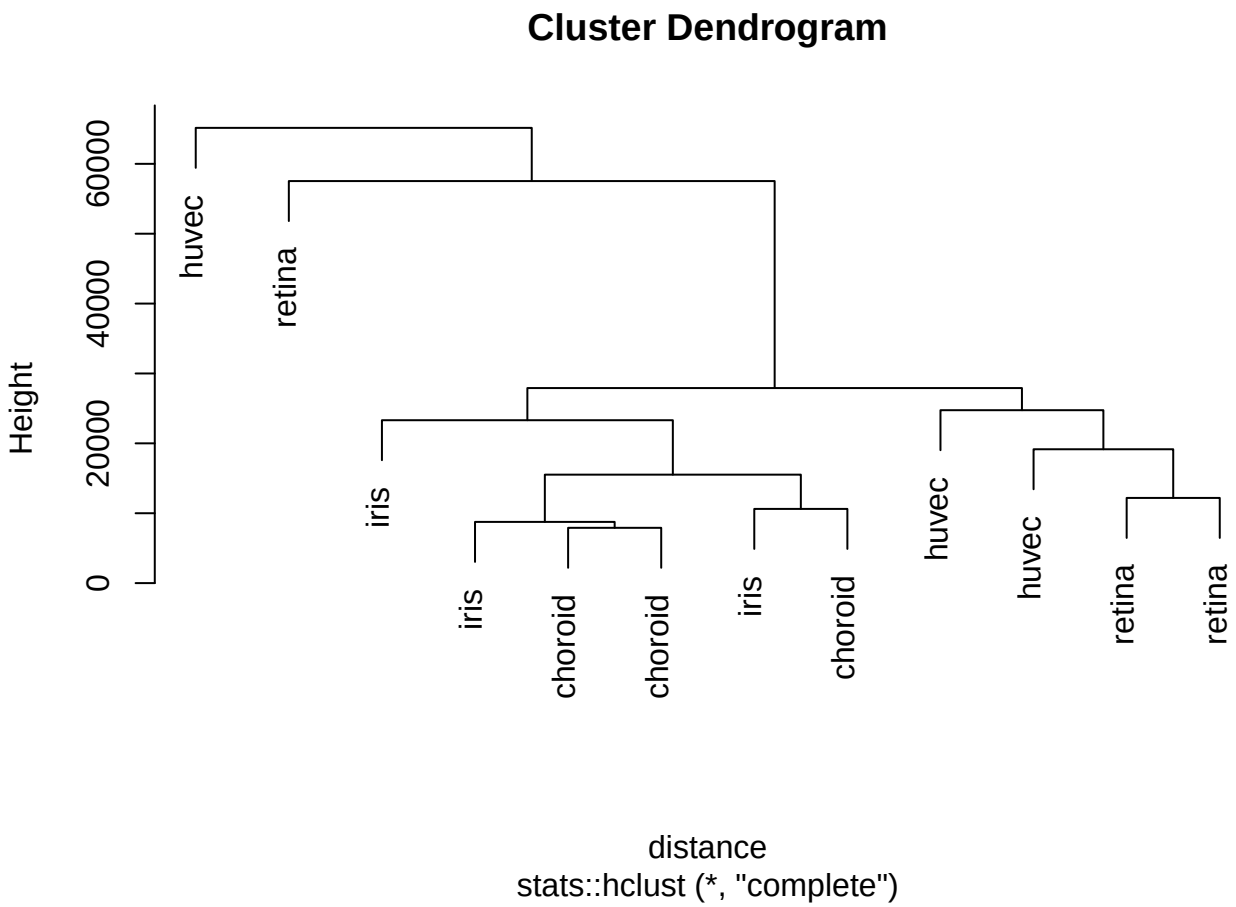
Now, we compute a distance matrix for these 12 columns, using the `Chebyshev distance` as a metric. The `Chebyshev distance` gives us the maximum difference between any pair of points in two vectors.

Subsequently, we conduct a hierarchical cluster analysis on this distance matrix and plot the results in a cluster dendogram. Note that there are 12 leaf nodes in the plotted dendogram, each node corresponding to one .CEL file. The labels tell us which tissue the .CEL file belonged to.

Note that HCA is done by a stepwise procedure. At each step, the two columns with the smallest dissimilarity are merged. We can therefore e.g. say that the bottom right leaf nodes (retina and retina) had a quite small dissimilarity. Also e.g. the 5th and 6th leaf nodes from the left (choroid and choroid) had a quite small dissimilarity. After all, we can conclude similarities between .CEL files of the same tissue are recognized. However, it is also clear that similarities between different tissues are recognized (such as iris and choroid).

References:


- https://bioconductor.org/help/course-materials/2009/SeattleApr09/AffyAtoZ/AffymetrixAtoZSlides.pdf
- https://stats.stackexchange.com/questions/209606/what-is-maximum-and-its-computation-in-the-function-dist-stats-in-
- https://en.wikipedia.org/wiki/Chebyshev_distance
- https://stats.stackexchange.com/questions/82326/how-to-interpret-the-dendrogram-of-a-hierarchical-cluster-analysis
- http://www.econ.upf.edu/~michael/stanford/maeb7.pdf

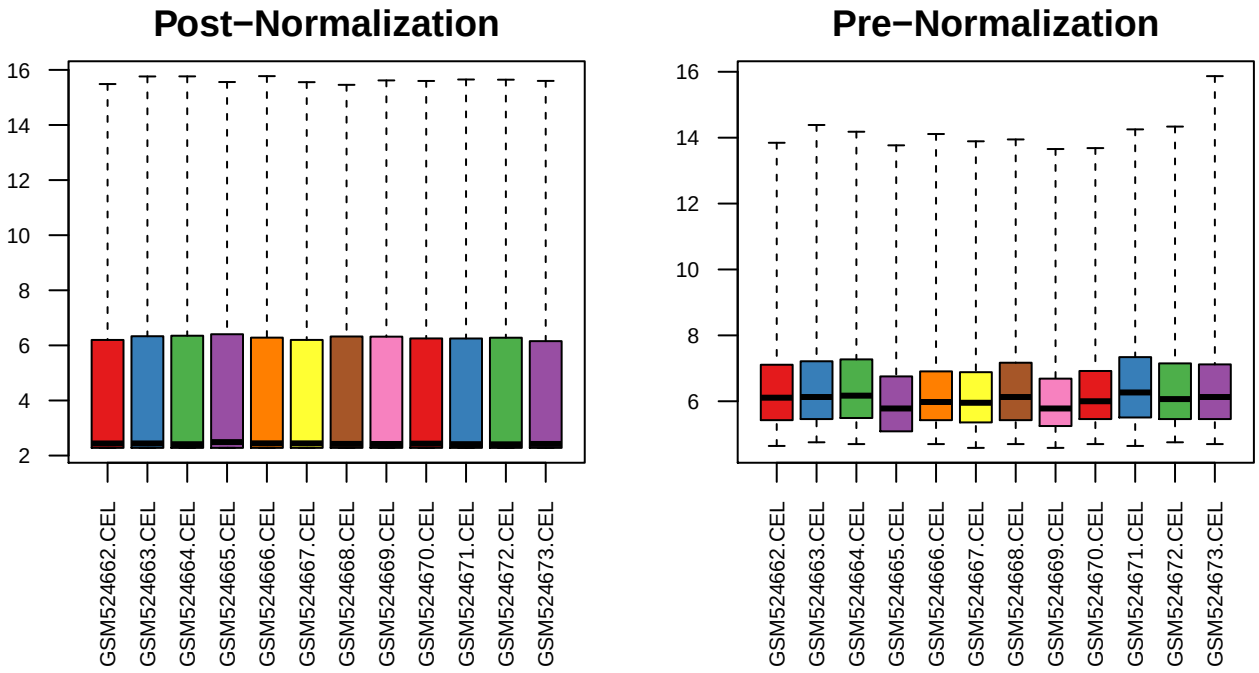

**Box Plots with vs. without normalization**

The previous box plots (using the `celfiles`, an `AffyBatch` object as input) were not based on normalized features. Here, we use the `gcrma()` function with `celfiles` as input to extract the `ExpressionSet` object `celfiles.gcrma`. When applying the `gcrma()` function, the default parameter `normalize` is set to true. Therefore, we obtain normalized data.

Comparison: After normalizing, we can observe the following.

- Subtle differences in the inter-quartile ranges across samples get reduced. Almost all inter-quartile ranges look very similar in the `Post-Normalization` plot whereas they noticeably differ in the `Pre-Normalization` plot.
- The medians decrease from values of around 6 to values of near 2.
- The upper quartile is much more spread out than the lower quartile in `Post-Normalization`. Likely, the lower quartiles correspond to genes that are not expressed whereas the upper quartiles correspond to genes that are expressed.

```
## Adjusting for optical effect............Done.
## Computing affinities.Done.
## Adjusting for non-specific binding............Done.
## Normalizing
## Calculating Expression
```




**Cluster analysis (normalized, .CEL names as labels)**

Here, we perform another cluster analyzis. This time, however, we use the normalized data, which we created for the box plot comparison above. We use the same distance metric for hierarchical clustering as before.

Again, we have 12 leaf nodes. However, the labels are now the names of the .CEL files (instead of their corresponding tissues). Still, we can see that the displayed dendogram differs from the dendogram that we received without normalization. This illustrates that normalization is indeed relevant here since it leads to different conclusions.

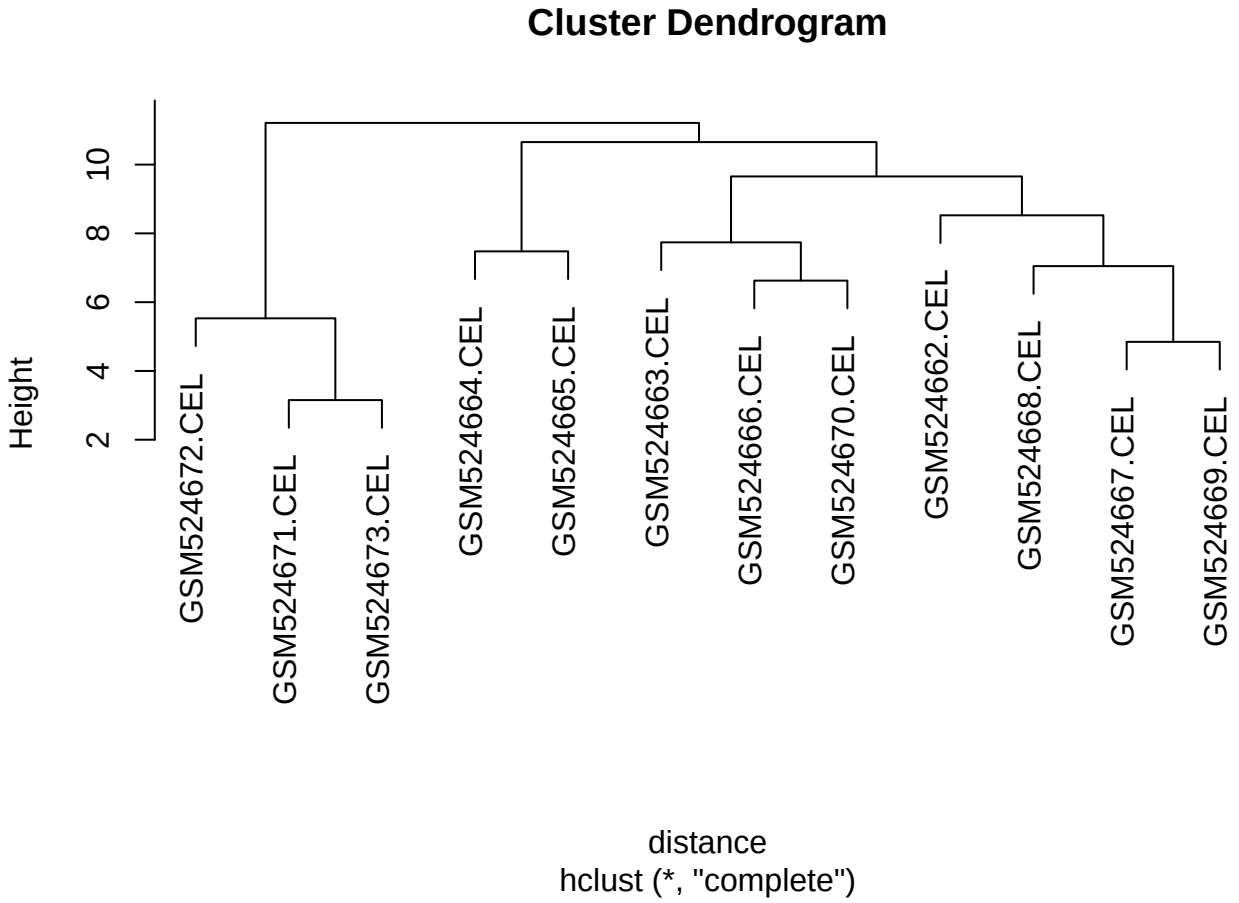



Table 1: Phenodata

| Name | FileName | Targets |
|---|---|---|
| GSM524662.CEL | GSM524662.CEL | iris |
| GSM524663.CEL | GSM524663.CEL | retina |
| GSM524664.CEL | GSM524664.CEL | retina |
| GSM524665.CEL | GSM524665.CEL | iris |
| GSM524666.CEL | GSM524666.CEL | retina |
| GSM524667.CEL | GSM524667.CEL | iris |
| GSM524668.CEL | GSM524668.CEL | choroid |
| GSM524669.CEL | GSM524669.CEL | choroid |
| GSM524670.CEL | GSM524670.CEL | choroid |
| GSM524671.CEL | GSM524671.CEL | huvec |
| GSM524672.CEL | GSM524672.CEL | huvec |
| GSM524673.CEL | GSM524673.CEL | huvec |

**Linear Regression using Contrast Matrix**

First, we create a model matrix. It has four columns, one for each tissue, and twelve rows, one for each sample (i.e. .CEL files). Every row only has a single 1 indicating which tissue the sample belongs to.

Second, we create a contrast matrix (using the model matrix as input to provide the tissue names). This contrast matrix has 4 rows, one for each tissue, and 3 columns, one for each contrast. The three contrasts are `huvec - choroid`, `huvec - retina` and `huvec - iris`. Basically, the `huvec` row contains a value of 1 for all columns (since it is a minuend). The `choroid`, `retina` and `iris` rows have a value of -1 in their

corresponding contrast column (since they are subtrahends in the contrasts).

Third, we use the `lmFit()` function from the `limma` package. It takes the normalized data in form of the `ExpressionSet` object `celfiles.gcrma` and it takes the model matrix `design` which specifies the contrasts that are supposed to be analyzed. The function fits a linear model to the expression data for each sample. According to the documentation, the coefficients of the fitted models describe the differences between the RNA sources hybridized to the arrays.

Fourth, we compute estimated coefficients and standard errors for a given set of contrasts, using the fitted model from the previous step. We also compute various statistics for these coefficients and standard errors such as t-statistics, F-statistics and logodds.

Finally, we print a summary of the results.

```
## [1] "sampleschoroid" "sampleshuvec"   "samplesiris"    "samplesretina"
```

Table 2: Design Matrix

| choroid | huvec | iris | retina |
|---|---|---|---|
| 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 |

```
##      logFC              AveExpr             t                P.Value
##  Min.   :-9.19111   Min.   : 2.279   Min.   :-39.77473   Min.   :0.0000
##  1st Qu.:-0.05967   1st Qu.: 2.281   1st Qu.: -0.70649   1st Qu.:0.1523
##  Median : 0.00000   Median : 2.480   Median :  0.00000   Median :0.5079
##  Mean   :-0.02353   Mean   : 4.375   Mean   :  0.07441   Mean   :0.5346
##  3rd Qu.: 0.03986   3rd Qu.: 6.241   3rd Qu.:  0.67455   3rd Qu.:1.0000
##  Max.   : 8.67086   Max.   :15.541   Max.   :296.84201   Max.   :1.0000
##
##    adj.P.Val            B             getsymbols
##  Min.   :0.0000   Min.   :-7.710   YME1L1  :   22
##  1st Qu.:0.6036   1st Qu.:-7.710   HFE     :   15
##  Median :1.0000   Median :-7.451   CFLAR   :   14
##  Mean   :0.7436   Mean   :-6.582   NRP2    :   14
##  3rd Qu.:1.0000   3rd Qu.:-6.498   ARHGEF12:   13
##  Max.   :1.0000   Max.   :21.290   (Other) :41857
##                                    NA's    :12740
```

## Assignment 2

**Task:**

- Present the variables versus each other original, log–scaled and MA–plot for each considered pair both before and after normalization.

- A cluster analysis is performed on the page but not report. Present plots and also draw heatmaps.

# Assignment 3

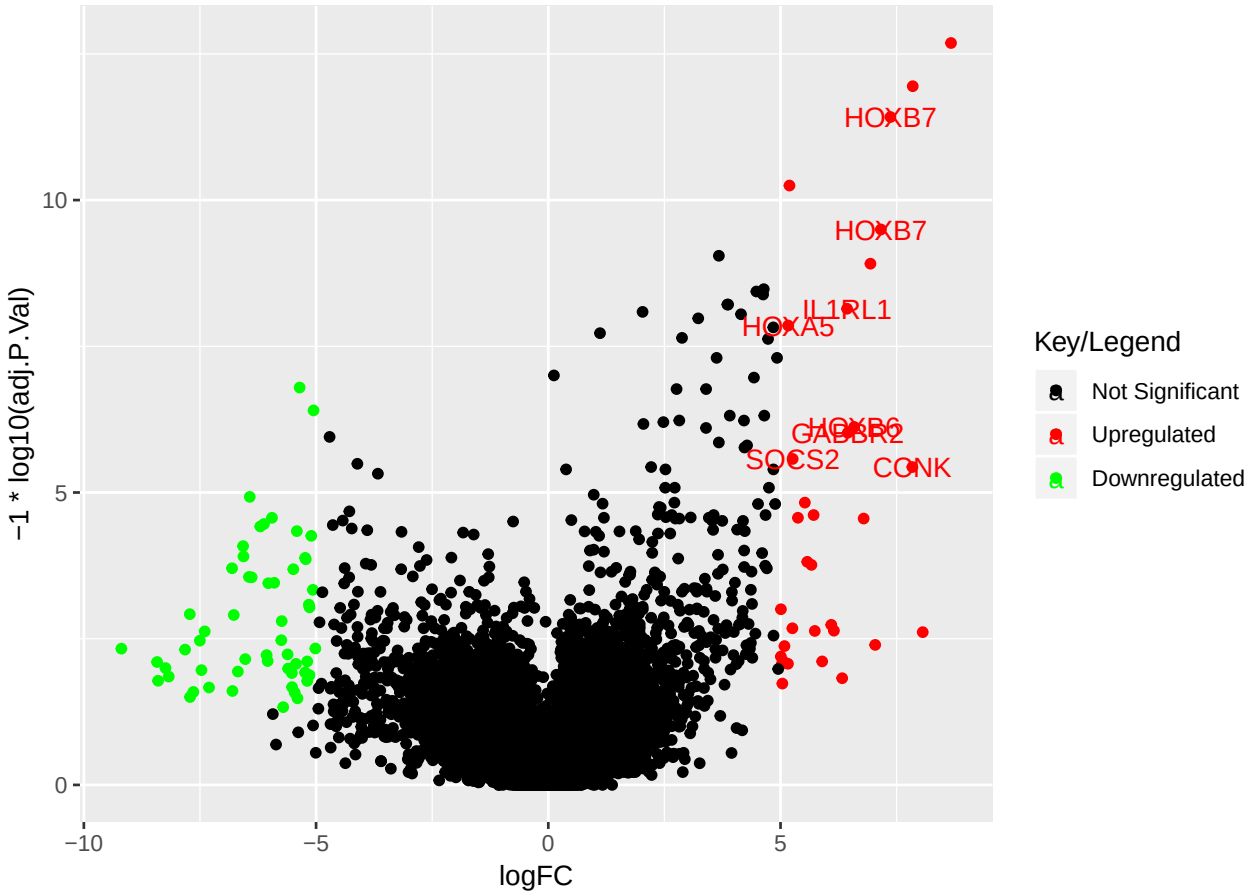
**Task:**

- Provide volcano plots for the other pairs.
- Indicate significantly differentially expressed genes.
- Explain how they are found.

## Volcano Plots

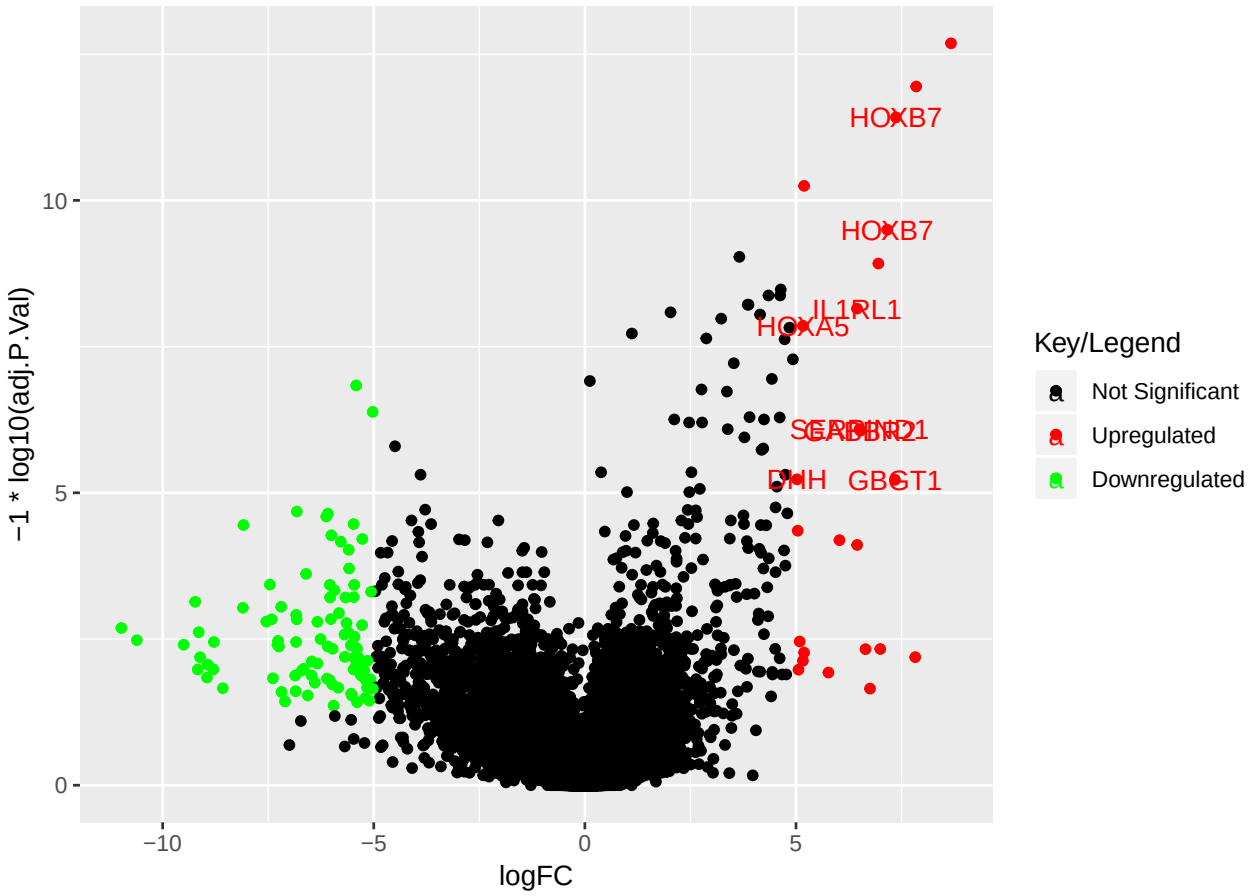
### Huvec - Choroid

```
##
##      1      2      3
## 54587     33     55
```



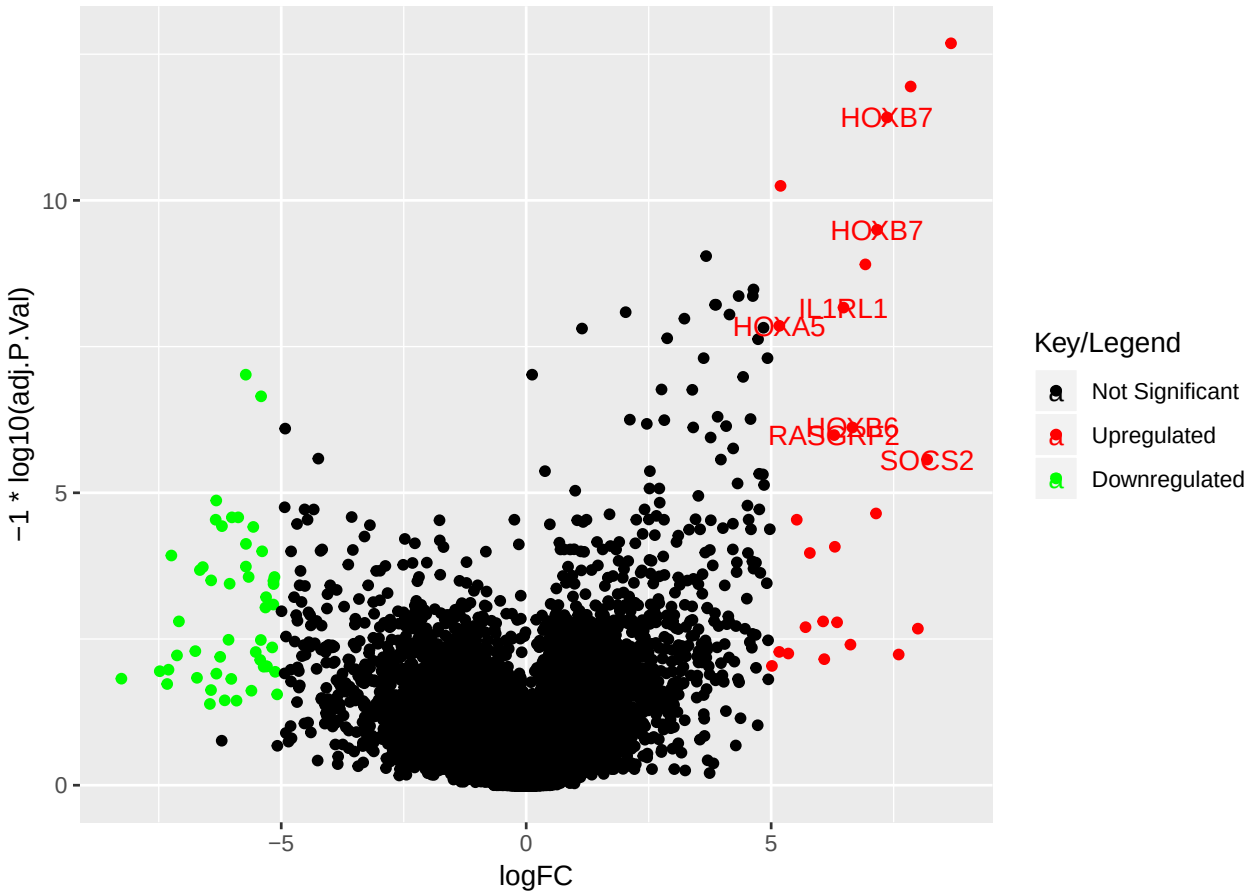

### Huvec - Retina

```
##
##      1      2      3
## 54557     24     94
```




**Huvec - Iris**

```
##
##     1     2     3
## 54601    25    49
```




## Significantly Differentially Expressed Genes

We observe the following differentially expresses genes:

```
##  [1] "HOXB7"    "IL1RL1"   "HOXA5"    "HOXB6"    "GABBR2"   "SOCS2"
##  [7] "CCNK"     "SERPIND1" "DHH"      "GBGT1"    "RASGRF2"
```

## Explanation How They Are Found

A volcano plot prints significance against fold-change. A fold-change is a measurement between two variables and is used as a measurement between how much they changed during measurements. So if we put on those values on the y- and x-axis we will get a view of the statistical signifiance and the magnitude of the a change. This enables the viewers to quickly recognize not only significant, but also “strong influencing” genes.

Therefore we have the `log2` of the fold-change on the x-axis and the `-log10` of the p-value on the y-axis. Thus interesting data points are those which are far to the top (p-value) and far to the left or right (significant change in the fold-change). In this one we can observe another feature of the data expressed in color. Here it is the regulation of the data.

The datapoints (genes) we obtained are those which are far to the right and in the upper part of the volcano plot. We used the provided code filtering those and adding the names to them as the seperator. All obtained genes are genes which are upregulated.

### Assignment 4

**Task:**

- Try to find more information on the genes that are reported to be significantly differentially expressed. Report in your own words on what you find.
- Report all the Gene Ontology (GO) terms associated with each gene.
- Are any of the GO terms common between genes?
- If so do the common GO terms seem to be related to anything particular?
- Try to present GO analysis in an informative manner, if possible visualize.

# Appendix

```
knitr::opts_chunk$set(fig.width = 7, fig.height = 5, echo = FALSE,
                      warning = FALSE, message = FALSE)

# All privided Links:
# R Bio: Untangling Genomes
# https://www.bioconductor.org/help/course-materials/ 2015/Uruguay2015/
# Step by Step HUVEC and OVE
# https://www.bioconductor.org/help/course-materials/2015/Uruguay2015/ day3-gene.expression.html
# Cell Data:
# ftp://ftp.ncbi.nlm.nih.gov/geo/series/GSE20nnn/GSE20986/suppl/
# Description of Cell Data:
# https://www.ncbi.nlm.nih.gov/geo/query/acc. cgi?acc=GSE20986
# Additional
# https://www.bioconductor.org/help/course-materials/ 2015/Uruguay2015/day5-data_analysis.html
# Explanations Graphic
# https://www.bioconductor.org/help/course-materials/2015/Uruguay2015/V6-RNASeq.html

library(ggplot2)

# Use this if BiocManager is not installed
#if (!requireNamespace("BiocManager", quietly = TRUE))
#    install.packages("BiocManager")
#library("BiocManager")

# BiocManager packages
# BiocManager::install("GEOquery", version = "3.8")
# BiocManager::install("simpleaffy", version = "3.8")
# BiocManager::install("RColorBrewer", version = "3.8")
# BiocManager::install("affyPLM", version = "3.8")
# BiocManager::install("limma", version = "3.8")
# BiocManager::install("annotate", version = "3.8")
# BiocManager::install("hgu133plus2.db", version = "3.8")
library(GEOquery)
library(simpleaffy)
library(RColorBrewer)
library(affyPLM)
library(limma)
library(hgu133plus2.db)
library(annotate)
```

```r
# ------------------------------------------------------------------------------
# Question 1
# ------------------------------------------------------------------------------

# Data import ------------------------------------------------------------------

# Get the Data
x = getGEOSuppFiles("GSE20986")
x

# Untar and Unzip
# DONT ADD THE FILE TO GIT IT'S 60MB!
untar("GSE20986/GSE20986_RAW.tar", exdir = "data")
cels = list.files("data/", pattern = "[gz]")
sapply(paste("data", cels, sep = "/"), gunzip)


# Creating phenodata -----------------------------------------------------------

# It's a matrix with two columns. Each row has the same entries in their cells,
# which is the filename. The first column is called 'Name' and the second one
# is called 'FileName'.
phenodata = matrix(rep(list.files("data", pattern = "[CEL]"), 2), ncol =2)

# class(phenodata) # "matrix"

phenodata <- as.data.frame(phenodata)
colnames(phenodata) <- c("Name", "FileName")
# Adding a new column with the target
phenodata$Targets <- c("iris",
                       "retina",
                       "retina",
                       "iris",
                       "retina",
                       "iris",
                       "choroid",
                       "choroid",
                       "choroid",
                       "huvec",
                       "huvec",
                       "huvec")

# Writes the dataframe to a .txt file
write.table(phenodata, "data/phenodata.txt", quote = F, sep = "\t",
            row.names = F)

# Now the data is read again and a boxplot is created
celfiles <- simpleaffy::read.affy(covdesc = "phenodata.txt", path = "data")

# Simple box plot --------------------------------------------------------------

# Creates a boxplot for the log base 2 intensities including pm (perfect) and
# mm (mismatch)
```

```r
BiocGenerics::boxplot(celfiles)


# Improved box plot ------------------------------------------------------------

# Create "nice looking" color palettes. So this array actually has hex-color
# values inside.
cols = brewer.pal(8, "Set1")

# Help page: exprs, AffyBatch-method
# The columns are the different .CEL files and hold the data. It's a class re-
# presentation for the probe level data. The main component are the intensities
# from multiple arrays of the save CDF type.
eset <- affy::exprs(celfiles)
samples <- celfiles$Targets
# colnames(eset) # Print colnames

# The colnames are set to our targets
colnames(eset) <- samples

# The .CEL files and their content is being plotted. It's basically the same
# plot as before but this time with fancy colors
par(mar=c(8,2,2,2)) # bottom, left, top and right margins
BiocGenerics::boxplot(celfiles, col = cols, las = 2)


# Creating a distance matrix
distance <- stats::dist(t(eset), method = "maximum")

# "Hierarchical cluster analysis on a set of dissimilarities and methods for analyzing it."
clusters <- stats::hclust(distance)

# Plot the clusters in a tree diagram
# (https://en.wikipedia.org/wiki/Dendrogram)
plot(clusters)


# Box Plots with vs. without normalization -------------------------------------

# "Robust Multi-Array expression measure using sequence infornmation"
# Converts an AffyBatch into an ExpressionSet by using RMA (robust multi-array)
# expression measure with the help of probe sequence
celfiles.gcrma = gcrma::gcrma(celfiles) # Biobase / ExpressionSet

# Two Boxplots showing different .CELS before and after normalization
par(mfrow=c(1,2),
    mar=c(10,2,2,2)) # bottom, left, top and right margins
boxplot(celfiles.gcrma, col = cols, las = 2.5, cex.axis = 0.7, main = "Post-Normalization");
boxplot(celfiles, col = cols, las = 2.5, cex.axis = 0.7, main = "Pre-Normalization")

# Turns of the "devices"
# dev.off()
```

```r
# Create a distance matrix
distance <- dist(t(exprs(celfiles.gcrma)), method = "maximum")

# Performs a hierachical cluster analysis.
clusters <- hclust(distance)

# Plots the clusters, which is a Dendogram again
plot(clusters)
knitr::kable(phenodata, caption = "Phenodata")


# Create a model matrix
samples <- as.factor(samples) # Factor of length 12 with tissue names
design <- model.matrix(~0+samples) # Creates model matrix by expanding factors
colnames(design) # "sampleschoroid" "sampleshuvec"   "samplesiris"    "samplesretina"
colnames(design) <- c("choroid", "huvec", "iris", "retina") # change colnames
knitr::kable(design, caption = "Design Matrix") # Print design matrix

# Create a contrast matrix from the model matrix
contrast.matrix = makeContrasts(huvec_choroid = huvec - choroid,
                                huvec_retina = huvec - retina,
                                huvec_iris = huvec - iris,
                                levels = design)

# Fits a linear model for each gene given a series of arrays
# Results from the RMA and the contrast.matrix
fit = limma::lmFit(celfiles.gcrma, design)

# Use the model to fit microdata data (coefficients, errors)
huvec_fit <- contrasts.fit(fit, contrast.matrix)

# Calculates t-statistics, F-statistics and logodds for the fitted data
huvec_ebay <- eBayes(huvec_fit)

# Creates a list of probe name for 100000 entries. topTable takes the top-ranked
# genes from a linear fit
probenames.list <- rownames(topTable(huvec_ebay, number = 100000))

# Get symbols
getsymbols <- getSYMBOL(probenames.list, "hgu133plus2")

# And get the 100000 by the coefficient huvec_choroid
results <- topTable(huvec_ebay, number = 100000, coef = "huvec_choroid")

# Add the symbols to this
results <- cbind(results, getsymbols)

# Prints the summary of our results
summary(results)

# "Extract Data" depending on the p value and logFC
results$threshold <- "1"
a <- subset(results, adj.P.Val < 0.05 & logFC > 5)
```

```
results[rownames(a), "threshold"] <- "2"
b <- subset(results, adj.P.Val < 0.05 & logFC < -5)
results[rownames(b), "threshold"] <- "3"

# ------------------------------------------------------------------------------
# Question 2
# ------------------------------------------------------------------------------

# ------------------------------------------------------------------------------
# Question 3
# ------------------------------------------------------------------------------

significant_genes = c()

# ------------------------------------------------------------------------------
# Huvec - Choroid
# ------------------------------------------------------------------------------

current_genes = subset(results, logFC > 5 & -1*log10(adj.P.Val) > 5)
significant_genes = c(significant_genes, as.vector(current_genes$getsymbols))

table(results$threshold)

# Make a volcano plot
volcano <- ggplot(data = results,
                  aes(x = logFC, y = -1*log10(adj.P.Val),
                      colour = threshold,
                      label = getsymbols))
volcano <- volcano +
  geom_point() +
  scale_color_manual(values = c("black", "red", "green"),
                labels = c("Not Significant", "Upregulated", "Downregulated"),
                name = "Key/Legend")
volcano +
  geom_text(data = subset(results, logFC > 5 & -1*log10(adj.P.Val) > 5),
            aes(x = logFC, y = -1*log10(adj.P.Val), colour = threshold,
                label = getsymbols))

# These are the other combinatin where we need volcano plots for
# huvec_retina
# huvec_iris

# ------------------------------------------------------------------------------
# Huvec - Retina
# ------------------------------------------------------------------------------

results <- topTable(huvec_ebay, number = 100000, coef = "huvec_retina")
results <- cbind(results, getsymbols)

current_genes = subset(results, logFC > 5 & -1*log10(adj.P.Val) > 5)
significant_genes = c(significant_genes, as.vector(current_genes$getsymbols))

results$threshold <- "1"
```

```r
a <- subset(results, adj.P.Val < 0.05 & logFC > 5)
results[rownames(a), "threshold"] <- "2"
b <- subset(results, adj.P.Val < 0.05 & logFC < -5)
results[rownames(b), "threshold"] <- "3"
table(results$threshold)

# Make a volcano plot
volcano2 <- ggplot(data = results,
                   aes(x = logFC, y = -1*log10(adj.P.Val),
                       colour = threshold,
                       label = getsymbols))

volcano2 <- volcano2 +
  geom_point() +
  scale_color_manual(values = c("black", "red", "green"),
                 labels = c("Not Significant", "Upregulated", "Downregulated"),
                 name = "Key/Legend")

volcano2 +
  geom_text(data = subset(results, logFC > 5 & -1*log10(adj.P.Val) > 5),
            aes(x = logFC, y = -1*log10(adj.P.Val), colour = threshold,
                label = getsymbols))


# ------------------------------------------------------------------------------
# Huvec - Iris
# ------------------------------------------------------------------------------

results <- topTable(huvec_ebay, number = 100000, coef = "huvec_iris")
results <- cbind(results, getsymbols)

current_genes = subset(results, logFC > 5 & -1*log10(adj.P.Val) > 5)
significant_genes = c(significant_genes, as.vector(current_genes$getsymbols))

results$threshold <- "1"
a <- subset(results, adj.P.Val < 0.05 & logFC > 5)
results[rownames(a), "threshold"] <- "2"
b <- subset(results, adj.P.Val < 0.05 & logFC < -5)
results[rownames(b), "threshold"] <- "3"
table(results$threshold)

# Make a volcano plot
volcano2 <- ggplot(data = results,
                   aes(x = logFC, y = -1*log10(adj.P.Val),
                       colour = threshold,
                       label = getsymbols))

volcano2 <- volcano2 +
  geom_point() +
  scale_color_manual(values = c("black", "red", "green"),
                 labels = c("Not Significant", "Upregulated", "Downregulated"),
                 name = "Key/Legend")
```

```r
volcano2 +
  geom_text(data = subset(results, logFC > 5 & -1*log10(adj.P.Val) > 5),
            aes(x = logFC, y = -1*log10(adj.P.Val), colour = threshold,
                label = getsymbols))


significant_genes = unique(significant_genes)
significant_genes = significant_genes[!is.na(significant_genes)]
print(significant_genes)

# ------------------------------------------------------------------------------
# Question 4
# ------------------------------------------------------------------------------
```