3/4/2022

High Dimensional Statistics Project

AM: P3622004

Mihalis Galanakis

High Dimensional Statistics Project

Mihalis Galanakis

3/4/2022

Question 1 & 3

Exploring & visualizing the data

```
library("leukemiasEset")
```

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

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

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

```
## 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, gr
ep,
##     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.max, which.min
```

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

```
data(leukemiasEset)
x <- exprs(leukemiasEset)
str(x)
```

```
##  num [1:20172, 1:60] 3.39 3.54 9.82 4.75 3.31 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : chr [1:20172] "ENSG00000000003" "ENSG00000000005" "ENSG00000
000419" "ENSG00000000457" ...
##   ..$ : chr [1:60] "GSM330151.CEL" "GSM330153.CEL" "GSM330154.CEL" "
GSM330157.CEL" ...
```

```
summary(x)
```

```
##  GSM330151.CEL    GSM330153.CEL    GSM330154.CEL    GSM330157.CEL   
##  Min.   : 2.573   Min.   : 2.463   Min.   : 2.418   Min.   : 2.548  
##  1st Qu.: 3.842   1st Qu.: 4.033   1st Qu.: 3.782   1st Qu.: 3.720  
##  Median : 5.064   Median : 5.215   Median : 5.093   Median : 5.038  
##  Mean   : 5.520   Mean   : 5.501   Mean   : 5.537   Mean   : 5.531  
##  3rd Qu.: 6.816   3rd Qu.: 6.604   3rd Qu.: 6.909   3rd Qu.: 7.023  
##  Max.   :14.556   Max.   :14.407   Max.   :14.504   Max.   :14.586  
##  GSM330171.CEL    GSM330174.CEL    GSM330178.CEL    GSM330182.CEL   
##  Min.   : 2.464   Min.   : 2.549   Min.   : 2.511   Min.   : 2.610  
##  1st Qu.: 3.912   1st Qu.: 3.726   1st Qu.: 3.861   1st Qu.: 3.753  
##  Median : 5.138   Median : 5.015   Median : 5.152   Median : 5.077  
##  Mean   : 5.536   Mean   : 5.522   Mean   : 5.582   Mean   : 5.538  
##  3rd Qu.: 6.745   3rd Qu.: 6.997   3rd Qu.: 6.915   3rd Qu.: 6.988  
##  Max.   :14.516   Max.   :14.462   Max.   :14.326   Max.   :14.524  
##  GSM330185.CEL    GSM330186.CEL    GSM330195.CEL    GSM330201.CEL   
##  Min.   : 2.575   Min.   : 2.537   Min.   : 2.492   Min.   : 2.564  
##  1st Qu.: 3.840   1st Qu.: 3.851   1st Qu.: 3.733   1st Qu.: 3.800  
##  Median : 5.127   Median : 5.108   Median : 5.036   Median : 5.085  
##  Mean   : 5.542   Mean   : 5.543   Mean   : 5.506   Mean   : 5.523  
##  3rd Qu.: 6.907   3rd Qu.: 6.886   3rd Qu.: 6.954   3rd Qu.: 6.875  
##  Max.   :14.403   Max.   :14.474   Max.   :14.516   Max.   :14.497  
##  GSM330532.CEL    GSM330546.CEL    GSM330559.CEL    GSM330566.CEL   
##  Min.   : 2.549   Min.   : 2.624   Min.   : 2.459   Min.   : 2.592  
##  1st Qu.: 3.867   1st Qu.: 3.905   1st Qu.: 3.735   1st Qu.: 4.047  
##  Median : 5.183   Median : 5.132   Median : 5.077   Median : 5.272  
##  Mean   : 5.557   Mean   : 5.531   Mean   : 5.551   Mean   : 5.534  
##  3rd Qu.: 6.923   3rd Qu.: 6.801   3rd Qu.: 6.979   3rd Qu.: 6.690  
##  Max.   :14.407   Max.   :14.384   Max.   :14.403   Max.   :14.234  
##  GSM330571.CEL    GSM330574.CEL    GSM330580.CEL    GSM330584.CEL   
##  Min.   : 2.539   Min.   : 2.560   Min.   : 2.618   Min.   : 2.595  
##  1st Qu.: 3.823   1st Qu.: 3.777   1st Qu.: 4.090   1st Qu.: 3.725  
##  Median : 5.091   Median : 5.037   Median : 5.185   Median : 5.056  
##  Mean   : 5.553   Mean   : 5.510   Mean   : 5.566   Mean   : 5.531  
##  3rd Qu.: 6.866   3rd Qu.: 6.887   3rd Qu.: 6.650   3rd Qu.: 7.020  
##  Max.   :14.556   Max.   :14.363   Max.   :14.500   Max.   :14.512  
##  GSM330593.CEL    GSM330603.CEL    GSM330611.CEL    GSM330612.CEL   
##  Min.   : 2.496   Min.   : 2.530   Min.   : 2.433   Min.   : 2.567  
##  1st Qu.: 3.945   1st Qu.: 3.746   1st Qu.: 4.022   1st Qu.: 3.951  
##  Median : 5.153   Median : 5.065   Median : 5.164   Median : 5.142  
##  Mean   : 5.536   Mean   : 5.558   Mean   : 5.582   Mean   : 5.553  
##  3rd Qu.: 6.744   3rd Qu.: 7.006   3rd Qu.: 6.696   3rd Qu.: 6.776  
##  Max.   :14.347   Max.   :14.387   Max.   :14.490   Max.   :14.352  
##  GSM330933.CEL    GSM330934.CEL    GSM330969.CEL    GSM330979.CEL   
##  Min.   : 2.539   Min.   : 2.472   Min.   : 2.613   Min.   : 2.533  
##  1st Qu.: 3.786   1st Qu.: 3.967   1st Qu.: 4.076   1st Qu.: 3.746  
##  Median : 5.040   Median : 5.147   Median : 5.222   Median : 5.036  
##  Mean   : 5.530   Mean   : 5.545   Mean   : 5.569   Mean   : 5.516  
##  3rd Qu.: 6.931   3rd Qu.: 6.727   3rd Qu.: 6.633   3rd Qu.: 6.945  
##  Max.   :14.551   Max.   :14.582   Max.   :14.719   Max.   :14.586  
##  GSM330980.CEL    GSM330982.CEL    GSM330987.CEL    GSM330999.CEL
```

```
##  Min.   : 2.587   Min.   : 2.598   Min.   : 2.502   Min.   : 2.576
##  1st Qu.: 3.790   1st Qu.: 3.908   1st Qu.: 3.901   1st Qu.: 3.839
##  Median : 5.055   Median : 5.153   Median : 5.103   Median : 5.095
##  Mean   : 5.527   Mean   : 5.550   Mean   : 5.515   Mean   : 5.526
##  3rd Qu.: 6.870   3rd Qu.: 6.812   3rd Qu.: 6.752   3rd Qu.: 6.839
##  Max.   :14.621   Max.   :14.586   Max.   :14.485   Max.   :14.621
##  GSM331004.CEL    GSM331009.CEL    GSM331037.CEL    GSM331048.CEL
##  Min.   : 2.503   Min.   : 2.485   Min.   : 2.598   Min.   : 2.516
##  1st Qu.: 3.740   1st Qu.: 3.910   1st Qu.: 3.769   1st Qu.: 3.862
##  Median : 5.014   Median : 5.099   Median : 5.042   Median : 5.056
##  Mean   : 5.518   Mean   : 5.533   Mean   : 5.520   Mean   : 5.534
##  3rd Qu.: 6.948   3rd Qu.: 6.730   3rd Qu.: 6.896   3rd Qu.: 6.843
##  Max.   :14.691   Max.   :14.742   Max.   :14.689   Max.   :14.520
##  GSM331377.CEL    GSM331378.CEL    GSM331381.CEL    GSM331382.CEL
##  Min.   : 2.528   Min.   : 2.531   Min.   : 2.440   Min.   : 2.557
##  1st Qu.: 3.898   1st Qu.: 3.885   1st Qu.: 3.911   1st Qu.: 3.902
##  Median : 5.122   Median : 5.136   Median : 5.096   Median : 5.131
##  Mean   : 5.564   Mean   : 5.556   Mean   : 5.564   Mean   : 5.535
##  3rd Qu.: 6.801   3rd Qu.: 6.805   3rd Qu.: 6.772   3rd Qu.: 6.745
##  Max.   :14.388   Max.   :14.454   Max.   :14.292   Max.   :14.420
##  GSM331383.CEL    GSM331386.CEL    GSM331387.CEL    GSM331388.CEL
##  Min.   : 2.476   Min.   : 2.543   Min.   : 2.524   Min.   : 2.525
##  1st Qu.: 3.888   1st Qu.: 3.832   1st Qu.: 3.886   1st Qu.: 3.924
##  Median : 5.132   Median : 5.126   Median : 5.117   Median : 5.149
##  Mean   : 5.561   Mean   : 5.571   Mean   : 5.562   Mean   : 5.575
##  3rd Qu.: 6.775   3rd Qu.: 6.896   3rd Qu.: 6.795   3rd Qu.: 6.789
##  Max.   :14.366   Max.   :14.334   Max.   :14.392   Max.   :14.412
##  GSM331389.CEL    GSM331390.CEL    GSM331392.CEL    GSM331393.CEL
##  Min.   : 2.574   Min.   : 2.480   Min.   : 2.639   Min.   : 2.565
##  1st Qu.: 4.046   1st Qu.: 3.776   1st Qu.: 4.257   1st Qu.: 3.880
##  Median : 5.208   Median : 5.060   Median : 5.286   Median : 5.138
##  Mean   : 5.573   Mean   : 5.543   Mean   : 5.577   Mean   : 5.560
##  3rd Qu.: 6.636   3rd Qu.: 6.908   3rd Qu.: 6.495   3rd Qu.: 6.790
##  Max.   :14.482   Max.   :14.424   Max.   :14.552   Max.   :14.383
##  GSM331660.CEL    GSM331661.CEL    GSM331663.CEL    GSM331666.CEL
##  Min.   : 2.491   Min.   : 2.527   Min.   : 2.478   Min.   : 2.616
##  1st Qu.: 3.821   1st Qu.: 3.755   1st Qu.: 3.790   1st Qu.: 3.837
##  Median : 5.092   Median : 5.121   Median : 5.107   Median : 5.107
##  Mean   : 5.537   Mean   : 5.551   Mean   : 5.543   Mean   : 5.539
##  3rd Qu.: 6.840   3rd Qu.: 6.969   3rd Qu.: 6.888   3rd Qu.: 6.820
##  Max.   :14.362   Max.   :14.475   Max.   :14.423   Max.   :14.500
##  GSM331668.CEL    GSM331670.CEL    GSM331671.CEL    GSM331672.CEL
##  Min.   : 2.453   Min.   : 2.485   Min.   : 2.525   Min.   : 2.555
##  1st Qu.: 3.930   1st Qu.: 3.748   1st Qu.: 3.807   1st Qu.: 3.777
##  Median : 5.189   Median : 5.105   Median : 5.090   Median : 5.130
##  Mean   : 5.588   Mean   : 5.533   Mean   : 5.553   Mean   : 5.551
##  3rd Qu.: 6.822   3rd Qu.: 6.957   3rd Qu.: 6.888   3rd Qu.: 6.939
##  Max.   :14.489   Max.   :14.386   Max.   :14.407   Max.   :14.390
##  GSM331673.CEL    GSM331674.CEL    GSM331675.CEL    GSM331677.CEL
##  Min.   : 2.491   Min.   : 2.461   Min.   : 2.539   Min.   : 2.510
```

```
##  1st Qu.: 3.884   1st Qu.: 3.850   1st Qu.: 3.734   1st Qu.: 3.763
##  Median : 5.162   Median : 5.123   Median : 5.136   Median : 5.085
##  Mean   : 5.540   Mean   : 5.555   Mean   : 5.535   Mean   : 5.545
##  3rd Qu.: 6.792   3rd Qu.: 6.817   3rd Qu.: 6.946   3rd Qu.: 6.945
##  Max.   :14.482   Max.   :14.458   Max.   :14.384   Max.   :14.421
```

```
length(rownames(x))
```

```
## [1] 20172
```

```
# Number of genes profiled
```

```
types <- leukemiasEset$LeukemiaType
summary(types)
```

```
## ALL AML CLL CML NoL
##  12  12  12  12  12
```

```
# How many patients in each type of leukemia group
```

```
new_mat <- x[,leukemiasEset$LeukemiaType=='AML' | leukemiasEset$Leukemi
aType=='NoL']
dim(new_mat)
```

```
## [1] 20172    24
```

```
# Check that we have the right data set to work with
```

```
summary(new_mat)
```

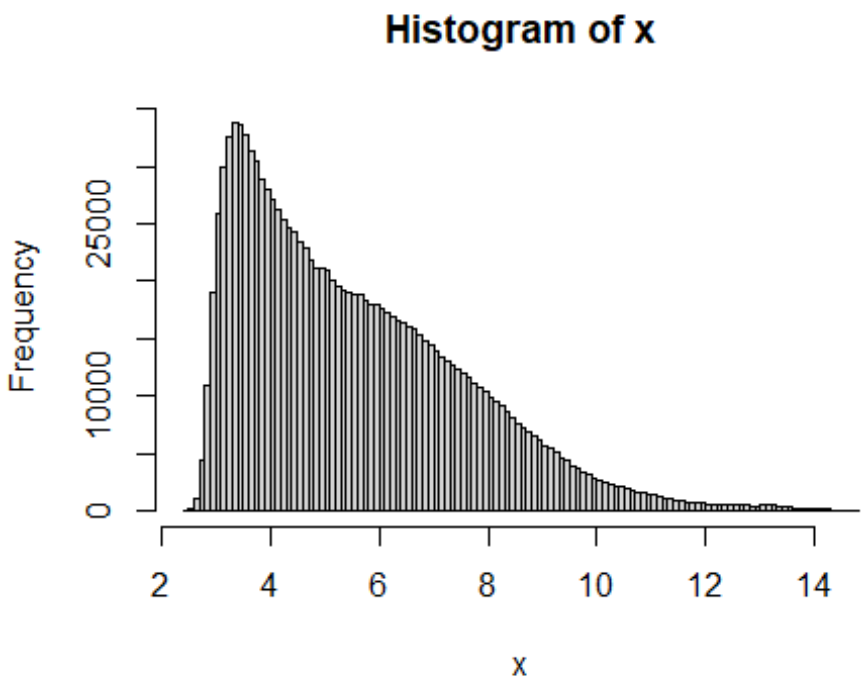
```
##  GSM330532.CEL    GSM330546.CEL    GSM330559.CEL    GSM330566.CEL
##  Min.   : 2.549   Min.   : 2.624   Min.   : 2.459   Min.   : 2.592
##  1st Qu.: 3.867   1st Qu.: 3.905   1st Qu.: 3.735   1st Qu.: 4.047
##  Median : 5.183   Median : 5.132   Median : 5.077   Median : 5.272
##  Mean   : 5.557   Mean   : 5.531   Mean   : 5.551   Mean   : 5.534
##  3rd Qu.: 6.923   3rd Qu.: 6.801   3rd Qu.: 6.979   3rd Qu.: 6.690
##  Max.   :14.407   Max.   :14.384   Max.   :14.403   Max.   :14.234
##  GSM330571.CEL    GSM330574.CEL    GSM330580.CEL    GSM330584.CEL
##  Min.   : 2.539   Min.   : 2.560   Min.   : 2.618   Min.   : 2.595
##  1st Qu.: 3.823   1st Qu.: 3.777   1st Qu.: 4.090   1st Qu.: 3.725
##  Median : 5.091   Median : 5.037   Median : 5.185   Median : 5.056
##  Mean   : 5.553   Mean   : 5.510   Mean   : 5.566   Mean   : 5.531
##  3rd Qu.: 6.866   3rd Qu.: 6.887   3rd Qu.: 6.650   3rd Qu.: 7.020
##  Max.   :14.556   Max.   :14.363   Max.   :14.500   Max.   :14.512
##  GSM330593.CEL    GSM330603.CEL    GSM330611.CEL    GSM330612.CEL
##  Min.   : 2.496   Min.   : 2.530   Min.   : 2.433   Min.   : 2.567
##  1st Qu.: 3.945   1st Qu.: 3.746   1st Qu.: 4.022   1st Qu.: 3.951
##  Median : 5.153   Median : 5.065   Median : 5.164   Median : 5.142
##  Mean   : 5.536   Mean   : 5.558   Mean   : 5.582   Mean   : 5.553
##  3rd Qu.: 6.744   3rd Qu.: 7.006   3rd Qu.: 6.696   3rd Qu.: 6.776
##  Max.   :14.347   Max.   :14.387   Max.   :14.490   Max.   :14.352
```

```
##  GSM331660.CEL    GSM331661.CEL    GSM331663.CEL    GSM331666.CEL
##  Min.   : 2.491   Min.   : 2.527   Min.   : 2.478   Min.   : 2.616
##  1st Qu.: 3.821   1st Qu.: 3.755   1st Qu.: 3.790   1st Qu.: 3.837
##  Median : 5.092   Median : 5.121   Median : 5.107   Median : 5.107
##  Mean   : 5.537   Mean   : 5.551   Mean   : 5.543   Mean   : 5.539
##  3rd Qu.: 6.840   3rd Qu.: 6.969   3rd Qu.: 6.888   3rd Qu.: 6.820
##  Max.   :14.362   Max.   :14.475   Max.   :14.423   Max.   :14.500
##  GSM331668.CEL    GSM331670.CEL    GSM331671.CEL    GSM331672.CEL
##  Min.   : 2.453   Min.   : 2.485   Min.   : 2.525   Min.   : 2.555
##  1st Qu.: 3.930   1st Qu.: 3.748   1st Qu.: 3.807   1st Qu.: 3.777
##  Median : 5.189   Median : 5.105   Median : 5.090   Median : 5.130
##  Mean   : 5.588   Mean   : 5.533   Mean   : 5.553   Mean   : 5.551
##  3rd Qu.: 6.822   3rd Qu.: 6.957   3rd Qu.: 6.888   3rd Qu.: 6.939
##  Max.   :14.489   Max.   :14.386   Max.   :14.407   Max.   :14.390
##  GSM331673.CEL    GSM331674.CEL    GSM331675.CEL    GSM331677.CEL
##  Min.   : 2.491   Min.   : 2.461   Min.   : 2.539   Min.   : 2.510
##  1st Qu.: 3.884   1st Qu.: 3.850   1st Qu.: 3.734   1st Qu.: 3.763
##  Median : 5.162   Median : 5.123   Median : 5.136   Median : 5.085
##  Mean   : 5.540   Mean   : 5.555   Mean   : 5.535   Mean   : 5.545
##  3rd Qu.: 6.792   3rd Qu.: 6.817   3rd Qu.: 6.946   3rd Qu.: 6.945
##  Max.   :14.482   Max.   :14.458   Max.   :14.384   Max.   :14.421
```

```
# Summary statistics

which(is.na(new_mat))

## integer(0)

# No missing values
```

Plot the histogram of the whole expression matrix and check the data distribution

```
hist(x, breaks = 100)
```




```
# We observe it's right skewed!
```

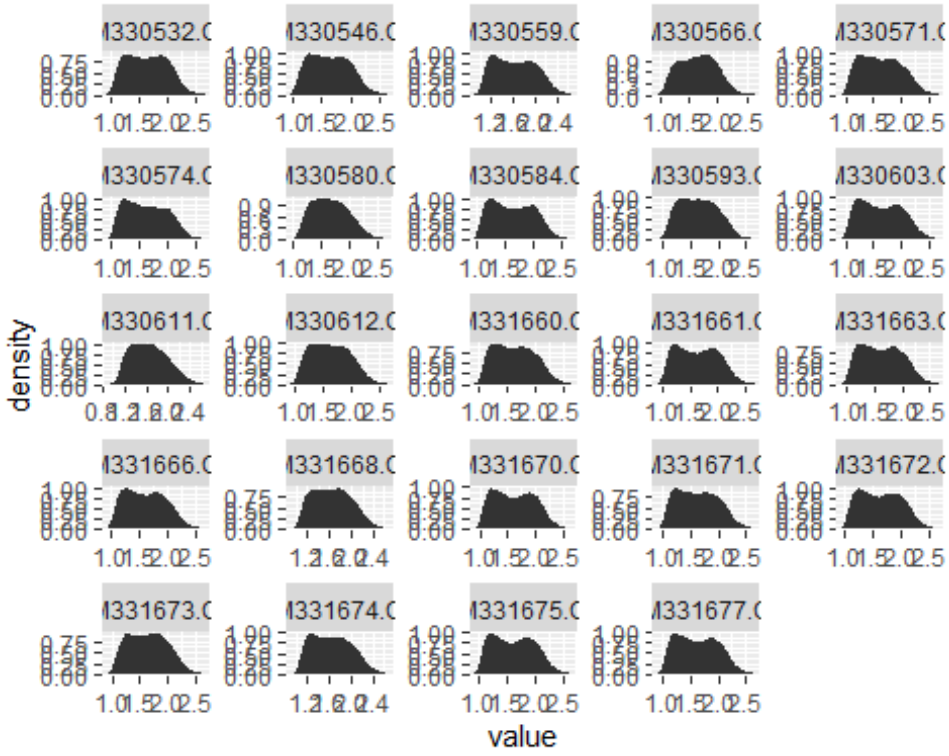
Creating a small multiple chart

```
library(reshape2)
library(ggplot2)
df1 <- melt(as.data.frame(new_mat))

## No id variables; using all as measure variables

ggplot(data = df1, aes(x = value)) +
  stat_density() +
  facet_wrap(~variable, scales = "free")
```


```
# We observe that our data are nowhere near being normal
# Right skewed data so the log transformation should be more appropriat
e
```

Log transformation

```
df2 <- melt(as.data.frame(log(new_mat)))

## No id variables; using all as measure variables

ggplot(data = df2, aes(x = value)) +
  stat_density() +
  facet_wrap(~variable, scales = "free")
```




```
# We observe that the picture now looks closer to normality than before
!
```

Useful summary statistics

```
apply(new_mat,2,range)
```

```
##      GSM330532.CEL GSM330546.CEL GSM330559.CEL GSM330566.CEL GSM3305
71.CEL
## [1,]      2.548731      2.624108      2.459458      2.591882      2.
539496
## [2,]     14.406832     14.383963     14.402716     14.233605     14.
556264
##      GSM330574.CEL GSM330580.CEL GSM330584.CEL GSM330593.CEL GSM3306
03.CEL
## [1,]       2.56021      2.618374       2.59485      2.496146      2.
530432
## [2,]      14.36320     14.500195      14.51158     14.346659     14.
386684
##      GSM330611.CEL GSM330612.CEL GSM331660.CEL GSM331661.CEL GSM3316
63.CEL
## [1,]      2.432825      2.567399      2.491387      2.526767       2
.47845
## [2,]     14.489737     14.352242     14.362445     14.474812      14
.42314
##      GSM331666.CEL GSM331668.CEL GSM331670.CEL GSM331671.CEL GSM3316
72.CEL
## [1,]      2.616153      2.452677      2.485268      2.524846      2.
555388
## [2,]     14.500166     14.488940     14.385826     14.406832     14.
389920
##      GSM331673.CEL GSM331674.CEL GSM331675.CEL GSM331677.CEL
## [1,]       2.49106      2.461213      2.538758      2.510073
## [2,]      14.48225     14.458293     14.383963     14.421144
```

```
#  It's not often used because it's very sensitive to outliers

apply(new_mat,2,IQR)
```

```
## GSM330532.CEL GSM330546.CEL GSM330559.CEL GSM330566.CEL GSM330571.CE
L
##      3.056642      2.896110      3.244112      2.642878      3.04263
6
## GSM330574.CEL GSM330580.CEL GSM330584.CEL GSM330593.CEL GSM330603.CE
L
##      3.109597      2.559956      3.295814      2.798405      3.25960
2
## GSM330611.CEL GSM330612.CEL GSM331660.CEL GSM331661.CEL GSM331663.CE
L
##      2.674320      2.825195      3.019740      3.214763      3.09870
5
## GSM331666.CEL GSM331668.CEL GSM331670.CEL GSM331671.CEL GSM331672.CE
L
##      2.983206      2.891645      3.208607      3.080498      3.16272
```

```
8
## GSM331673.CEL GSM331674.CEL GSM331675.CEL GSM331677.CEL
##      2.907652      2.966984      3.211548      3.182213

# It's pretty robust to outliers. It's used a lot in combination with t
he median.

apply(new_mat,2,median)

## GSM330532.CEL GSM330546.CEL GSM330559.CEL GSM330566.CEL GSM330571.CE
L
##      5.183401      5.131827      5.076911      5.271958      5.09105
8
## GSM330574.CEL GSM330580.CEL GSM330584.CEL GSM330593.CEL GSM330603.CE
L
##      5.036518      5.185305      5.055979      5.152545      5.06511
1
## GSM330611.CEL GSM330612.CEL GSM331660.CEL GSM331661.CEL GSM331663.CE
L
##      5.164260      5.141501      5.092062      5.121016      5.10697
3
## GSM331666.CEL GSM331668.CEL GSM331670.CEL GSM331671.CEL GSM331672.CE
L
##      5.106549      5.188785      5.105193      5.090007      5.12958
4
## GSM331673.CEL GSM331674.CEL GSM331675.CEL GSM331677.CEL
##      5.161670      5.122651      5.136021      5.085279

apply(new_mat,2,mean)

## GSM330532.CEL GSM330546.CEL GSM330559.CEL GSM330566.CEL GSM330571.CE
L
##      5.557318      5.530640      5.550560      5.533707      5.55344
6
## GSM330574.CEL GSM330580.CEL GSM330584.CEL GSM330593.CEL GSM330603.CE
L
##      5.510033      5.566124      5.530860      5.535596      5.55824
4
## GSM330611.CEL GSM330612.CEL GSM331660.CEL GSM331661.CEL GSM331663.CE
L
##      5.582216      5.552828      5.536801      5.551041      5.54297
9
## GSM331666.CEL GSM331668.CEL GSM331670.CEL GSM331671.CEL GSM331672.CE
L
##      5.538984      5.587689      5.532583      5.553264      5.55061
5
## GSM331673.CEL GSM331674.CEL GSM331675.CEL GSM331677.CEL
##      5.540235      5.554737      5.534797      5.544794
```

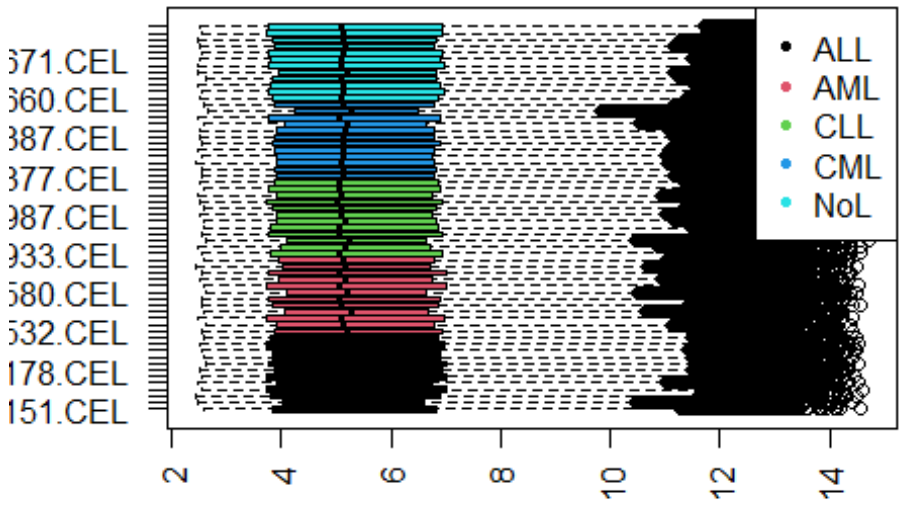
Checking the correlations

```
library(Hmisc)

## Loading required package: lattice

## Loading required package: survival

## Loading required package: Formula

##
## Attaching package: 'Hmisc'

## The following object is masked from 'package:Biobase':
##
##     contents

## The following objects are masked from 'package:base':
##
##     format.pval, units

mat <- as.matrix(new_mat)
newmat <- mat[,1:5]
rcorr(newmat)

##               GSM330532.CEL GSM330546.CEL GSM330559.CEL GSM330566.CE
L
## GSM330532.CEL          1.00          0.92          0.92          0.9
0
## GSM330546.CEL          0.92          1.00          0.94          0.8
8
## GSM330559.CEL          0.92          0.94          1.00          0.8
7
## GSM330566.CEL          0.90          0.88          0.87          1.0
0
## GSM330571.CEL          0.91          0.93          0.96          0.8
7
##               GSM330571.CEL
## GSM330532.CEL          0.91
## GSM330546.CEL          0.93
## GSM330559.CEL          0.96
## GSM330566.CEL          0.87
## GSM330571.CEL          1.00
##
## n= 20172
##
##
## P
##               GSM330532.CEL GSM330546.CEL GSM330559.CEL GSM330566.CE
L
## GSM330532.CEL                0             0             0
## GSM330546.CEL  0                            0             0
```

```
## GSM330559.CEL  0             0                          0
## GSM330566.CEL  0             0             0
## GSM330571.CEL  0             0             0            0
##                GSM330571.CEL
## GSM330532.CEL  0
## GSM330546.CEL  0
## GSM330559.CEL  0
## GSM330566.CEL  0
## GSM330571.CEL

# Really strong correlations observed
```

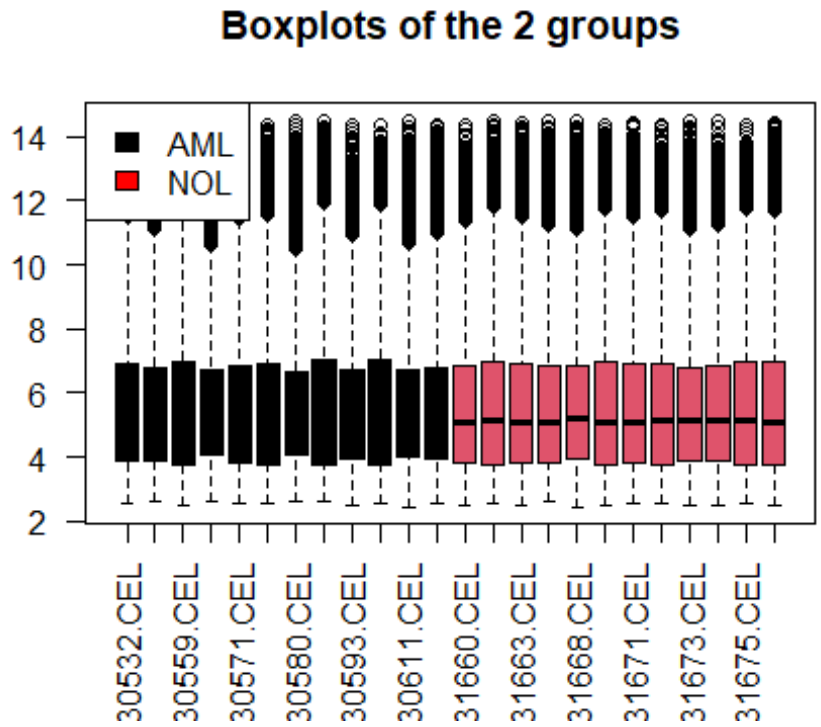
Examine data distributions for all individual patients using colored boxplots

```
boxplot(x, horizontal = T, las = 2, col = leukemiasEset$LeukemiaType)
legend("topright", col = unique(as.numeric(leukemiasEset$LeukemiaType))
, legend = unique(leukemiasEset$LeukemiaType),
pch = 20)
```



Boxplots of the 2 groups of people (those with AML and the control group)

```
boxplot(new_mat, las = 2, col = rep(c(1,2), each = 12), main = "Boxplot
s of the 2 groups")
legend('topleft',legend = c('AML','NOL'),fill = c('black','red'))
```

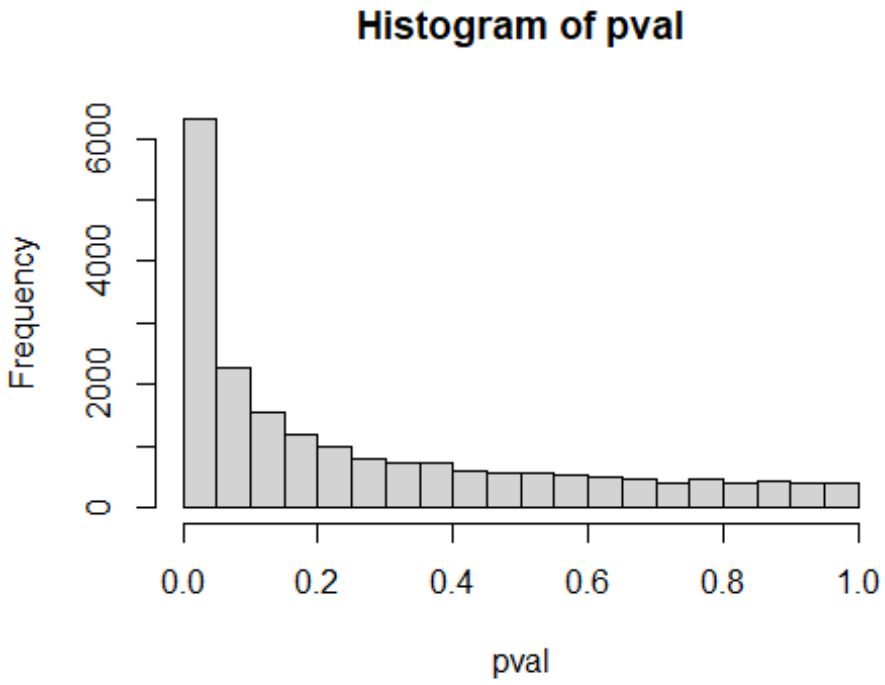


We can't observe if there's a difference from this plot! We have to dig deeper.

Multiple means testing for the 2 groups (people with AML and healthy ones)

Histogram of the p_values

```
hist(pval)
```



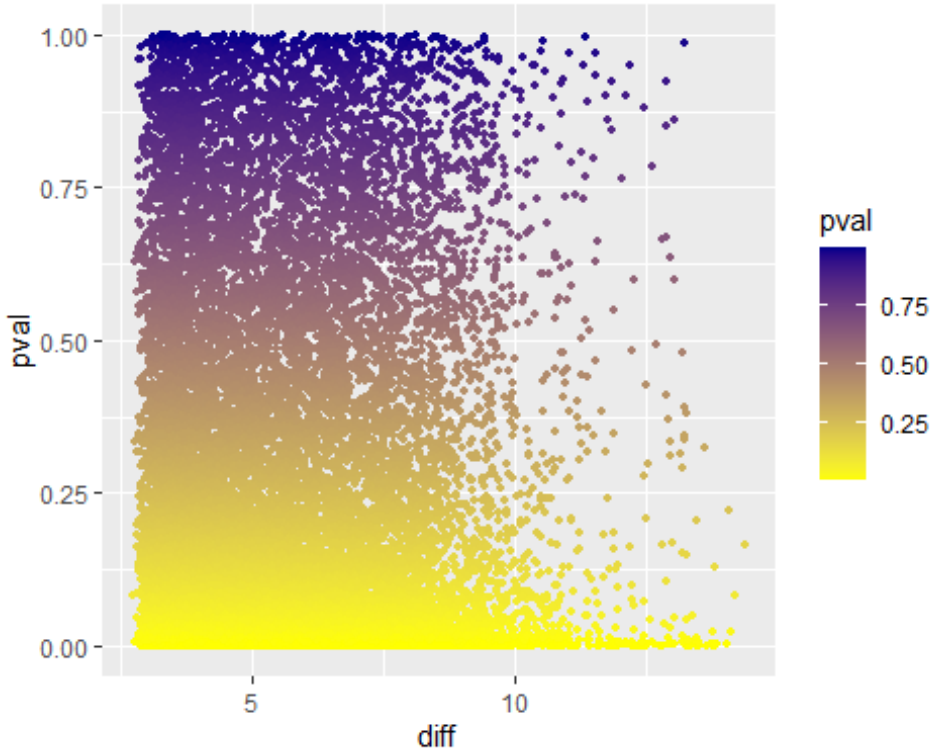

```
# As it was expected to be!
```

Create a data frame that has the estimated means and their corresponding p_values

```
dat <- data.frame(diff,pval)
important_dat <- dat[dat$pval < 0.05,]
```

Plot the differences of the means of the 2 groups with their corresponding p_values

```
ggplot(dat, aes(diff, pval)) +
  geom_point(aes(color = pval), size = 1) +
  scale_color_gradient(low = "yellow", high = "darkblue")
```




```
# We observe that the darker the color the less the p_value hence when
the p_values are for example less than 0.05 we reject H0 => there's a
difference in the means of the 2 groups
```

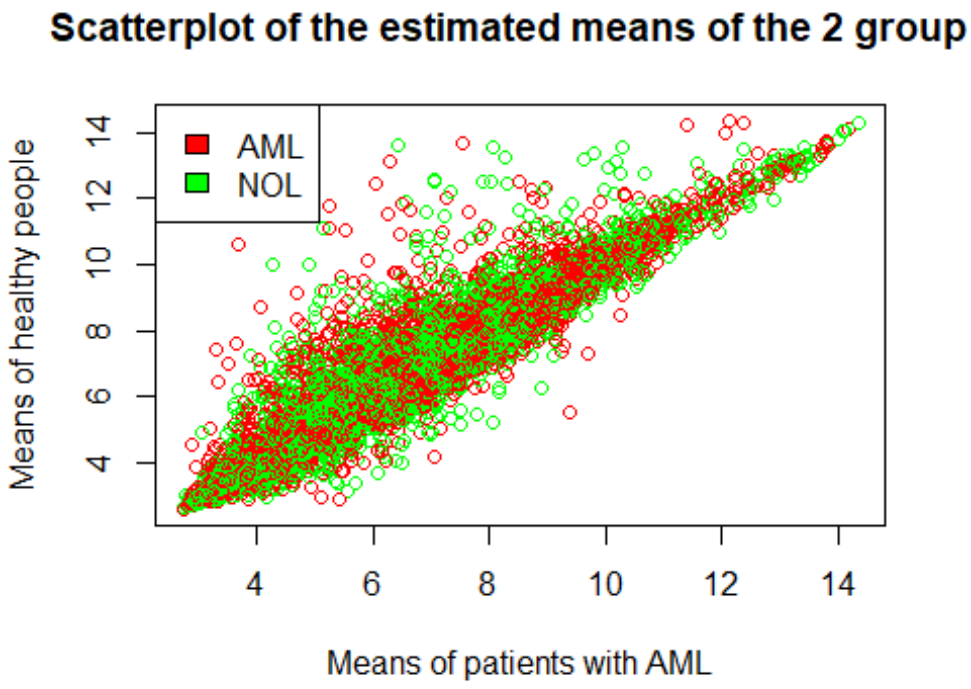
This is a quite informative plot!

Estimating means for each group

```
n <- dim(new_mat)[1]
m1 <- numeric(n)
m2 <- numeric(n)
for (i in 1:n) {
  m1[i] <- mean(new_mat[i,1:12])
  m2[i] <- mean(new_mat[i,13:24])
}
```

Plotting the means for each group

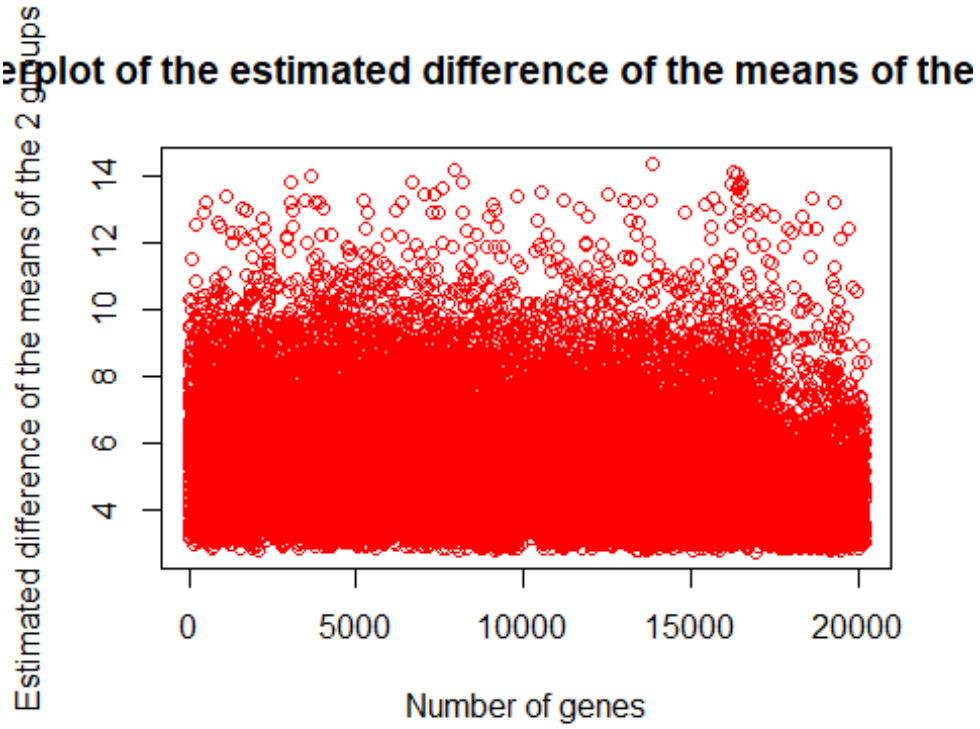
```
plot(m1,m2, col=c('red','green'), main='Scatterplot of the estimated me
ans of the 2 groups', xlab='Means of patients with AML',ylab='Means of
healthy people')
legend('topleft',legend = c('AML','NOL'),fill = c('red','green'))
```



We observe that there's indeed difference in the means of the 2 groups. Should there not exist, they would be observed the one overlapping the other in a straight line!

Plotting the difference of the means of the 2 groups

```
plot(diff, col=c('red'),main='Scatterplot of the estimated difference o
f the means of the 2 groups',
xlab='Number of genes',ylab='Estimated difference of the means of the 2
groups')
```



We can observe the estimated difference of the means of the 2 groups for each row of our data matrix (20172 genes-20172 rows).

We can observe that there seems to be a difference as the estimated differences are not concentrated evenly!

Question 2

PCA

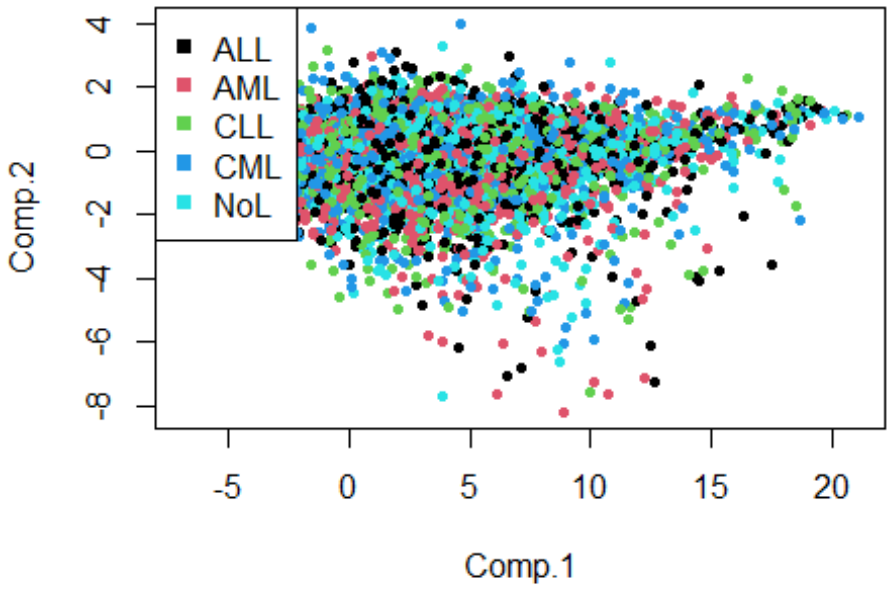
```
set.pr <- princomp(scale(new_mat))
summary(set.pr)
```

```
## Importance of components:
##                            Comp.1     Comp.2     Comp.3      Comp.4
Comp.5
## Standard deviation     4.7055682 0.72362631 0.51930174 0.418959227 0
.358978163
## Proportion of Variance 0.9226446 0.02181921 0.01123699 0.007313981 0
.005369655
## Cumulative Proportion  0.9226446 0.94446380 0.95570079 0.963014770 0
.968384424
##                              Comp.6      Comp.7      Comp.8      Comp
.9
## Standard deviation     0.327725051 0.312941823 0.304080572 0.2543518
58
## Proportion of Variance 0.004475376 0.004080727 0.003852899 0.0026957
53
## Cumulative Proportion  0.972859801 0.976940528 0.980793427 0.9834891
80
##                             Comp.10     Comp.11     Comp.12     Comp.1
3
## Standard deviation     0.24429637 0.229732623 0.211157873 0.20334769
4
## Proportion of Variance 0.00248682 0.002199154 0.001857911 0.00172301
4
## Cumulative Proportion  0.98597600 0.988175153 0.990033064 0.99175607
8
##                             Comp.14     Comp.15      Comp.16      Com
p.17
## Standard deviation     0.184907946 0.172850685 0.1539460251 0.144950
8760
## Proportion of Variance 0.001424693 0.001244952 0.0009875231 0.000875
4916
## Cumulative Proportion  0.993180772 0.994425723 0.9954132464 0.996288
7380
##                             Comp.18      Comp.19      Comp.20      C
omp.21
## Standard deviation     0.1340399307 0.1270538730 0.1221539066 0.1165
177028
## Proportion of Variance 0.0007486497 0.0006726453 0.0006217632 0.0005
657103
## Cumulative Proportion  0.9970373877 0.9977100330 0.9983317962 0.9988
975065
##                             Comp.22      Comp.23     Comp.24
## Standard deviation     0.1028464307 0.0959831286 0.081660164
```

```
## Proportion of Variance 0.0004407464 0.0003838841 0.000277863
## Cumulative Proportion  0.9993382529 0.9997221370 1.000000000

# The 1st PC explains the 92.2% of the total variability
# The first 2 PCs explain the 94.4% of the total variability
# The first 3 PCs explain the 95.5% of the total variability
# The first 4 PCs explain the 96% of the total variability
# We observe that the standard deviations with value more than 1, is th
e first one ONLY
```

Overall plot of all the Leukemia types using the first 2 PCs

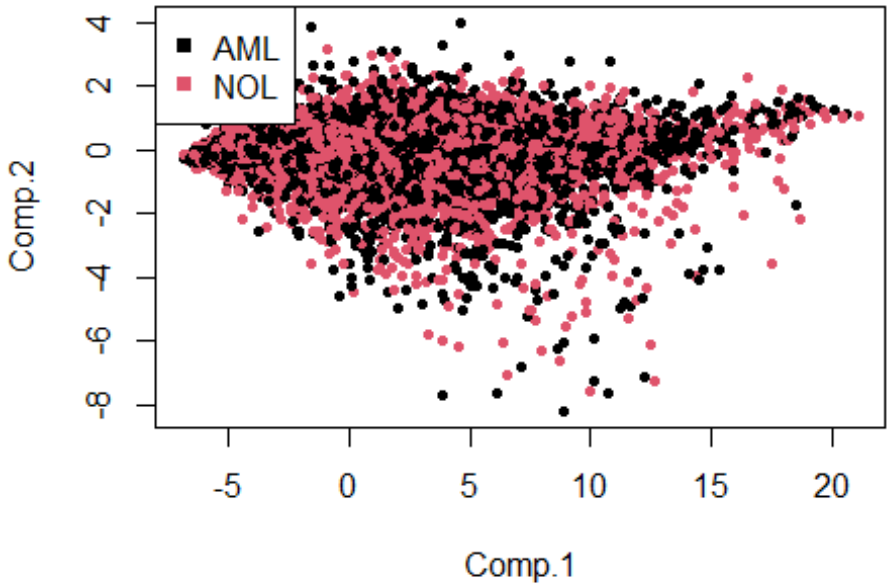
```
plot(set.pr$scores[, 1:2], col = leukemiasEset$LeukemiaType, pch = 20)
legend("topleft", legend = unique(leukemiasEset$LeukemiaType), pch = 15
, col = unique(as.numeric(leukemiasEset$LeukemiaType)))
```



We observe that there's a lot of overlap in the initial data and the 5 groups of patients.

Plot of the 2 Leukemia types we're interested in, using the first 2 PCs

```
factorlvl <- gl(2, 12, labels = c("AML", "NOL"))
plot(set.pr$scores[, 1:2], col = factorlvl  , pch = 20)
legend("topleft", legend = unique(factorlvl), pch = 15, col = unique(fa
ctorlvl))
```

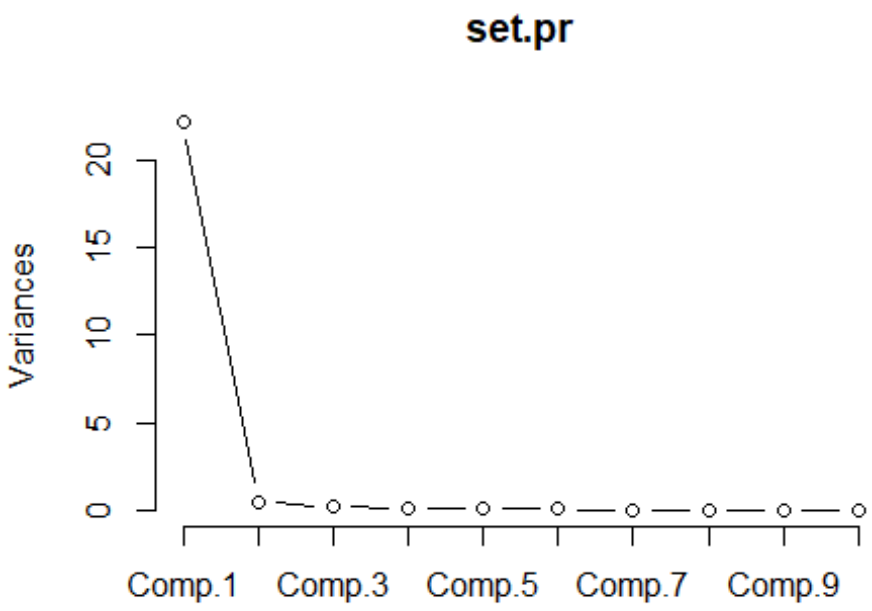


We observe that there's a lot of overlap between the 2 groups!

It's quite difficult to have an obvious partitioning.

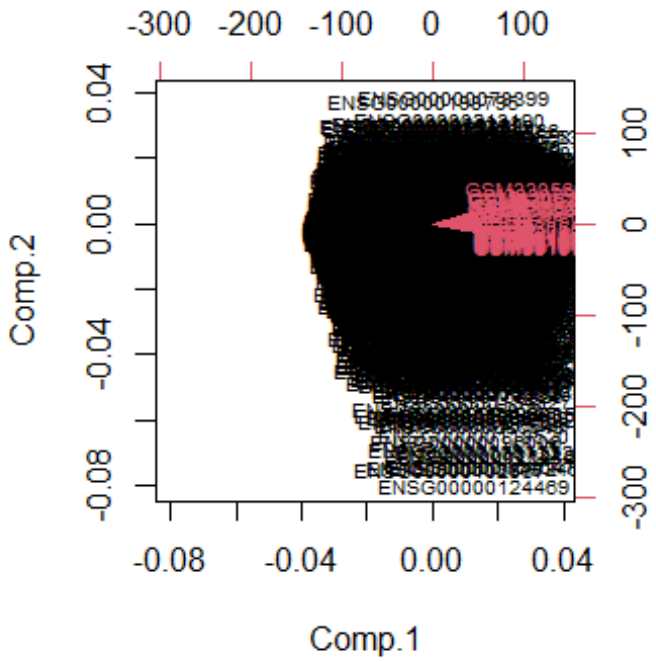
The screeplot indicates to use the first 2 PCs as the greatest angle is located between the first and the second component

```
screeplot(set.pr,type="lines")
```



Biplot offers useful insights about the variables

```
biplot(set.pr, choices=c(1,2) ,cex=.6)
```



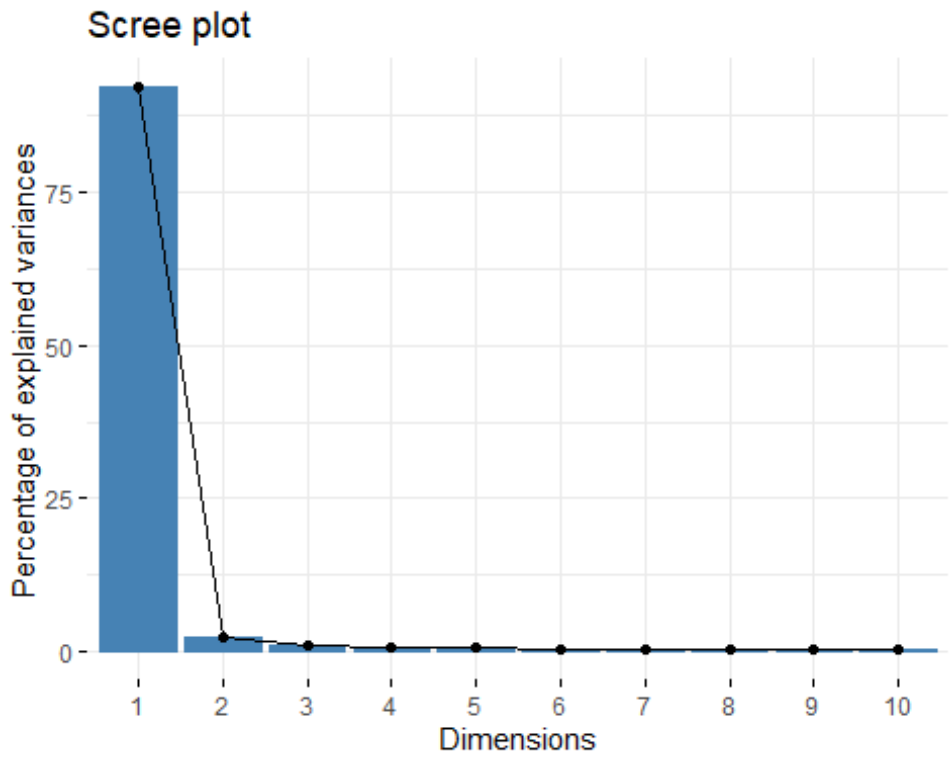
Second way to perform PCA

```
library(factoextra)
# Compute PCA
res.pca <- prcomp(new_mat, scale = TRUE)
summary(res.pca)

## Importance of components:
##                            PC1     PC2     PC3     PC4     PC5     PC
6     PC7
## Standard deviation     4.7057 0.72364 0.51931 0.41897 0.35899 0.3277
3 0.31295
## Proportion of Variance 0.9226 0.02182 0.01124 0.00731 0.00537 0.0044
8 0.00408
## Cumulative Proportion  0.9226 0.94446 0.95570 0.96301 0.96838 0.9728
6 0.97694
##                            PC8    PC9    PC10   PC11    PC12    PC13
PC14
## Standard deviation     0.30409 0.2544 0.24430 0.2297 0.21116 0.20335
0.18491
## Proportion of Variance 0.00385 0.0027 0.00249 0.0022 0.00186 0.00172
0.00142
## Cumulative Proportion  0.98079 0.9835 0.98598 0.9882 0.99003 0.99176
```

```
0.99318
##                            PC15    PC16    PC17    PC18    PC19    PC
20    PC21
## Standard deviation     0.17285 0.15395 0.14495 0.13404 0.12706 0.122
16 0.11652
## Proportion of Variance 0.00124 0.00099 0.00088 0.00075 0.00067 0.000
62 0.00057
## Cumulative Proportion  0.99443 0.99541 0.99629 0.99704 0.99771 0.998
33 0.99890
##                            PC22    PC23    PC24
## Standard deviation     0.10285 0.09599 0.08166
## Proportion of Variance 0.00044 0.00038 0.00028
## Cumulative Proportion  0.99934 0.99972 1.00000
```

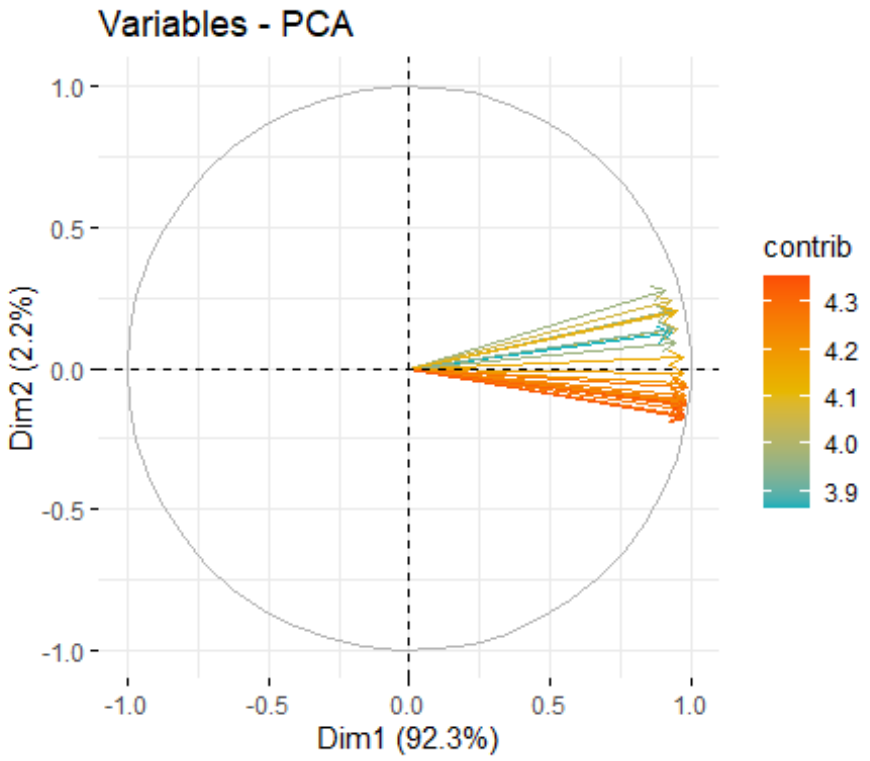
Visualizations

```
fviz_eig(res.pca)
```

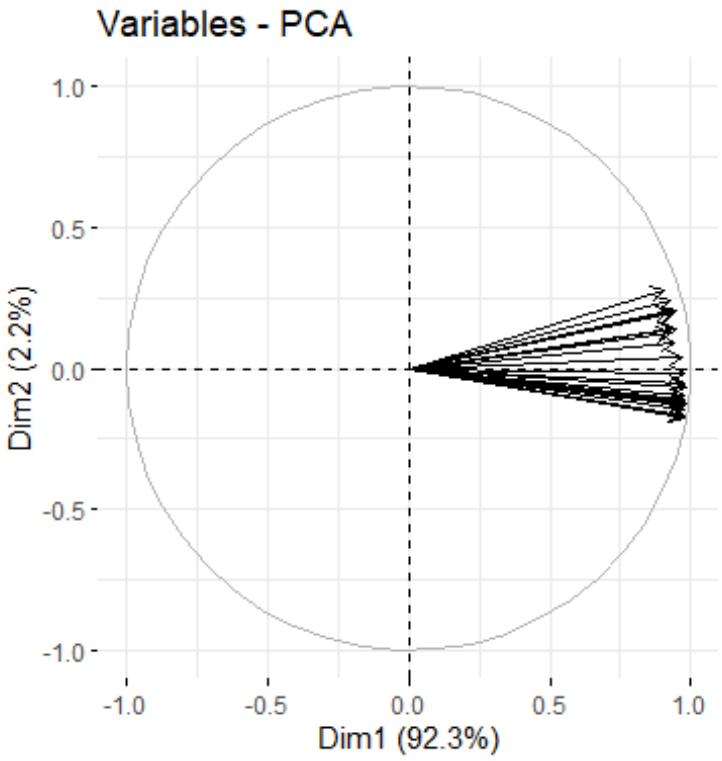


We would keep 2 Pcs as between those 2 there's the greatest angle observed!

```r
# Visualizing the eigenvalues (scree plot). Shows the percentage of var
iances explained by each principal component.


# Graph of variables. Positive correlated variables point to the same s
ide of the plot. Negative correlated variables point to opposite sides
of the graph.
fviz_pca_var(res.pca,
             col.var = "contrib", # Color by contributions to the PC
             gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
             repel = TRUE     # Avoid text overlapping
)
```



```
fviz_pca_var(res.pca,
             repel = TRUE     # Avoid text overlapping
)
```



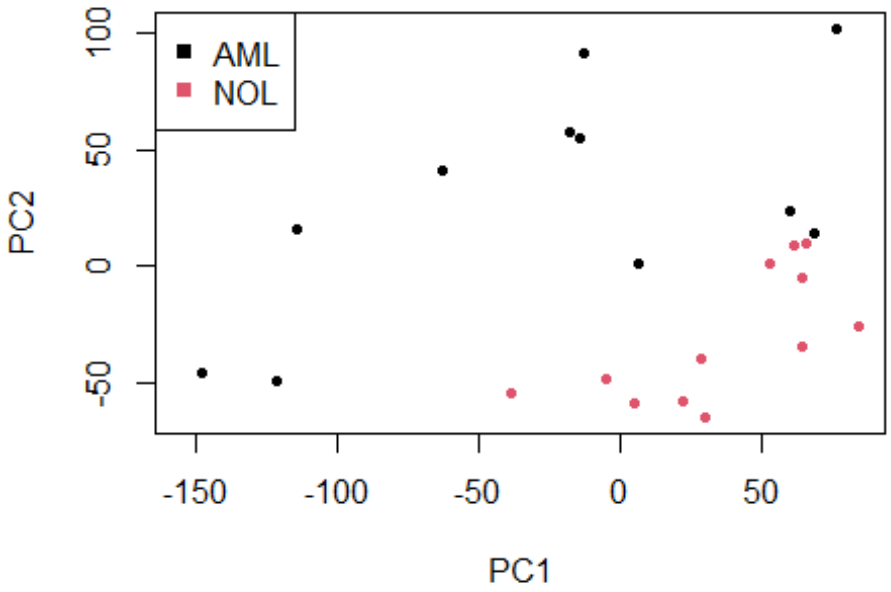
PCA on the transposed matrix of the new data

```
dim(t(new_mat))

## [1]    24 20172

# Check that we got the correct data

t_set.pr <- prcomp(scale(t(new_mat)))
summary(t_set.pr)

## Importance of components:
##                             PC1     PC2      PC3      PC4      PC5
PC6
## Standard deviation     73.0464 49.2185 39.78212 38.52811 34.55513 30
.72136
## Proportion of Variance  0.2645  0.1201  0.07846  0.07359  0.05919  0
.04679
## Cumulative Proportion   0.2645  0.3846  0.46306  0.53665  0.59584  0
.64263
##                             PC7      PC8      PC9     PC10     PC11
PC12
## Standard deviation     30.65738 26.71738 26.21590 24.67480 23.66234
23.29927
```

```
## Proportion of Variance  0.04659  0.03539  0.03407  0.03018  0.02776
0.02691
## Cumulative Proportion   0.68922  0.72461  0.75868  0.78886  0.81662
0.84353
##                            PC13     PC14    PC15     PC16     PC17
PC18
## Standard deviation     21.68917 20.01097 19.7843 17.68876 16.81582 1
6.47620
## Proportion of Variance  0.02332  0.01985  0.0194  0.01551  0.01402
0.01346
## Cumulative Proportion   0.86685  0.88670  0.9061  0.92162  0.93564
0.94909
##                            PC19     PC20     PC21     PC22    PC23
PC24
## Standard deviation     15.5601 15.19749 14.67486 13.67730 12.3037 1.
133e-13
## Proportion of Variance  0.0120  0.01145  0.01068  0.00927  0.0075 0.
000e+00
## Cumulative Proportion   0.9611  0.97255  0.98322  0.99250  1.0000 1.
000e+00

# We observe that we need about 11 PCs to have 81.6% of the total varia
bility explained
```

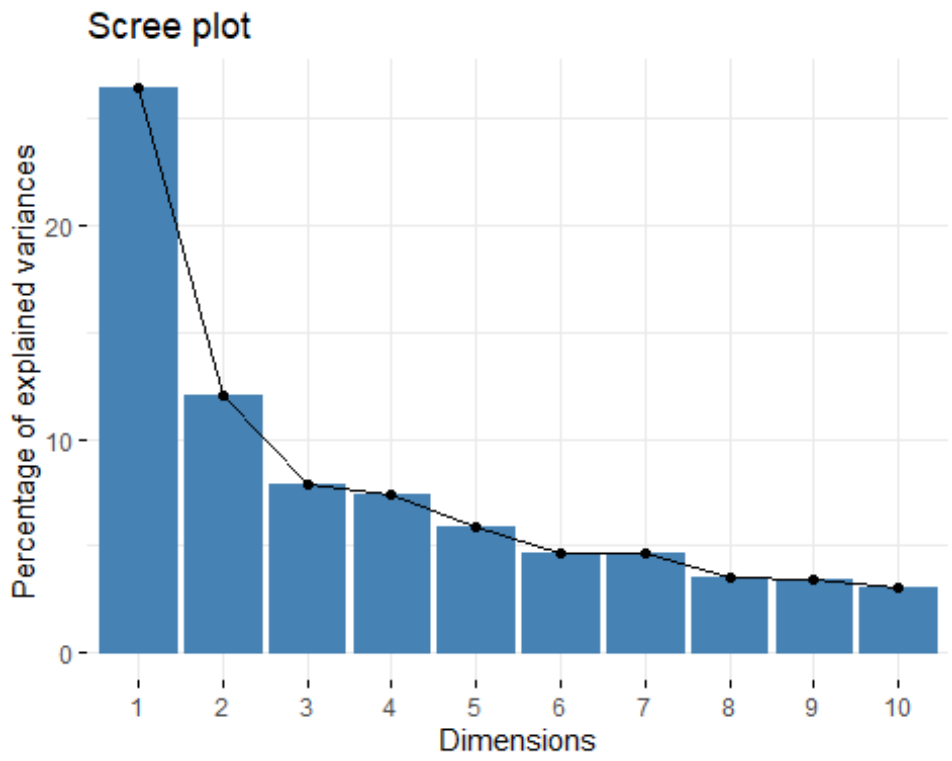
Vizualizations

```
# Plot of the 2 Leukemia types we're interested in, using the first 2 PCs
plot(t_set.pr$x[,1:2] , col = factorlvl  , pch = 20)
legend("topleft", legend = unique(factorlvl), pch = 15, col = unique(factorlvl))
```

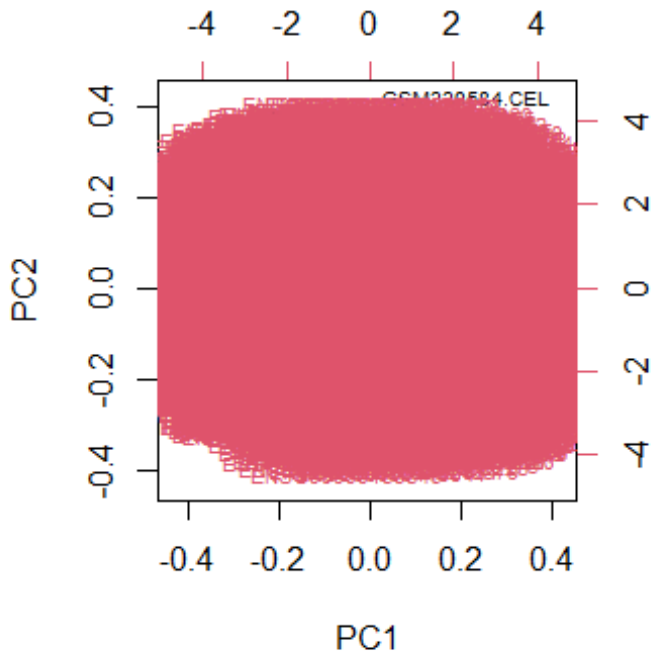


We observe that with the transposed data the partitioning of the 2 groups is much better (there's not as much overlap as before)!

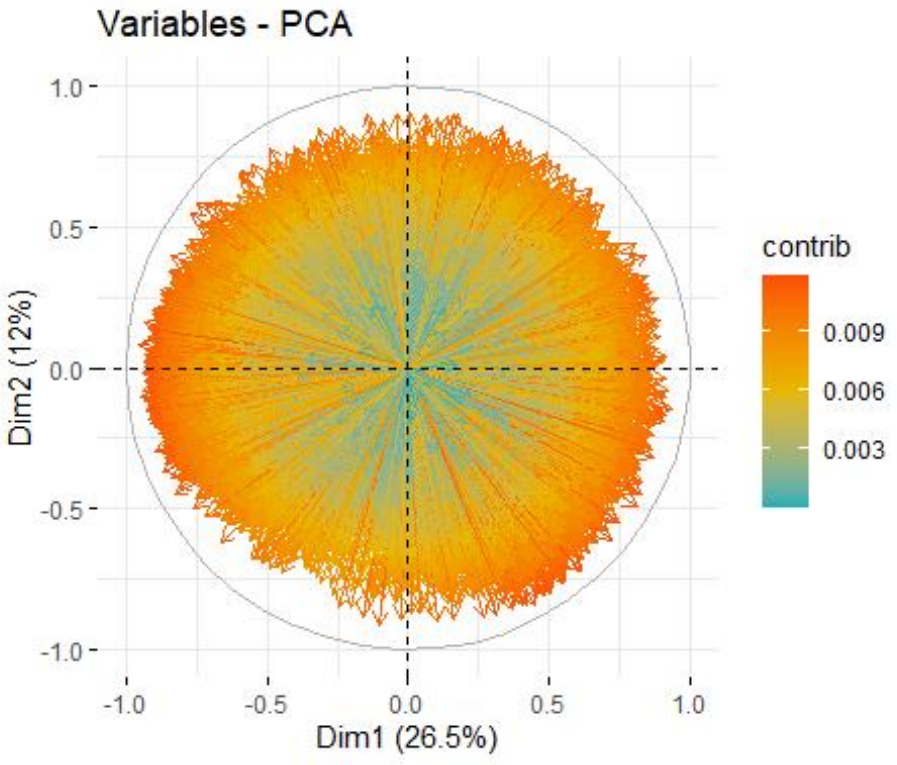
```r
# Visualize eigenvalues (scree plot). Show the percentage of variances explained by each principal component.
fviz_eig(t_set.pr)
```



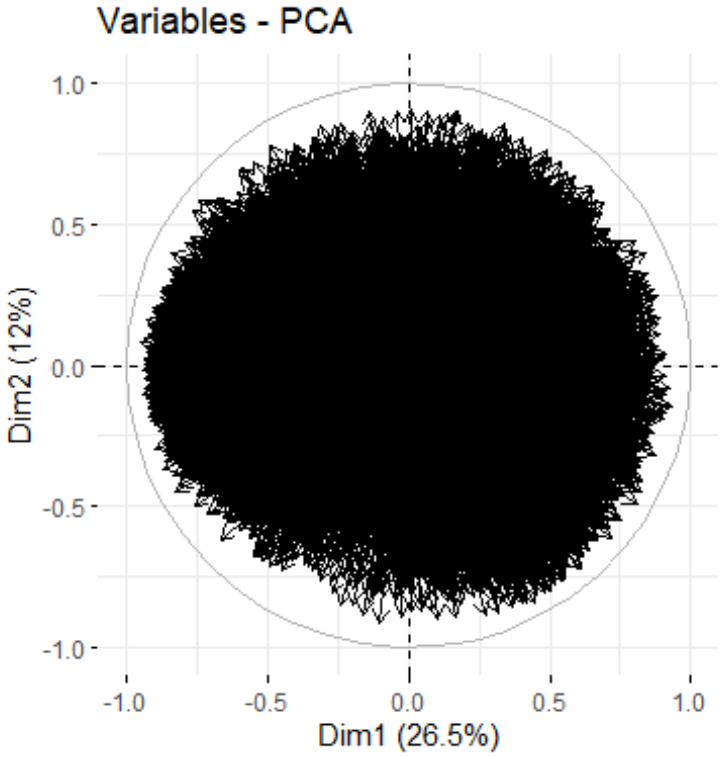

```r
biplot(t_set.pr, choices=c(1,2) ,cex=.6)
```




```
# Biplot of the transposed data now
# Definitely not informative!


# Graph of variables. Positive correlated variables point to the same s
ide of the plot. Negative correlated variables point to opposite sides
of the graph.
fviz_pca_var(t_set.pr,
             col.var = "contrib", # Color by contributions to the PC
             gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
             repel = TRUE     # Avoid text overlapping
)
```



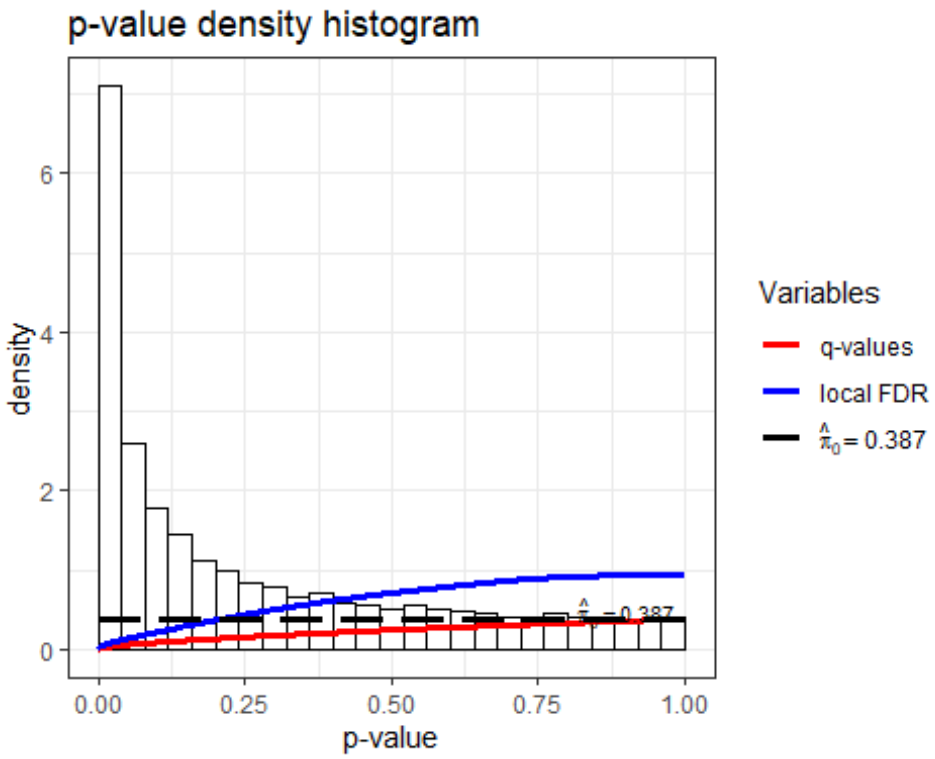

```
fviz_pca_var(t_set.pr,
             repel = TRUE     # Avoid text overlapping
)
```



Question 4

Once we've examined the distribution of the p-values the function qvalue can be used to calculate the q-values:

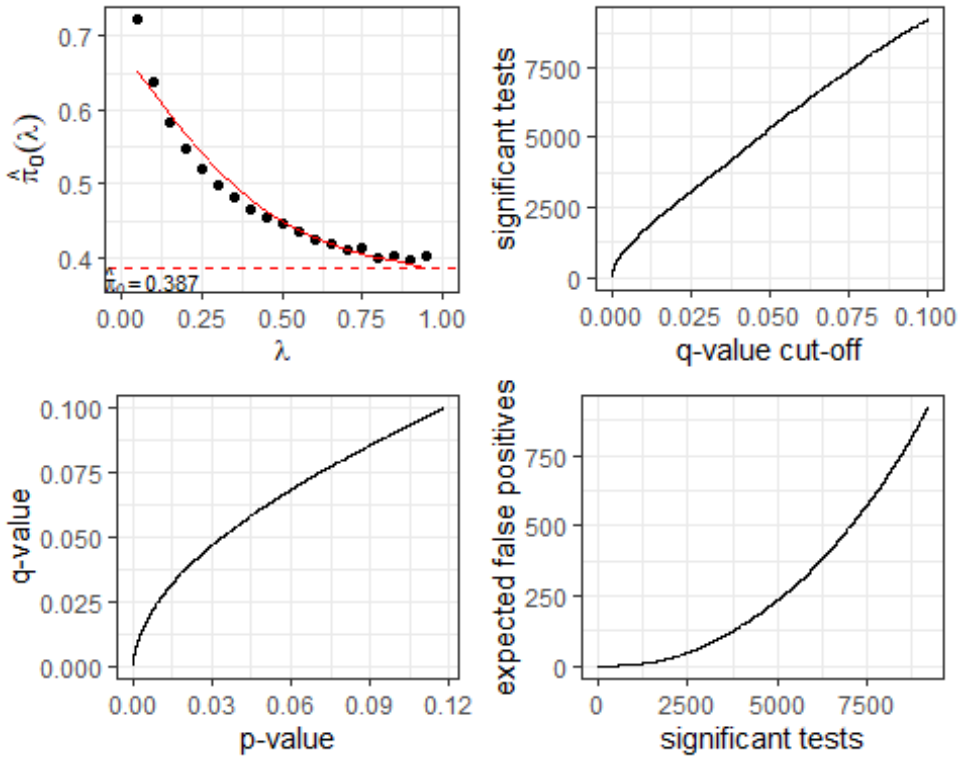
```
library(qvalue)
qobj <- qvalue(pval)
names(qobj)

## [1] "call"       "pi0"        "qvalues"    "pvalues"    "lfdr"
## [6] "pi0.lambda" "lambda"     "pi0.smooth"

# Summarizing results
summary(qobj)

##
## Call:
## qvalue(p = pval)
##
## pi0: 0.3869355
##
## Cumulative number of significant calls:
##
##           <1e-04 <0.001 <0.01 <0.025 <0.05 <0.1    <1
## p-value      559   1240  3078   4625  6325 8603 20172
## q-value      149    465  1687   3050  5334 9207 20172
## local FDR     84    268   920   1669  2706 4843 20172

# One very important statistic that is obtained with the software is an
estimate of the overall proportion of true null hypotheses, π0:
pi0 <- qobj$pi0
pi0

## [1] 0.3869355

# Histogram
hist(qobj)
```




```r
# The q-value is the minimum FDR incurred when calling a test significa
nt. The q-values can be extracted from the qvalue object by:
qvalues <- qobj$qvalues
```

Visualizing results

```
# The hist and plot functions can be used to visualize the results from
qvalue. The function plot allows one to view several useful plots:
# The estimated π0 versus the tuning parameter λ
# The q-values versus the p-values
# The number of significant tests versus each q-value cut-off
# The number of expected false positives versus the number of significa
nt tests
# Applying plot to the qvalue object, we get:
plot(qobj)
```



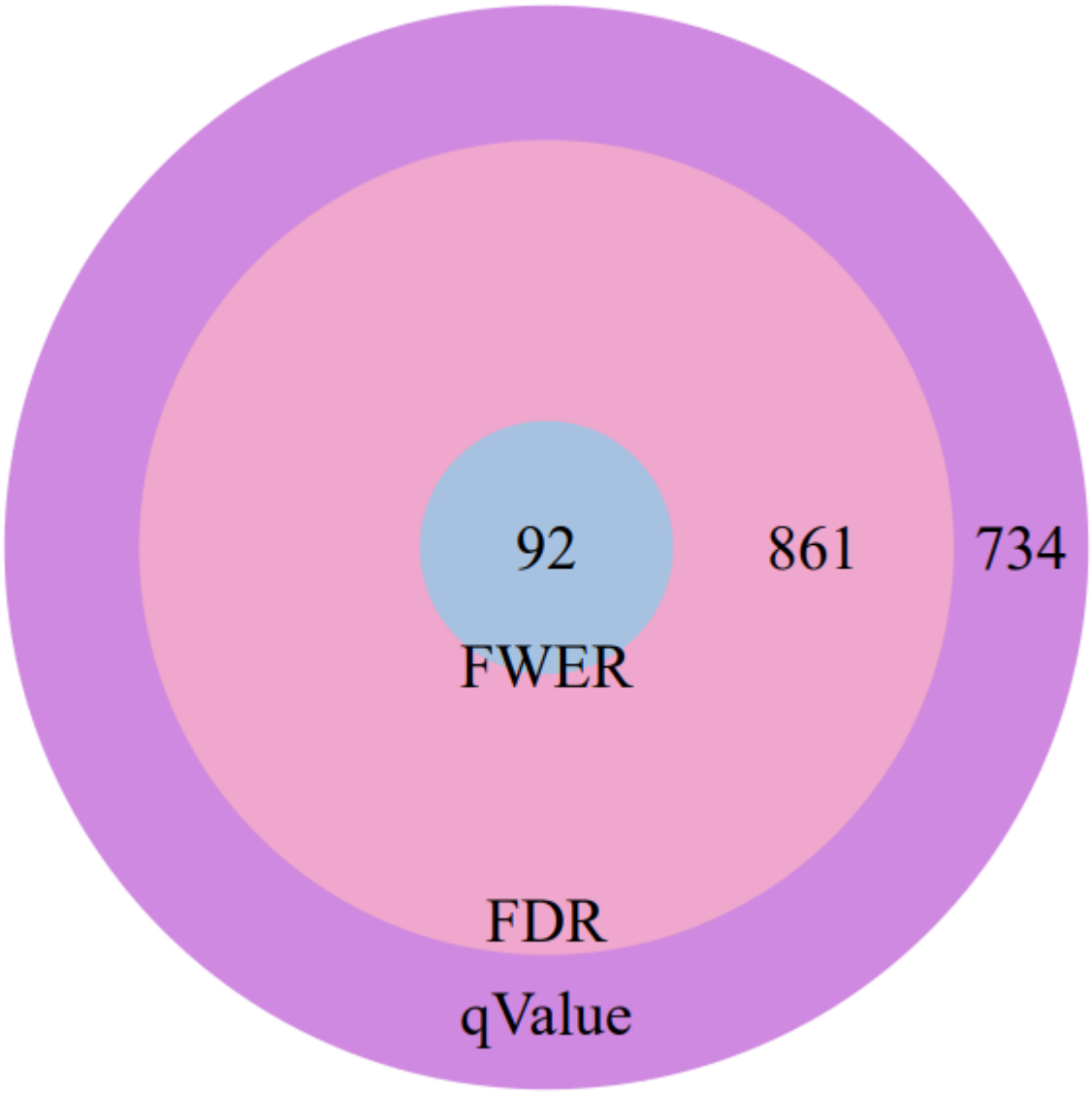
Question 5

```
library(qvalue)
library(VennDiagram)
m <- length(pval)
qStar <- 0.01
# Type I error rate target value

# Return a list of significant genes, while controlling FWER at q = 0.0
1
fwer_genes <- m * pval < qStar
table(fwer_genes)

## fwer_genes
## FALSE  TRUE
## 20080    92

# 92 genes are differentially expressed at q = 0.01 (based on Bonferron
i procedure)

# Return a list of significant genes, while controlling FDR at q = 0.05
fdr_genes <- qvalue(p = pval, lambda = 0, fdr.level = qStar)$significan
t
table(fdr_genes)

## fdr_genes
## FALSE  TRUE
## 19219   953

# 953 genes are differentially expressed at q = 0.01 (based on BH proce
dure)

# Return a list of significant genes, while controlling FDR at q = 0.05
qvalue_genes <- qvalue(p = pval, fdr.level = qStar)$significant
table(qvalue_genes)

## qvalue_genes
## FALSE  TRUE
## 18485  1687

# 1687 genes are differentially expressed at q = 0.01 (based on pFDR pr
ocedure)
```

Venn them

```
# n1 <- sum(fwer_genes)
# n2 <- sum(fdr_genes)
# n3 <- sum(qvalue_genes)
# n12 <- sum(fwer_genes * fdr_genes)
# n13 <- sum(fwer_genes * qvalue_genes)
# n23 <- sum(fdr_genes * qvalue_genes)
```

```
# n123 <- sum(fwer_genes * fdr_genes * qvalue_genes )

# pdf(file = 'venn_type1errors.pdf', width = 6, height = 6)
# grid.newpage()
# draw.triple.venn(area1 = n1, area2 = n2, area3 = n3, n12 = n12, n23 =
n23, n13 = n13, alpha = rep(0.7, 3),
# n123 = n123, category = c("FWER", "FDR", "qValue"), lty = "blank",
# fill = c("skyblue", "pink1", "mediumorchid"), cat.dist = c( -0.01, -0
.02, -0.02), cat.cex = 2, cat.pos = 180, cex = 2)
# dev.off()
```

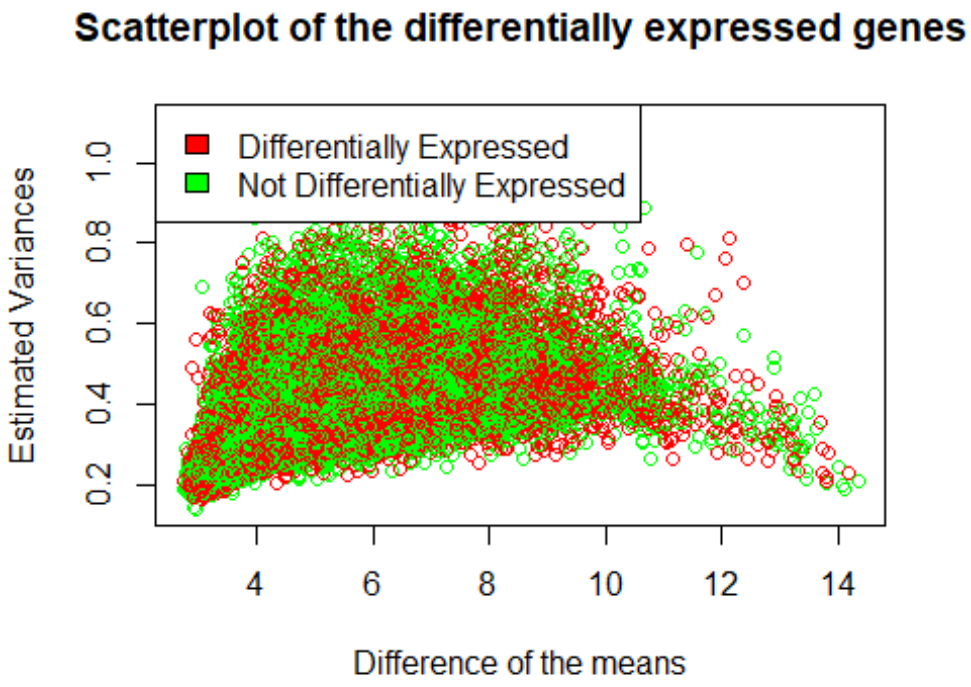


We observe that qValue performs better in comparison to FWER and FDR, that is to mean that it provides us more genes that are indeed differentially expressed!

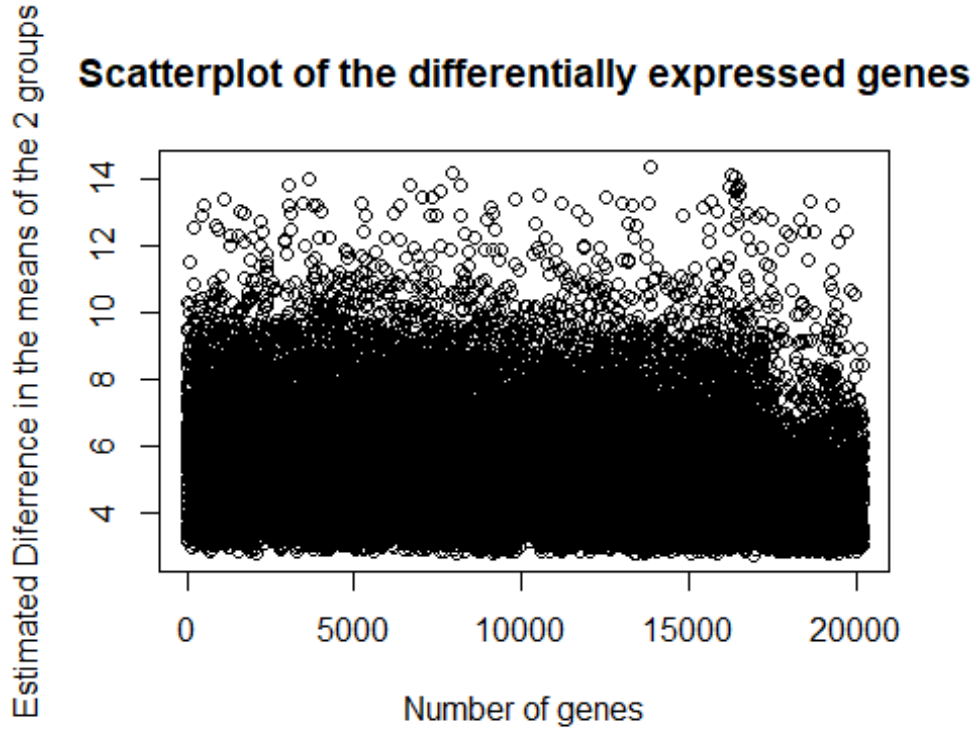
Question 6

Plotting the colored genes when controlling the FDR

```
estimated_var <- sqrt(std.err)
plot(diff, estimated_var,  col=c('red','green'), main='Scatterplot of t
he differentially expressed genes', xlab='Difference of the means',
ylab='Estimated Variances')
legend('topleft',legend = c('Differentially Expressed','Not Differentia
lly Expressed'),fill = c('red','green'))
```



```
plot(diff , col=as.factor(fdr_genes[fdr_genes==T]), main='Scatterplot o
f the differentially expressed genes', xlab='Number of genes',
ylab='Estimated Diferrence in the means of the 2 groups')
```

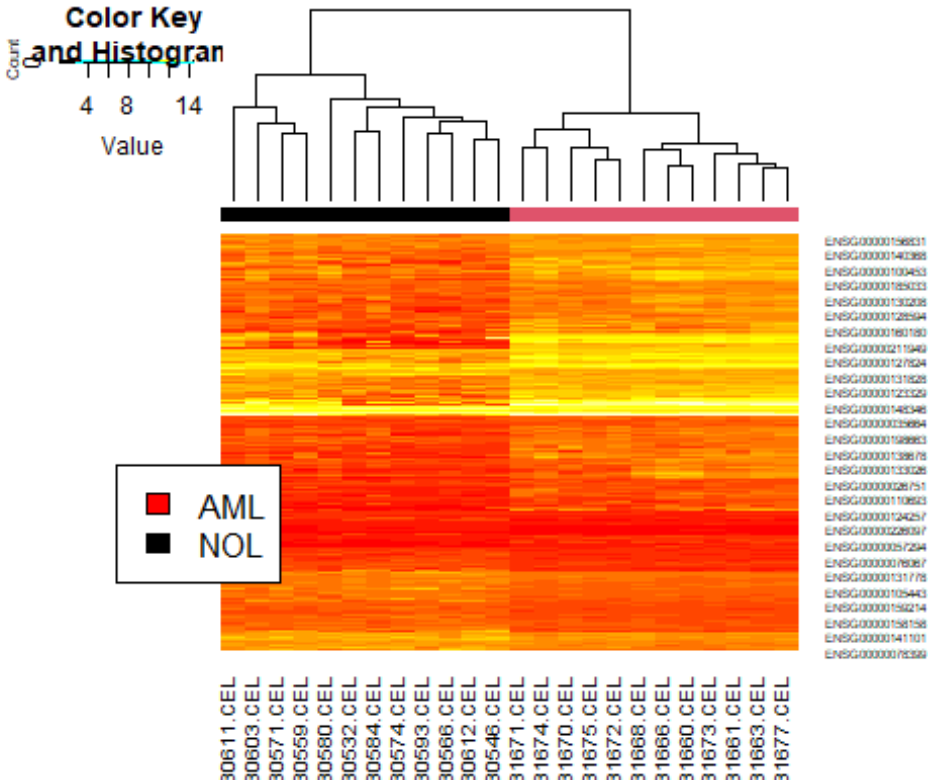



```
range(diff)
```

```
## [1]  2.718753 14.358476
```

Creating a Heatmap

```
library(gplots)
fdr <- p.adjust(p=pval,method = 'fdr')
heat <- new_mat[fdr < 0.01,]
factorlvl <- gl(2, 12, labels = c("AML", "NOL"))
heatmap.2(heat, trace = "none", ColSideColors = as.character(as.numeric
(factorlvl)), dendrogram = c('column'))
legend('bottomleft',legend = c('AML','NOL'),fill = c('red','black'))
```



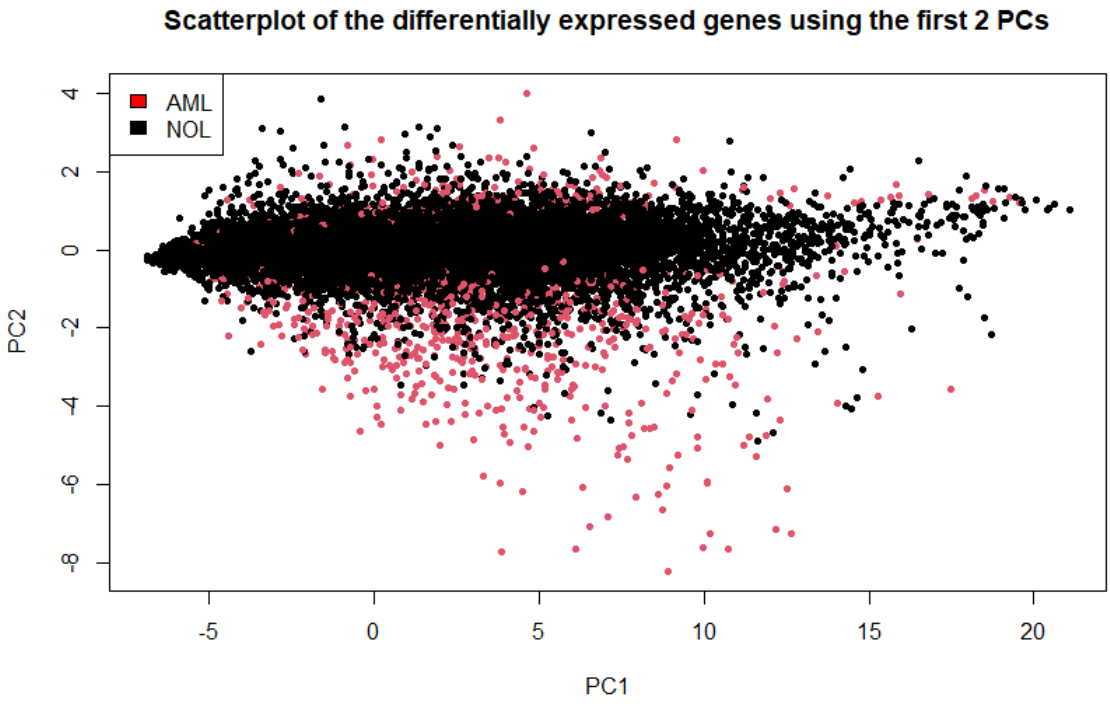
We observe that there are some differences observed between the 2 groups at some points. However, we would prefer the previous plots as they are more informative as regards the genes that are differentially expressed.

plot(set.pr$scores[, 1:2], col = as.factor(fdr_genes), pch = 20,

main='Scatterplot of the differentially expressed genes using the first 2 PCs',
xlab='PC1',

ylab='PC2')

legend("topleft", legend=c('AML','NOL') ,fill = c('red','black'))



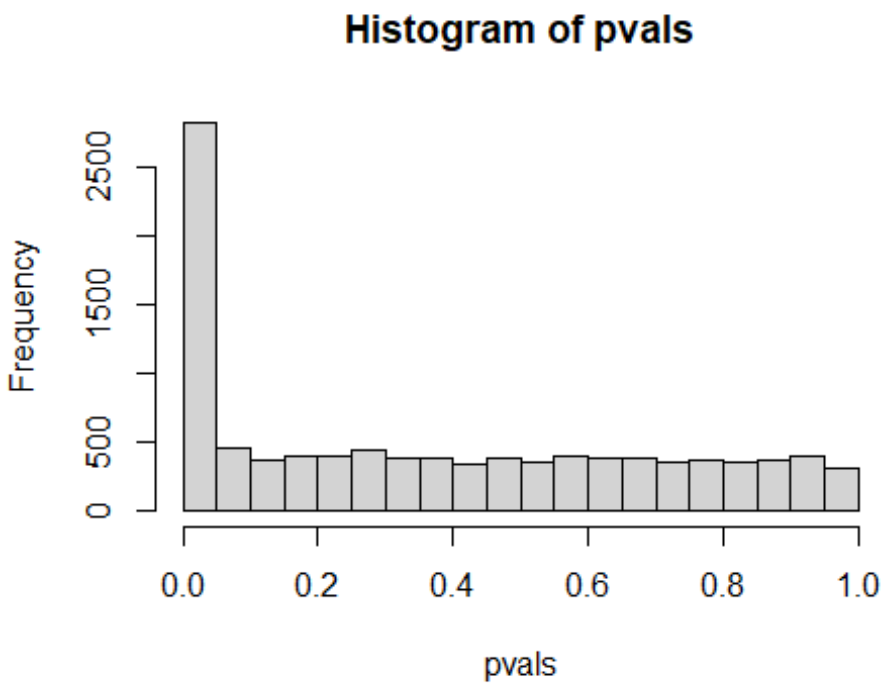
We observe that there's a lot of overlap between the 2 groups but the situation is clearly better than before!

Using the FDR made improvements in the partitioning of the 2 groups!

Exercise 2

Question 1 Multiple testing simulation study

```
n <- 500
p <- 100
b <- numeric(p)
m <- 10000
ground.truth <- numeric(m)
pvals <- numeric(m)
for (j in 1:m) {
  x <- matrix(rnorm(n*p),nrow = n, ncol = p)
  # Generate the p regression coefficients b1, . . . , bp as follows:
  b <- numeric(p)
  if( runif(1) < 0.3){ b[1] <- rnorm(1) }
  # Generate the values of the response variable from a typical normal
linear model, that is:
  y <- x%*%b + rnorm(n)
  ground.truth[j] <- ifelse(b[1]==0, 1, 2)
# This means that βi = 0 for all i > 2, while the first coefficient (b1
) is zero with probability
# 0.7, while it is different than zero with probability 0.3.
  df <- data.frame(x)
  f <- lm(y~., data = df)
  pvals[j] <- pf(summary(f)$fstatistic[1], summary(f)$fstatistic[2], su
mmary(f)$fstatistic[3], lower.tail=FALSE)
}
hist(pvals,30)
```




```
table(ground.truth)
```

```
## ground.truth
##    1    2
## 6983 3017
```

```
# 1 refers to true null hypothesis, while 2 refers to non-true null hyp
othesis

qstar <- 0.05
# Type I error rate target value

sortedP <- pvals[order(pvals)]
# Necessary for the Bonferroni procedure

bonfSelected <- sortedP < qstar/m
table(bonfSelected)
```

```
## bonfSelected
## FALSE  TRUE
##  7987  2013
```

```
# Based on Bonferroni procedure we will reject 2020 hypotheses and we w
ill fail to reject 7980 hypotheses

# Now let's create a confusion matrix regarding the Bonferroni procedur
e
```

```r
tt <- table(ground.truth[order(pvals)],bonfSelected)
tt
```

```
##    bonfSelected
##     FALSE TRUE
##   1  6983    0
##   2  1004 2013
```

```r
tt[2,2]/sum(tt[2,])
```

```
## [1] 0.6672191
```

```r
# Estimated power

# Second way of controlling FWER with Bonferroni procedure using the p.
adjust command
adjustedP_Bonf <- p.adjust(p=pvals,method = 'bonf')
bonfSelected2 <- adjustedP_Bonf < qstar
table(bonfSelected2)
```

```
## bonfSelected2
## FALSE  TRUE
##  7987  2013
```

```r
# As expected it's the same output as before

# Now let's create a confusion matrix regarding the Bonferroni procedur
e
tt2 <- table(ground.truth,bonfSelected2)
tt2
```

```
##             bonfSelected2
## ground.truth FALSE TRUE
##            1  6983    0
##            2  1004 2013
```

```r
tt2[2,2]/sum(tt2[2,])
```

```
## [1] 0.6672191
```

```r
# Estimated power



# Controlling FWER with Holm procedure
adjustedP_holm <- p.adjust(p=pvals,method = 'holm')
holmSelected <- adjustedP_holm < qstar
table(holmSelected)
```

```
## holmSelected
## FALSE  TRUE
##  7981  2019
```

```r
# Now let's create a confusion matrix regarding the Holm procedure
tt.h <- table(ground.truth,holmSelected)
tt.h
```

```
##             holmSelected
## ground.truth FALSE TRUE
##            1  6983    0
##            2   998 2019
```

```r
tt.h[2,2]/sum(tt.h[2,])
```

```
## [1] 0.6692078
```

```r
# Estimated power
```

```r
# Controlling FWER with Hochberg procedure
adjustedP_hochberg <- p.adjust(p=pvals,method = 'hochberg')
hochbergSelected <- adjustedP_hochberg < qstar
table(hochbergSelected)
```

```
## hochbergSelected
## FALSE  TRUE
##  7981  2019
```

```r
# Now let's create a confusion matrix regarding the Hochberg procedure
tt.hb <- table(ground.truth,hochbergSelected)
tt.hb
```

```
##             hochbergSelected
## ground.truth FALSE TRUE
##            1  6983    0
##            2   998 2019
```

```r
tt.hb[2,2]/sum(tt.hb[2,])
```

```
## [1] 0.6692078
```

```r
# Estimated power
```

```r
# Controlling FWER with Hommel procedure
adjustedP_hommel <- p.adjust(p=pvals,method = 'hommel')
hommelSelected <- adjustedP_hommel < qstar
table(hommelSelected)
```

```
## hommelSelected
## FALSE  TRUE
##  7980  2020
```

```r
# Now let's create a confusion matrix regarding the Hommel procedure
tt.hml <- table(ground.truth,hommelSelected)
tt.hml
```

```
##             hommelSelected
## ground.truth FALSE TRUE
##            1  6983    0
##            2   997 2020
```

```r
tt.hml[2,2]/sum(tt.hml[2,])
```

```
## [1] 0.6695393
```

```r
# Estimated power
```

```r
# Controlling FDR with BH procedure
adjustedP_BH <- p.adjust(p=pvals,method = 'BH')
BHSelected <- adjustedP_BH < qstar
table(BHSelected)
```

```
## BHSelected
## FALSE  TRUE
##  7546  2454
```

```r
# Now let's create a confusion matrix regarding the BH procedure
tt.BH <- table(ground.truth,BHSelected)
tt.BH
```

```
##             BHSelected
## ground.truth FALSE TRUE
##            1  6893   90
##            2   653 2364
```

```r
tt.BH[2,2]/sum(tt.BH[2,])
```

```
## [1] 0.7835598
```

```r
# Estimated power
```

```r
# Controlling FDR with BY procedure
adjustedP_BY <- p.adjust(p=pvals,method = 'BY')
BYSelected <- adjustedP_BY < qstar
table(BYSelected)
```

```
## BYSelected
## FALSE  TRUE
##  7766  2234
```

```r
# Now let's create a confusion matrix regarding the BY procedure
tt.BY <- table(ground.truth,BYSelected)
tt.BY
```

```
##             BYSelected
## ground.truth FALSE TRUE
##            1  6980    3
##            2   786 2231
```

```r
tt.BY[2,2]/sum(tt.BY[2,])
```

```
## [1] 0.7394763
```

```r
# Estimated power
```

```r
# Controlling pFDR
pFDRSelected <- qvalue(p = pvals, fdr.level = qstar)$significant
table(pFDRSelected)
```

```
## pFDRSelected
## FALSE  TRUE
##  7481  2519
```

```r
# Now let's create a confusion matrix regarding the pFDR procedure
tt.pFDR <- table(ground.truth,pFDRSelected)
tt.pFDR
```

```
##             pFDRSelected
## ground.truth FALSE TRUE
##            1  6852  131
##            2   629 2388
```

```r
tt.pFDR[2,2]/sum(tt.pFDR[2,])
```

```
## [1] 0.7915147
```

```r
# Estimated power
```

Question 2

```
# Creating the vectors to store the values
alpha <- seq(0.001,0.999,0.01)
bonf <- numeric(length(alpha))
holm <- numeric(length(alpha))
hb <- numeric(length(alpha))
hml <- numeric(length(alpha))
bh <- numeric(length(alpha))
by <- numeric(length(alpha))
q <- numeric(length(alpha))


for (i in 1:length(alpha)) {
  adjustedP_Bonf <- p.adjust(p=pvals,method = 'bonf')
  bonfSelected <- adjustedP_Bonf < alpha[i]
  table.bonf <- table(ground.truth, bonfSelected)
  bonf[i] <- table.bonf[2,2]/sum(table.bonf[2,])
}



for (i in 1:length(alpha)) {
  adjustedP_holm <- p.adjust(p = pvals, method = 'holm')
  holmSelected <- adjustedP_holm < alpha[i]
  table.holm <- table(ground.truth, holmSelected)
  holm[i] <- table.holm[2,2]/sum(table.holm[2,])
}



for (i in 1:length(alpha)) {
  adjustedP_hb <- p.adjust(p = pvals, method = 'hochberg')
  hbSelected <- adjustedP_hb < alpha[i]
  table.hb <- table(ground.truth, hbSelected)
  hb[i] <- table.hb[2,2]/sum(table.hb[2,])
}



for (i in 1:length(alpha)) {
  adjustedP_hml<- p.adjust(p = pvals, method = 'hommel')
  hmlSelected <- adjustedP_hml < alpha[i]
  table.hml <- table(ground.truth, hmlSelected)
  hml[i] <- table.hml[2,2]/sum(table.hml[2,])
}
```

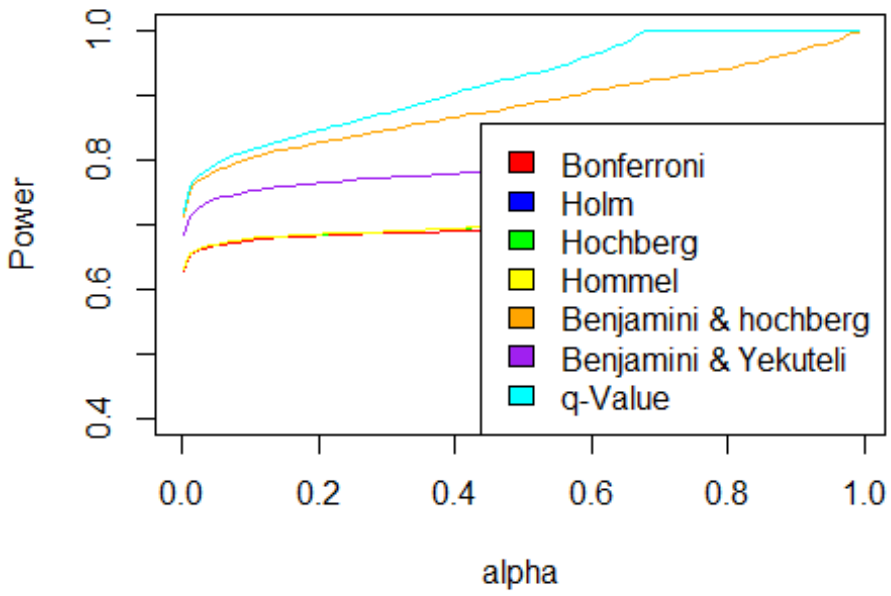
```r
for (i in 1:length(alpha)) {
  adjustedP_BH <- p.adjust(p = pvals, method = 'BH')
  BHSelected <- adjustedP_BH < alpha[i]
  table.bh <- table(ground.truth, BHSelected)
  bh[i] <- table.bh[2,2]/sum(table.bh[2,])
}
```

```r
for (i in 1:length(alpha)) {
  adjustedP_BY <- p.adjust(p = pvals, method = 'BY')
  BYSelected <- adjustedP_BY < alpha[i]
  table.by <- table(ground.truth, BYSelected)
  by[i] <- table.by[2,2]/sum(table.by[2,])
}
```

```r
for (i in 1:length(alpha)) {
  q.value <- qvalue(p = pvals, fdr.level = alpha[i])$significant
  if (sum(qvalue(p = pvals, fdr.level = alpha[i])$significant)==10000)
{
    q[i] <- 1} else {
      table.q <- table(ground.truth, q.value)
      q[i] <- table.q[2,2]/sum(table.q[2,])
    }
}
```

Plotting the different procedures

```
plot(alpha, bonf, type = "l", ylab = "Power", ylim=c(0.4,1), col='red')
lines(alpha, holm, type="l", col='blue')
lines(alpha, hb, type="l", col='green')
lines(alpha, hml, type="l", col='yellow')
lines(alpha, bh, type="l", col='orange')
lines(alpha, by, type="l", col='purple')
lines(alpha, q, type="l", col='cyan')
legend("bottomright", fill = c('red','blue','green','yellow','orange','
purple','cyan'), legend = c("Bonferroni", "Holm",
"Hochberg", "Hommel", "Benjamini & hochberg", "Benjamini & Yekuteli", "
q-Value"))
```



We observe that the qValue has the highest Power indicating that it performs better overall! The overall ranking of the methods goes as follows:

1) **qValue**
2) **Benjamini & Hochberg**
3) **Benjamini & Yekutieli**
4) **Hommel / Hochberg / Holm (equal results as they are overlapping one another)**
5) **Bonferroni**