Quiz 4

Monikrishna Roy

2021-10-27

Question 1

Install the library GEOquery from Bioconductor. Provide the code for installation,

Hint: use google search to find the library and then follow the instruction on Bioconductor to install the library

```
if (!requireNamespace("BiocManager", quietly = TRUE))
    install.packages("BiocManager")

BiocManager::install("GEOquery", update = F)
```

```
## 'getOption("repos")' replaces Bioconductor standard repositories, see
## '?repositories' for details
##
## replacement repositories:
##     CRAN: https://cloud.r-project.org
```

```
## Bioconductor version 3.10 (BiocManager 1.30.16), R 3.6.3 (2020-02-29)
```

```
## Warning: package(s) not installed when version(s) same as current; use `force = TRUE` to
##   re-install: 'GEOquery'
```

Question 2

Retrieve the expression data of the dataset GSE19804 from NIH Gene Expression Omnibus (GEO)

Hint: First, copy the file GSE19804_series_matrix.txt to your working folder and use the the function getGEO() from the library GEOQuery to get the whole object. Next, use the function exprs() to retrieve the data matrix. eset<-getGEO("GSE19804", filename="GSE19804_series_matrix.txt") data <- exprs(eset)

```
library(GEOquery)
```

```
## Loading required package: Biobase
```

```
## Loading required package: BiocGenerics
```

```
## Loading required package: parallel
```

```
##
## Attaching package: 'BiocGenerics'
```

```
## The following objects are masked from 'package:parallel':
##
##     clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,
##     clusterExport, clusterMap, parApply, parCapply, parLapply,
##     parLapplyLB, parRapply, parSapply, parSapplyLB
## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs

## The following objects are masked from 'package:base':
##
##     anyDuplicated, append, as.data.frame, basename, cbind, colnames,
##     dirname, do.call, duplicated, eval, evalq, Filter, Find, get, grep,
##     grepl, intersect, is.unsorted, lapply, Map, mapply, match, mget,
##     order, paste, pmax, pmax.int, pmin, pmin.int, Position, rank,
##     rbind, Reduce, rownames, sapply, setdiff, sort, table, tapply,
##     union, unique, unsplit, which, which.max, which.min

## Welcome to Bioconductor
##
##     Vignettes contain introductory material; view with
##     'browseVignettes()'. To cite Bioconductor, see
##     'citation("Biobase")', and for packages 'citation("pkgname")'.

## Setting options('download.file.method.GEOquery'='auto')

## Setting options('GEOquery.inmemory.gpl'=FALSE)
```

```r
dataset <- "GSE19804"
gsets <- getGEO(dataset, GSEMatrix = T, getGPL = T)
```

```
## Found 1 file(s)

## GSE19804_series_matrix.txt.gz

## Rows: 54675 Columns: 121

## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr   (1): ID_REF
## dbl (120): GSM494556, GSM494557, GSM494558, GSM494559, GSM494560, GSM494561,...

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.

## File stored at:

## /tmp/RtmpOQL8DA/GPL570.soft
```

```r
gset <- gsets[[1]]
expr <- exprs(gset)
```

Question 3

Retrieve the group information (cancer vs control) of the dataset GSE19804 from NIH GEO.

Hint: use function pData() to retrieve the group information. For example: pdata <- pData(eset) control<-rownames(pdata[grep("Lung Normal",pdata$title),]) cancer<-rownames(pdata[grep("Lung Cancer",pdata$title),])

```r
pdata <- pData(gset)
control <- rownames(pdata[grep("Lung Normal", pdata$title), ])
cancer <- rownames(pdata[grep("Lung Cancer", pdata$title), ])
```

Question 4

Perform a t-test to compare the cancer against control groups, compute the difference in mean log base 2 expression and create an output data frame that contains the following columns: gene ids (row names of expression data matrix), p-value, t-score, logFC

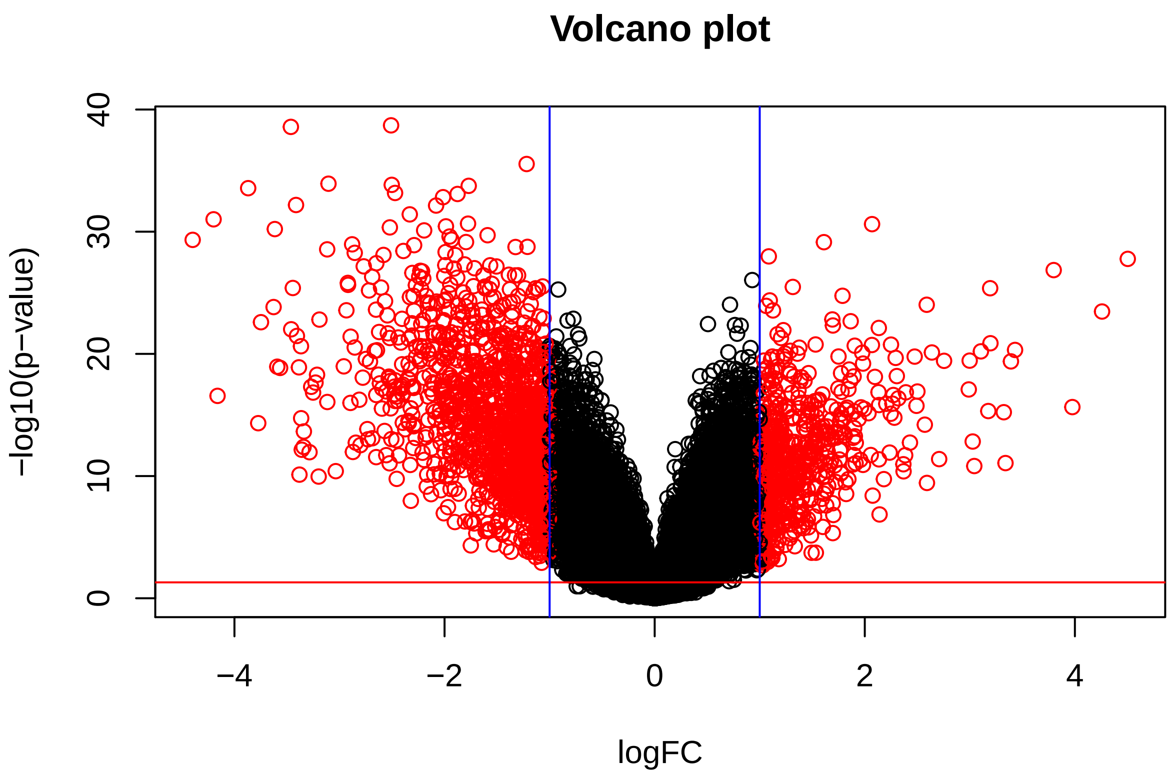
Hint: note that the data downloaded from NIH GEO is already in log scale.

```
# a function to calculate the difference in mean
cal_mean_diff <- function(x, cancer, control) {
  mean(x[cancer]) - mean(x[control])
}
# Function to calculate p-value
cal_p_value <- function(x, cancer, control) {
  t.test(x[cancer], x[control])$p.value
}
# Function to calculate t-score
cal_t_score <- function(x, cancer, control) {
  t.test(x[cancer], x[control])$statistic
}


# used apply to call the function
logFC <- apply(expr, 1, cal_mean_diff, cancer, control)
PValue <- apply(expr, MARGIN = 1, FUN = cal_p_value, cancer, control)
TScore <- apply(expr, MARGIN = 1, FUN = cal_t_score, cancer, control)

# rownames used as gene ids
geneIds <- rownames(expr)

df <- data.frame(
  row.names = NULL,
  "GeneID" = geneIds,
  "PValue" = PValue,
  "TScore" = TScore,
  "LogFC" = logFC
)
```

Question 5

Use absolute log fold change > 1 and raw p-value < 0.05 to select the differentially expressed (DE) genes. Show the Volcano plot. Color the DE genes in red.

```
plot(
  x = df$LogFC,
  y = -log10(df$PValue),
  xlab = 'logFC',
  ylab = '-log10(p-value)',
  main = "Volcano plot",
  col = ifelse(abs(df$LogFC) > 1 & df$PValue < 0.05, 'red', 'black')
)
abline(h = -log10(0.05), col = "red")
abline(v = -1, col = "blue")
abline(v = 1, col = "blue ")
```


Volcano plot
−log10(p−value)
0
10
20
30
40
−4
−2
0
2
4
logFC