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.
- Go carefully through the R code and explain in your words what each step does.

Provided source code we have to explain.

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

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

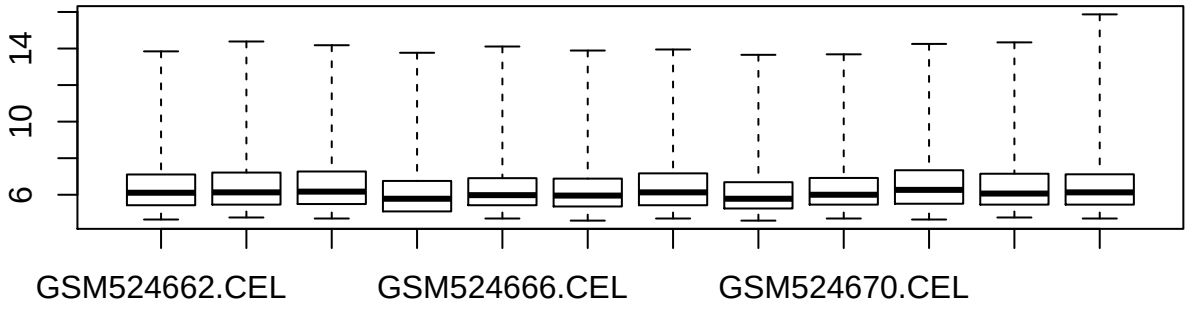
```
##                                                                    size
## /Users/flennic/git/bioinformatics/Lab4/GSE20986/GSE20986_RAW.tar 56360960
##                                                                   isdir
## /Users/flennic/git/bioinformatics/Lab4/GSE20986/GSE20986_RAW.tar FALSE
##                                                                    mode
## /Users/flennic/git/bioinformatics/Lab4/GSE20986/GSE20986_RAW.tar  644
##                                                                           mtime
## /Users/flennic/git/bioinformatics/Lab4/GSE20986/GSE20986_RAW.tar 2018-12-07 15:50:32
##                                                                           ctime
## /Users/flennic/git/bioinformatics/Lab4/GSE20986/GSE20986_RAW.tar 2018-12-07 15:50:32
##                                                                           atime
## /Users/flennic/git/bioinformatics/Lab4/GSE20986/GSE20986_RAW.tar 2018-12-07 15:50:19
##                                                                   uid gid
## /Users/flennic/git/bioinformatics/Lab4/GSE20986/GSE20986_RAW.tar 501  20
##                                                                     uname
## /Users/flennic/git/bioinformatics/Lab4/GSE20986/GSE20986_RAW.tar flennic
##                                                                   grname
## /Users/flennic/git/bioinformatics/Lab4/GSE20986/GSE20986_RAW.tar  staff
```

```
# 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)
```

```
## data/GSM524662.CEL.gz data/GSM524663.CEL.gz data/GSM524664.CEL.gz
##              13555726              13555055              13555639
## data/GSM524665.CEL.gz data/GSM524666.CEL.gz data/GSM524667.CEL.gz
##              13560122              13555663              13557614
## data/GSM524668.CEL.gz data/GSM524669.CEL.gz data/GSM524670.CEL.gz
##              13556090              13560054              13555971
## data/GSM524671.CEL.gz data/GSM524672.CEL.gz data/GSM524673.CEL.gz
```

```
##              13554926              13555042              13555290
```

```
# 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)
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 <- read.affy(covdesc = "phenodata.txt", path = "data")
# Creates a boxplot for the log base 2 intensities including pm (perfect) and
# mm (mismatch)
boxplot(celfiles)
```

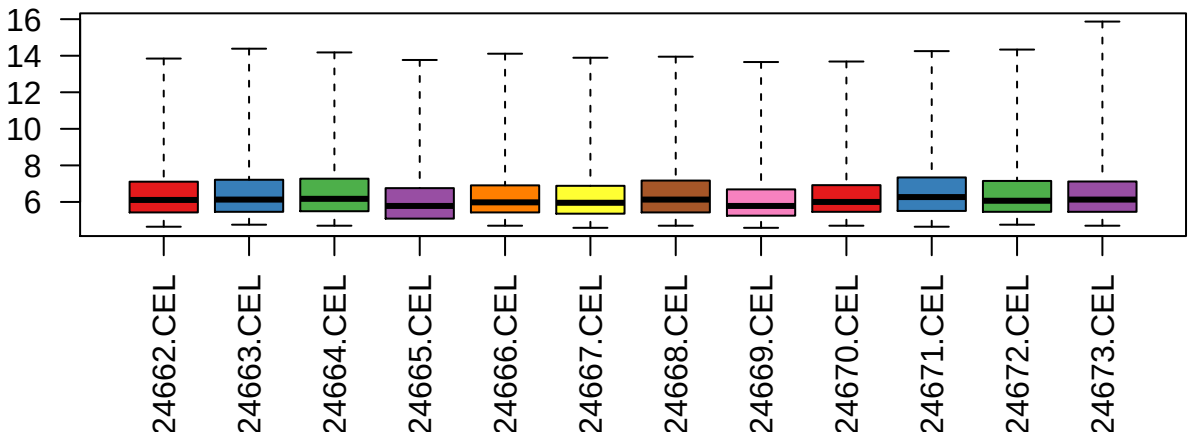



```
# Create "nice looking" color palettes. So this array actually has hex-color
# values inside.
cols = brewer.pal(8, "Set1")
# Help page: exprs,AffyBatch-method
# The columns are the different .CEL files and hold the data. It's a class re-
# presentation for the probe level data. The main component are the intensities
# from multiple arrays of the save CDF type.
eset <- exprs(celfiles)
samples <- celfiles$Targets
colnames(eset)
```

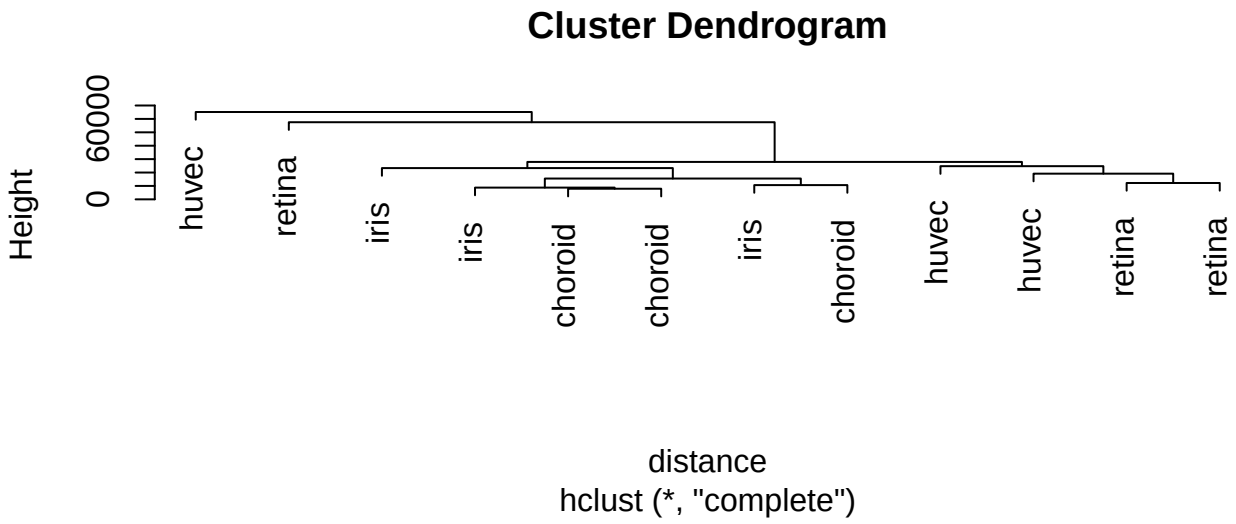
```
##  [1] "GSM524662.CEL" "GSM524663.CEL" "GSM524664.CEL" "GSM524665.CEL"
##  [5] "GSM524666.CEL" "GSM524667.CEL" "GSM524668.CEL" "GSM524669.CEL"
```

```
##  [9] "GSM524670.CEL" "GSM524671.CEL" "GSM524672.CEL" "GSM524673.CEL"
```

```r
# 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
boxplot(celfiles, col = cols, las = 2)
```




```r
# Creating a distance matrix
distance <- dist(t(eset), method = "maximum")
# "Hierarchical cluster analysis on a set of dissimilarities and methods for analyzing it."
clusters <- hclust(distance)
# Plot the clusters in a tree diagram
# (https://en.wikipedia.org/wiki/Dendrogram)
plot(clusters)
```

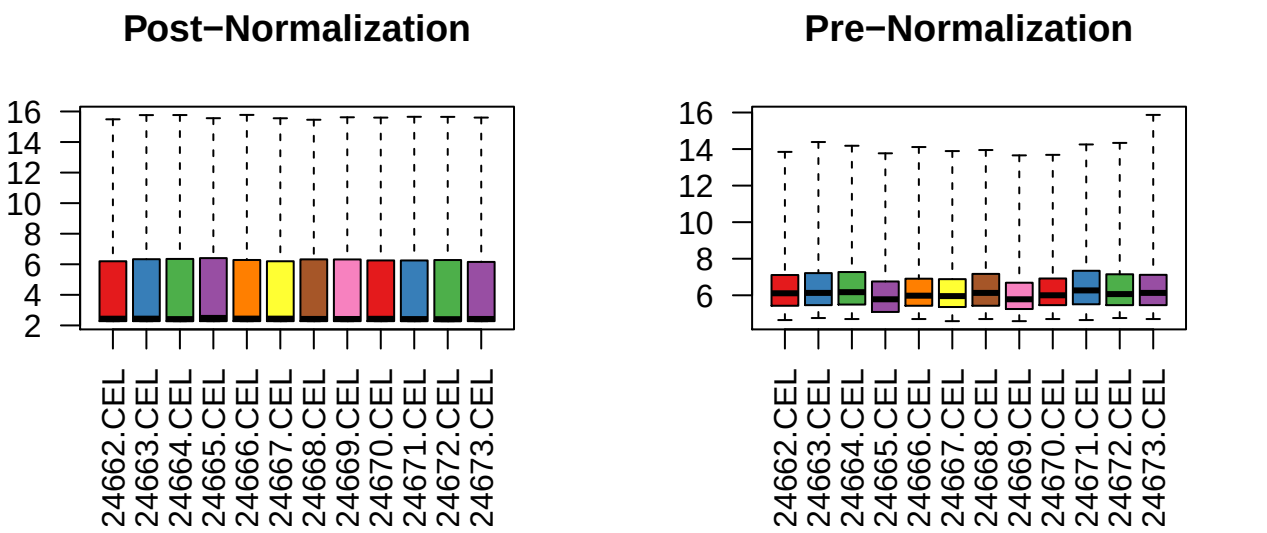



```r
# "Robust Multi-Array expression measure using sequence infornmation"
# Converts an AffyBatch into an ExpressionSet by using RMA (robust multi-array)
# expression measure with the help of probe sequence
celfiles.gcrma = gcrma(celfiles) # Biobase / ExpressionSet
```

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

```
## Calculating Expression
```

```
# These are two Boxplots showing the different .CELS before and after normali-
# zation
par(mfrow=c(1,2))
boxplot(celfiles.gcrma, col = cols, las = 2, main = "Post-Normalization");
boxplot(celfiles, col = cols, las = 2, main = "Pre-Normalization")
```




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

```
## null device
##           1
```

```
# 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)

phenodata
```

```
##             Name      FileName Targets
## 1  GSM524662.CEL GSM524662.CEL    iris
## 2  GSM524663.CEL GSM524663.CEL  retina
## 3  GSM524664.CEL GSM524664.CEL  retina
## 4  GSM524665.CEL GSM524665.CEL    iris
## 5  GSM524666.CEL GSM524666.CEL  retina
## 6  GSM524667.CEL GSM524667.CEL    iris
## 7  GSM524668.CEL GSM524668.CEL choroid
## 8  GSM524669.CEL GSM524669.CEL choroid
## 9  GSM524670.CEL GSM524670.CEL choroid
## 10 GSM524671.CEL GSM524671.CEL   huvec
## 11 GSM524672.CEL GSM524672.CEL   huvec
## 12 GSM524673.CEL GSM524673.CEL   huvec
```

```
samples <- as.factor(samples)
# Creates a design/model matrix by expanding factors.
design <- model.matrix(~0+samples)
```

```
# "sampleschoroid" "sampleshuvec"   "samplesiris"    "samplesretina"
colnames(design)
```

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

```
# Rename from "sample" to this
colnames(design) <- c("choroid", "huvec", "iris", "retina")
design
```

```
##    choroid huvec iris retina
## 1        0     0    1      0
## 2        0     0    0      1
## 3        0     0    0      1
## 4        0     0    1      0
## 5        0     0    0      1
## 6        0     0    1      0
## 7        1     0    0      0
## 8        1     0    0      0
## 9        1     0    0      0
## 10       0     1    0      0
## 11       0     1    0      0
## 12       0     1    0      0
## attr(,"assign")
## [1] 1 1 1 1
## attr(,"contrasts")
## attr(,"contrasts")$samples
## [1] "contr.treatment"
```

```
# "Construct the contrast matrix corresponding to specified contrasts of a se
# of parameters."
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 = 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 tkaes 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
```

```r
summary(results)
```

```
##      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
```

```r
# "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"
```

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

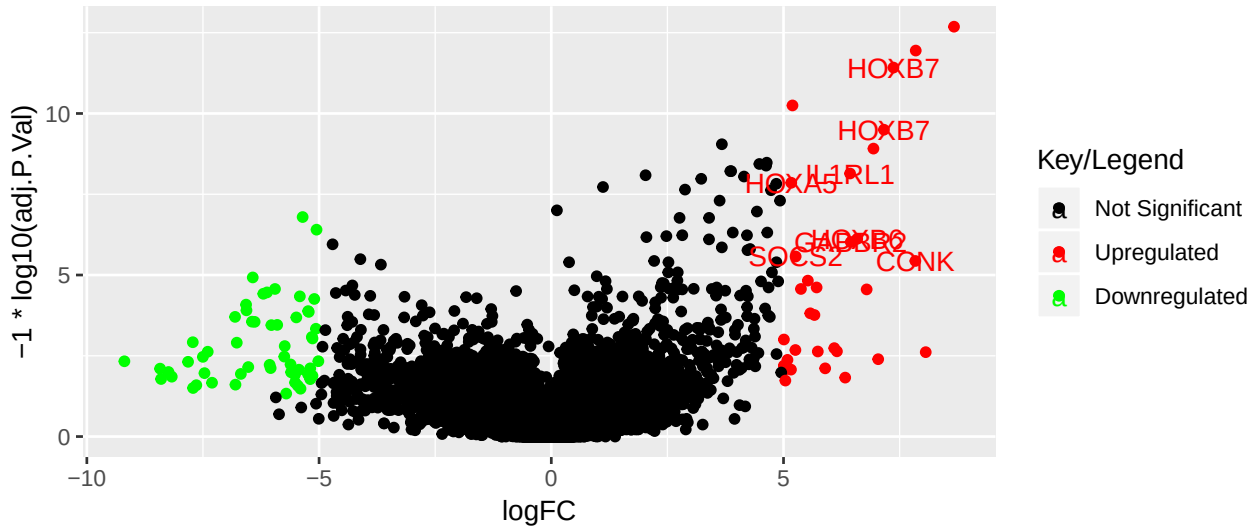
Task:

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

Volcano Plots

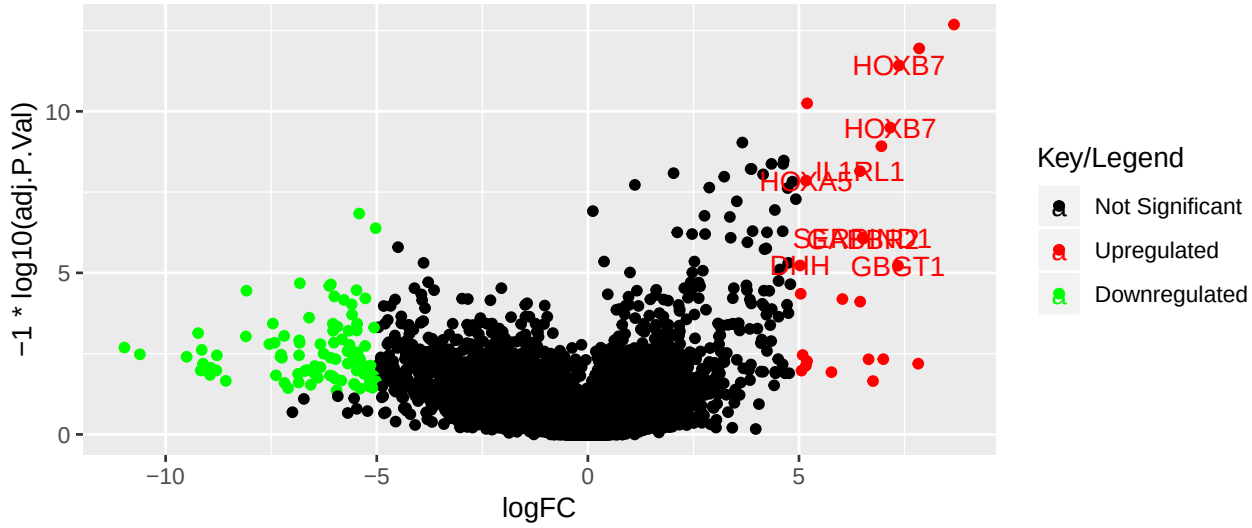
Huvec - Choroid

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



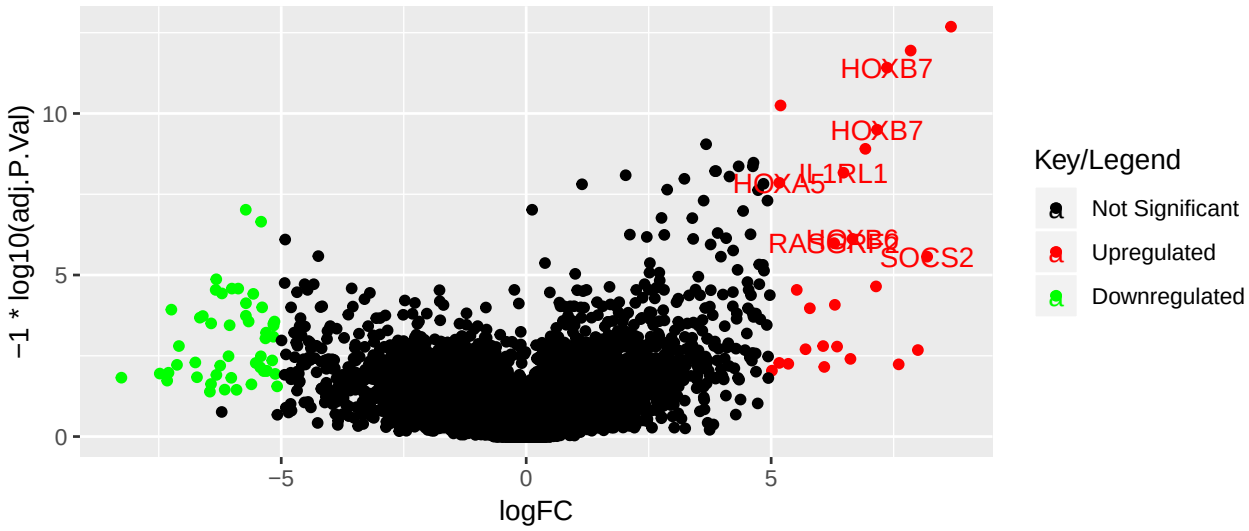
Huvec - Retina

```
##
##     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 = 3, 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)


# ------------------------------------------------------------------------------
# Question 1
# ------------------------------------------------------------------------------

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

# Untar and Unzip
# DONT ADD THE FILE TO GIT IT'S 60MB!
```

```r
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)
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 <- read.affy(covdesc = "phenodata.txt", path = "data")
# Creates a boxplot for the log base 2 intensities including pm (perfect) and
# mm (mismatch)
boxplot(celfiles)

# 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 <- exprs(celfiles)
samples <- celfiles$Targets
colnames(eset)
# 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
boxplot(celfiles, col = cols, las = 2)
# Creating a distance matrix
distance <- dist(t(eset), method = "maximum")
# "Hierarchical cluster analysis on a set of dissimilarities and methods for analyzing it."
clusters <- hclust(distance)
```

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

# These are two Boxplots showing the different .CELS before and after normali-
# zation
par(mfrow=c(1,2))
boxplot(celfiles.gcrma, col = cols, las = 2, main = "Post-Normalization");
boxplot(celfiles, col = cols, las = 2, main = "Pre-Normalization")

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

# 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)

phenodata

samples <- as.factor(samples)
# Creates a design/model matrix by expanding factors.
design <- model.matrix(~0+samples)
# "sampleschoroid" "sampleshuvec"   "samplesiris"    "samplesretina"
colnames(design)

# Rename from "sample" to this
colnames(design) <- c("choroid", "huvec", "iris", "retina")
design

# "Construct the contrast matrix corresponding to specified contrasts of a se
# of parameters."
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 = 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 tkaes the top-ranked
```

```r
# 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))
```

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

```r
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))


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

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