# Project_Glioblastoma

*Michela Notarangelo*

*23 marzo 2016*

# 1 1. How many genes and samples are there?

```
library(Biobase)
setwd("~/git/AppStatTrento/projects") # da commentare
eset <- readRDS("GBM_RNAseq2.rds")
subtypes = read.csv("GBM_subtypes.csv", as.is=TRUE)[, 1:2]
sampleNames(eset) = toupper(substr(sampleNames(eset), 1, 12))
dim(eset)
```

```
## Features  Samples
##    20501      152
```

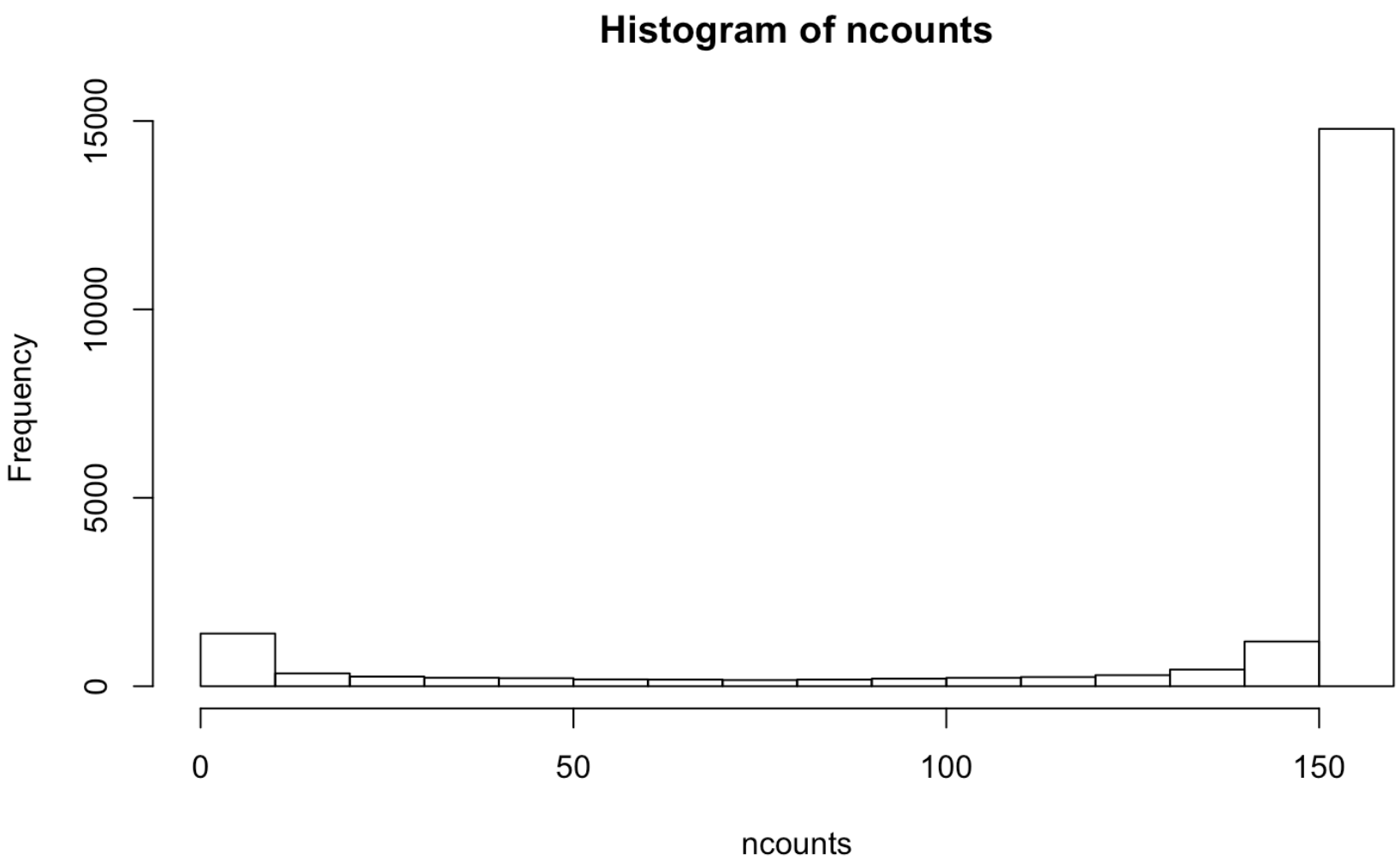
1. There are 20501 genes and 152 samples.

# 2 2. For each gene, calculate the number of samples where it has at least one count. How many genes have at least one count in at least 10% of the samples? Present a histogram of the % of samples in which each gene has 1 or more counts.

```
ex = exprs(eset)
ncounts = apply (ex, 1, function (x) sum(x>0))
summary(ncounts)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     0.0   148.0   152.0   130.3   152.0   152.0
```

```
subtypesx <- length(which(ncounts >= ncol(eset)/10))
hist(ncounts)
```

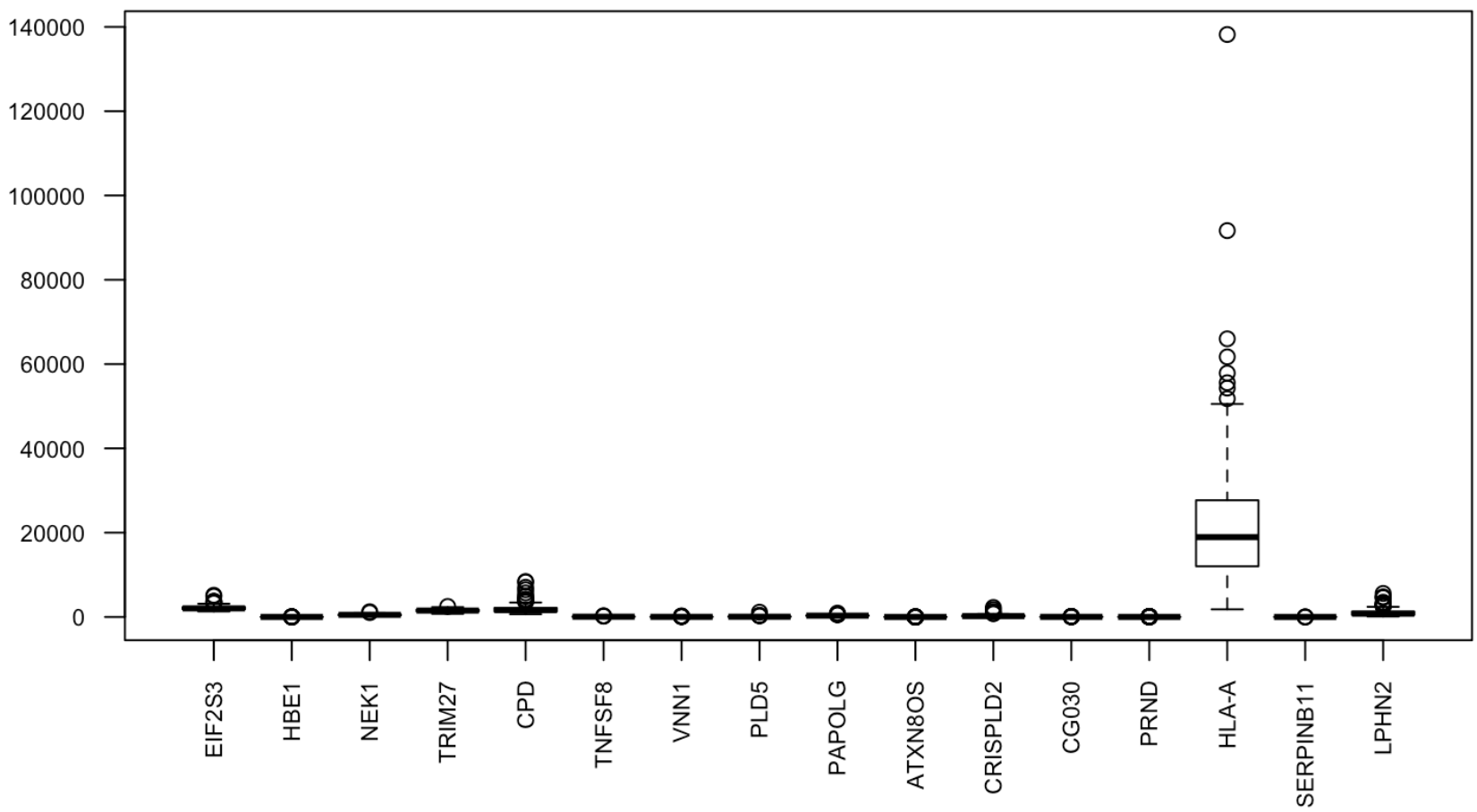



```
phenotype <- pData(eset)[row.names(pData(eset)) %in% subtypes$sample.id,]
subt.eset <- exprs(eset)[,sampleNames(eset) %in% subtypes$sample.id]
```

2. All the samples have at least one count. 18924 genes out of 20501.

# 3 3. Create a single boxplot showing the expression counts of 16 randomly selected genes. Do the data appear normally distributed?

```
set.seed(1)
randomGenes = sample (1:nrow(ex), size = 16)
par(las=2)
boxplot(t(ex [randomGenes,]),cex.axis=.7)
```

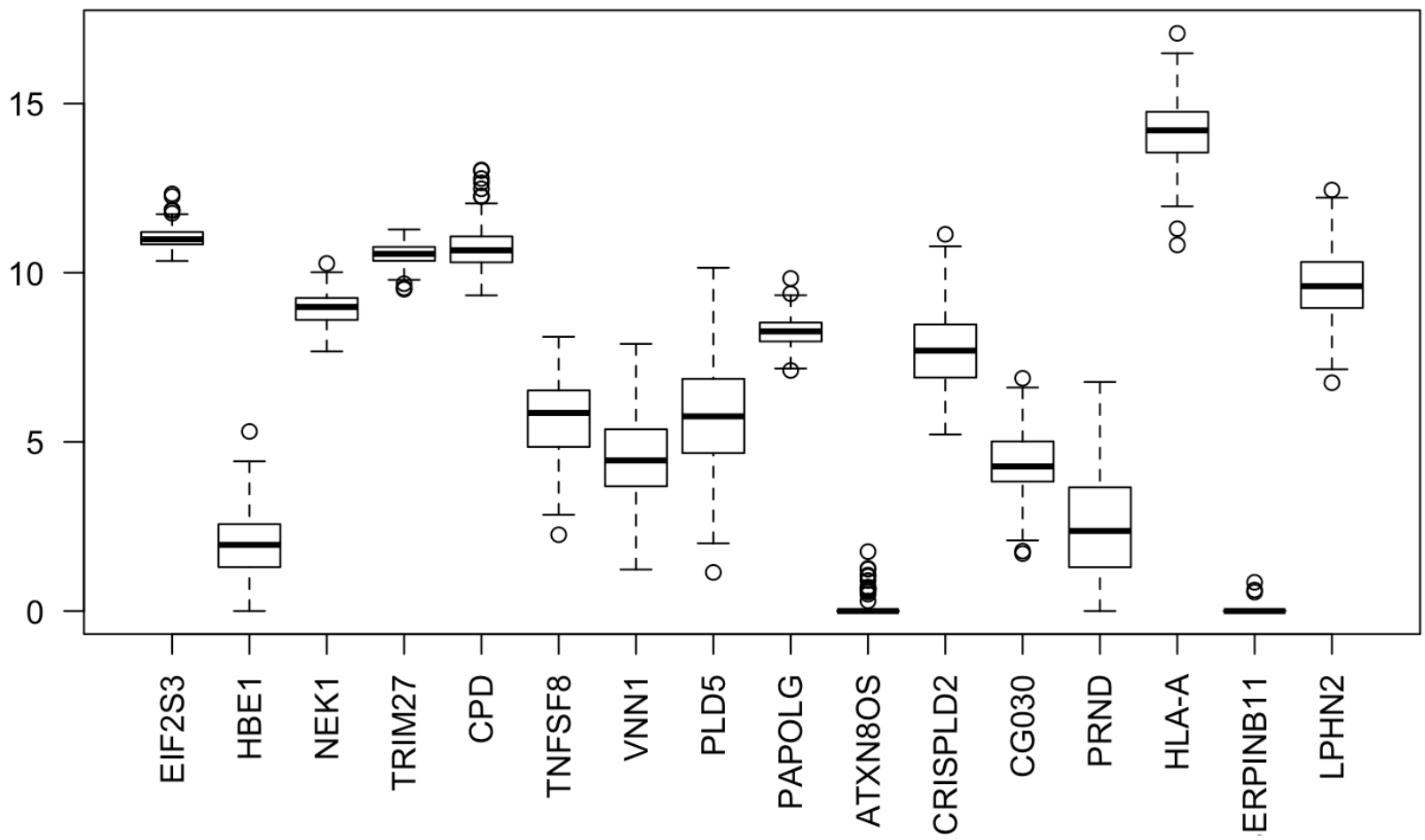



3. No, the data do not appear normally distributed.

4 4. Create another boxplot for the same genes, after adding 1 and taking base-2 logarithm of the expression counts. Is this more normally distributed? If so, use this transformed version of the dataset for further exploratory analysis.

```
 #>randomGenes
 #[1] 20403 19102  1092  8131  1424 16129  5015 12419  7230  3903 12838 1228
5 10006   794  2854 15571

log_randomGenes = log2(ex [randomGenes,] + 1)
par(las=2)
boxplot(t(log_randomGenes))
```

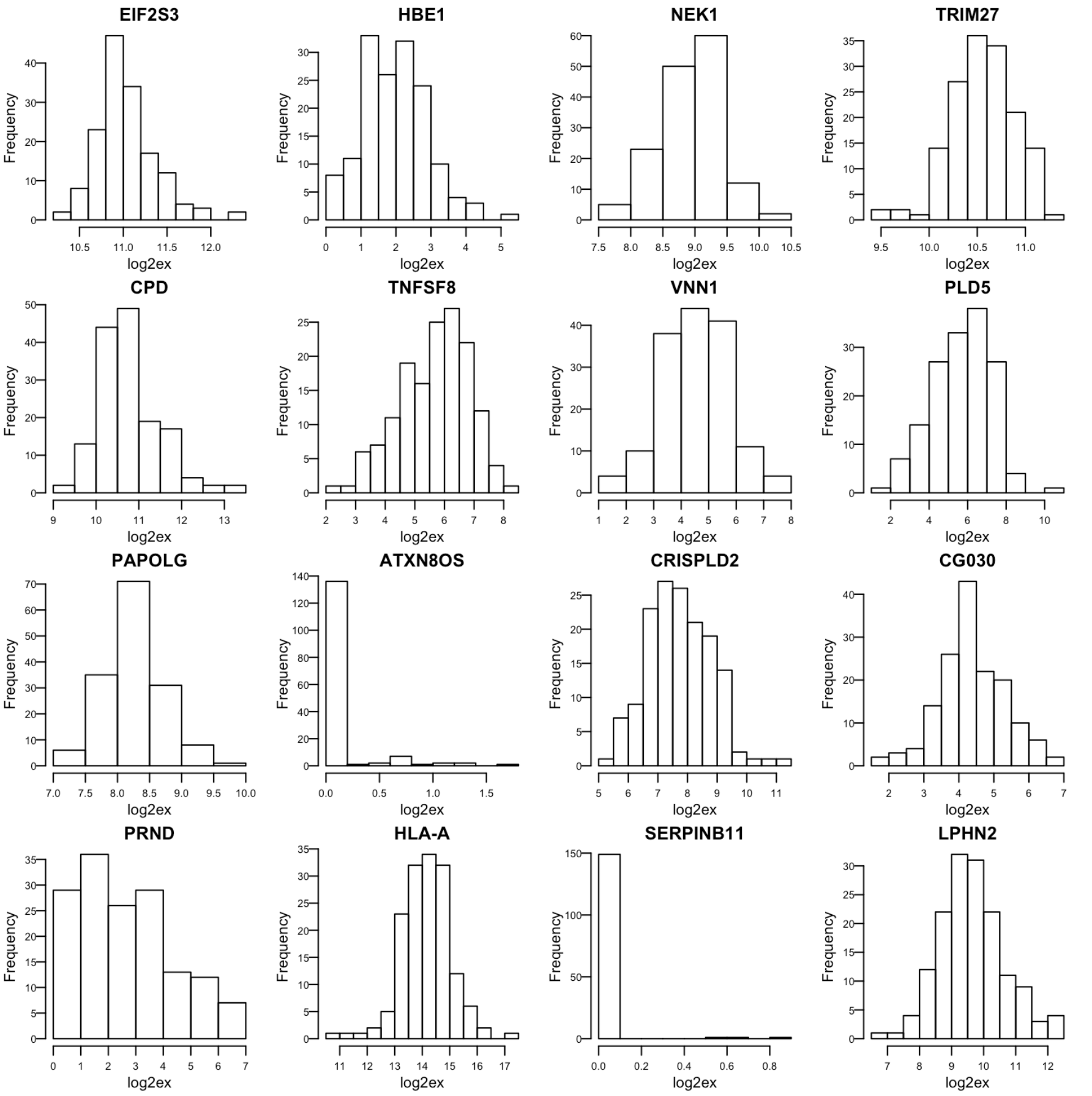



4. Yes, now the data look more normally distributed.

5 5. For another look at these gene expression distributions, make 16 histograms on a 4x4 grid.

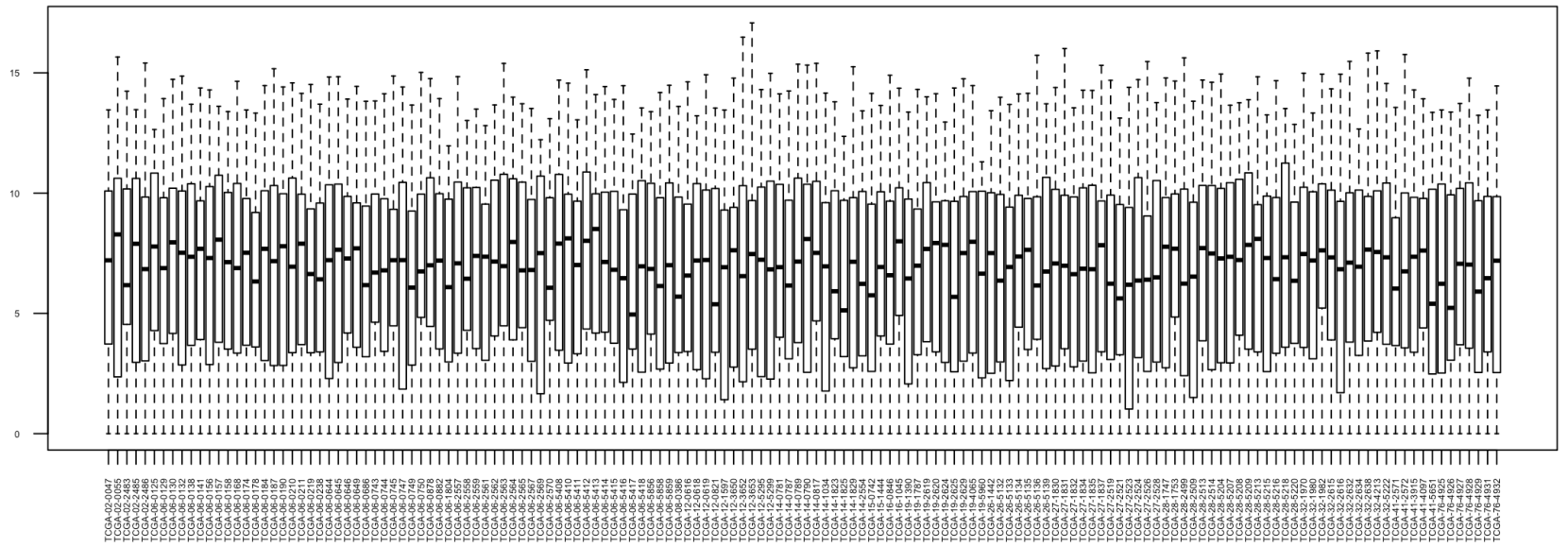
```
rafalib::mypar(mfrow = c(4,4), las=1, cex.axis=.7)
# apply(log_randomGenes,c(1), function(x) hist(x, xlab = "log2ex"))

for (i in 1:nrow(log_randomGenes)) {
  hist(log_randomGenes[i,], xlab="log2ex",
       main=rownames(log_randomGenes)[i])
}
```




6 6. Using whichever transformation between 4 and 5 resulted in a more nearly normal distribution, create a boxplot showing expression values for each sample. Do any samples look like outliers?

```
par(mfrow = c(1,1), las=2, cex.axis=.4)
boxplot(log_randomGenes)
```




6.The samples look seem to be pretty homogeneous, it could be said that there is almost any outlier.

# 7 subtypes?

560 subtypes per 152 expression data (no filtering needed)

```
subtypes = subtypes[match(sampleNames(eset), subtypes$sample.id), ]
eset$subtype = subtypes$Cluster
eset$subtype
```

```
##   [1] "Proneural"   "Mesenchymal" "G-CIMP"      "Classical"   "Mesenchyma
l"
##   [6] "Classical"   "G-CIMP"      "Mesenchymal" "Neural"      "Neural"
##  [11] "Mesenchymal" "Proneural"   "Classical"   "Classical"   "Mesenchyma
l"
##  [16] "Proneural"   "Neural"      "Mesenchymal" "Classical"   "Mesenchyma
l"
##  [21] "Mesenchymal" "Classical"   "Neural"      "Proneural"   "Mesenchyma
l"
##  [26] "Mesenchymal" "Proneural"   "Neural"      "Proneural"   "Classical"
##  [31] "Classical"   "Proneural"   "Classical"   "Neural"      "Mesenchyma
l"
##  [36] "Mesenchymal" "Neural"      "Classical"   "Mesenchymal" "Proneural"
##  [41] "Proneural"   "Mesenchymal" "Mesenchymal" "Classical"   "Classical"
##  [46] "Classical"   "Neural"      "Mesenchymal" "G-CIMP"      "Classical"
##  [51] "Mesenchymal" "Neural"      "Mesenchymal" "Neural"      "Classical"
##  [56] "Classical"   "Proneural"   "G-CIMP"      "Mesenchymal" "Classical"
##  [61] "Mesenchymal" "Neural"      "Neural"      "Proneural"   "Proneural"
##  [66] "Mesenchymal" "Neural"      "Proneural"   "Proneural"   "Classical"
##  [71] "Classical"   "Neural"      "Classical"   "Mesenchymal" "Classical"
##  [76] "Mesenchymal" "Classical"   "Neural"      "Mesenchymal" "Mesenchyma
l"
##  [81] "Proneural"   "Neural"      "Neural"      "Classical"   "Proneural"
##  [86] "Proneural"   "Mesenchymal" "Proneural"   "Mesenchymal" "Classical"
##  [91] "Neural"      "Proneural"   "Classical"   "G-CIMP"      NA
##  [96] "Proneural"   "G-CIMP"      "Classical"   "G-CIMP"      "Proneural"
## [101] "Proneural"   "Mesenchymal" "Mesenchymal" "Proneural"   "Neural"
## [106] "Mesenchymal" "Neural"      "Classical"   "Classical"   "Mesenchyma
l"
## [111] "G-CIMP"      "Classical"   "Mesenchymal" "Neural"      "Classical"
## [116] "Classical"   "Mesenchymal" NA            "Mesenchymal" "Mesenchyma
l"
## [121] "Classical"   "Neural"      "Mesenchymal" "Mesenchymal" "Mesenchyma
l"
## [126] "Mesenchymal" "Mesenchymal" "Mesenchymal" "Mesenchymal" "Classical"
## [131] "Classical"   "Neural"      "Neural"      "Mesenchymal" "Mesenchyma
l"
## [136] "Mesenchymal" "Proneural"   "Classical"   "Mesenchymal" "Proneural"
## [141] "Proneural"   "Classical"   "Mesenchymal" "Mesenchymal" "Proneural"
## [146] "Proneural"   "Classical"   "Neural"      "Classical"   "Neural"
## [151] "Classical"   "Proneural"
```

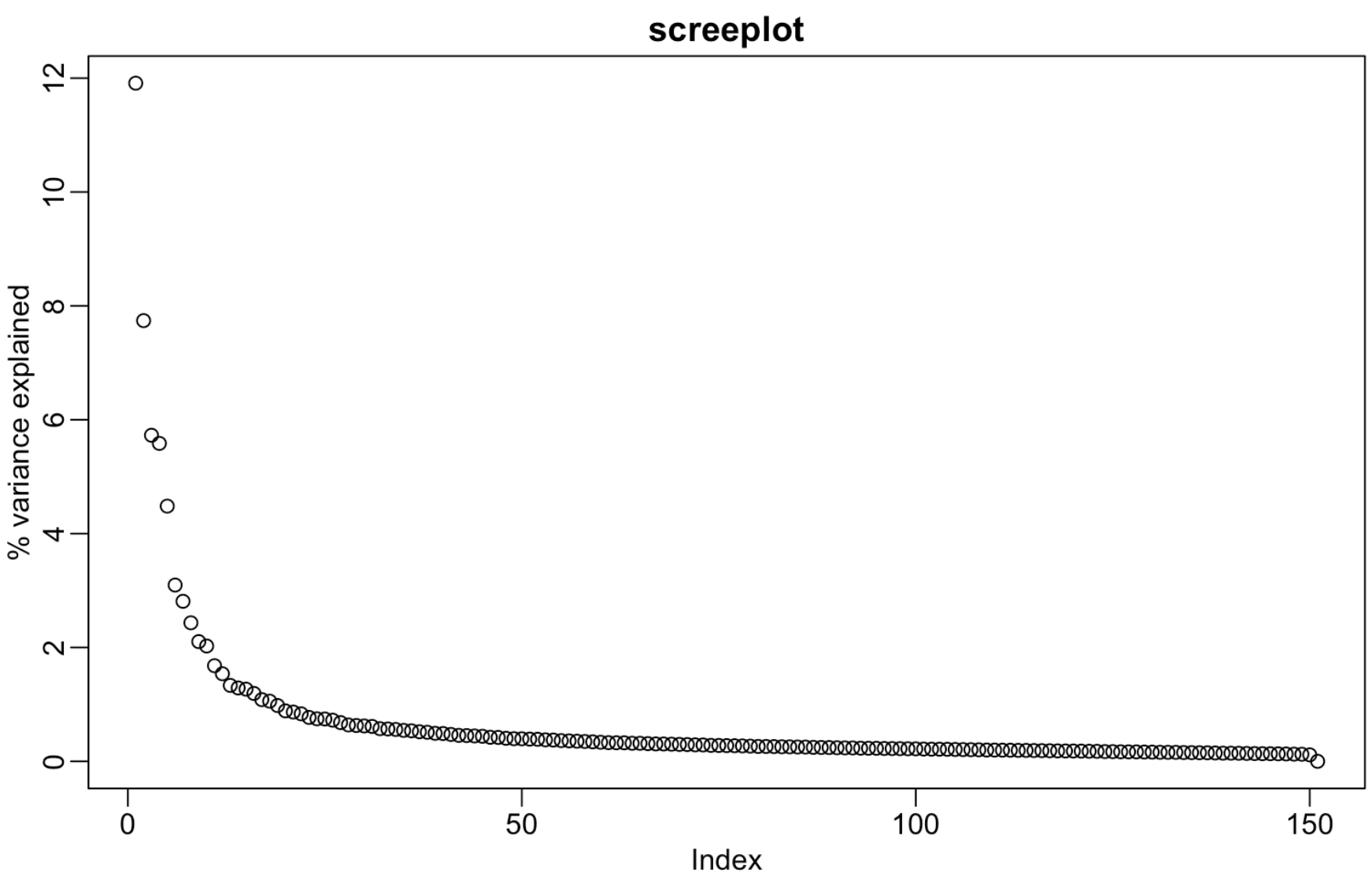
```
pc = princomp(log2(ex+1), cor = TRUE)
```

# 8 7. A visual look for cluster structure

9 Remove any genes that have non-zero counts in fewer than 10% of the samples. Using the more normally distributed transformation of the data, perform a Principal Components Analysis (PCA). A standard base R solution for PCA is the prcomp() function. Be sure to include variance scaling for the genes, which can be done manually or as an option to the prcomp() function. Show the screeplot of your PCA, and comment.

```
library(Biobase)
# subtypes = exprs(eset)[, subtypes$sample.id %in% sampleNames(eset)]
subtypes10 <- log2(subt.eset[ncounts > ncol(subt.eset)/10, ] +1)

subtypes10Scale <- t(scale(t(subtypes10), scale=FALSE))
p <- prcomp(subtypes10Scale)
rafalib::mypar()
plot(p$sdev^2 / sum(p$sdev^2)*100, ylab="% variance explained", main = 'scre
eplot')
```




```
sum((p$sdev^2 / sum(p$sdev^2)*100)[1:2])
```

```
## [1] 19.65223
```

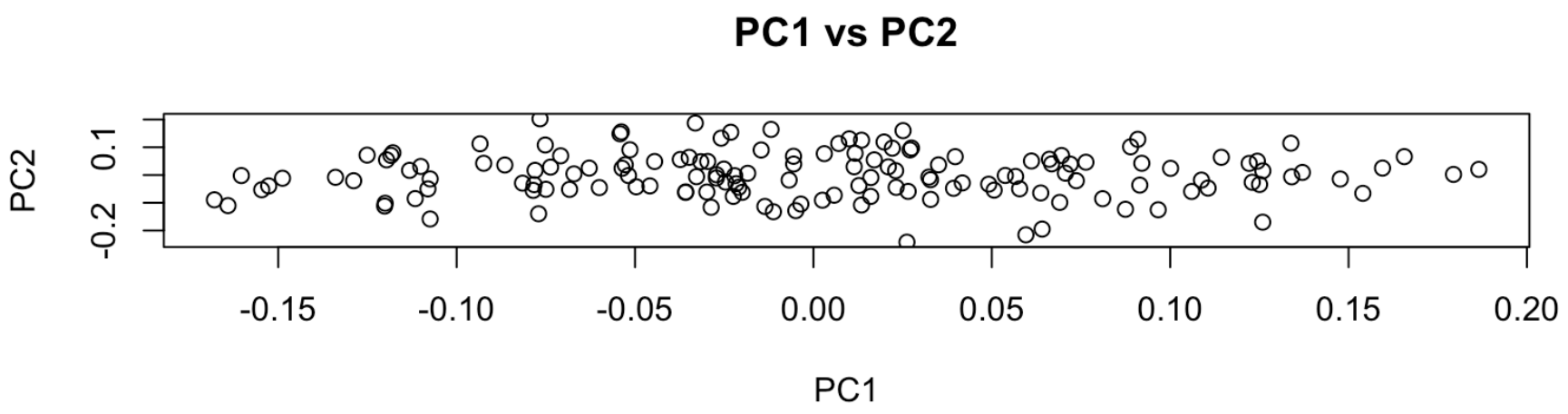
7. 19.63343 % is the variance expressed by my first two principal components, that makes them quite representative of my dataset,

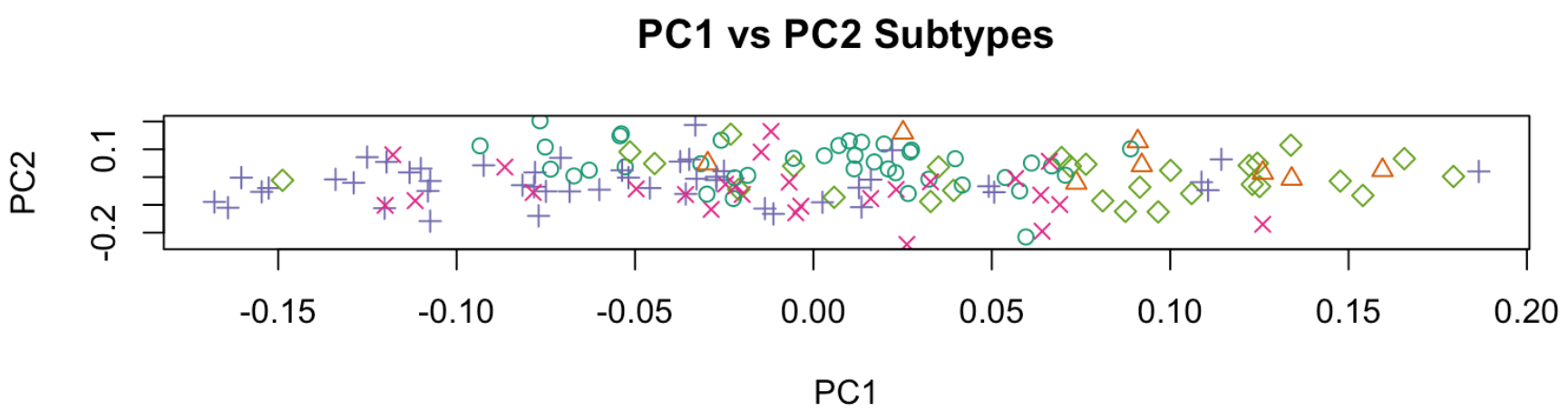
# 10 Differential Expression

# 11 8. Make two side-by-side plots of PC2 vs PC1. In the first, do not distinguish between subtypes. In the second, use color and data point shapes to distinguish between the reported subtypes. Do you see any obvious structure or clustering of the samples? Be VERY sure you have correctly aligned patient barcodes in the ExpressionSet and the barcodes file, for example using

identical().

```
par(mfrow= c(2,1))
plot(p$rotation[,1:2], xlab = "PC1", ylab = "PC2", main = "PC1 vs PC2")
plot(p$rotation[,1:2], xlab="PC1", ylab="PC2", main= "PC1 vs PC2 Subtypes",
col=factor(factor(eset$subtype)), pch=as.integer(factor(eset$subtype)))
```




8. Yes, I can see two main clusters on the right and on the left of my second plot, that follow the trend of two out of five subtypes.

# 12 9. Use the DESeq2 Bioconductor package to assess differential expression of each gene across the subtypes, using a one-way Analysis of Variance (ANOVA). Create a histogram of raw p-values.

```
subtypes<- na.exclude(subtypes)
subtypes10<- na.exclude(subtypes10)
library(DESeq2)
```

```
## Loading required package: S4Vectors
```

```
## Loading required package: stats4
```

```
## Loading required package: IRanges
```

```
## Loading required package: GenomicRanges
```

```
## Loading required package: GenomeInfoDb
```

```
## Loading required package: SummarizedExperiment
```

```
## Loading required package: Rcpp
```

```
## Loading required package: RcppArmadillo
```

```
subtypes.sort <- subtypes[order(subtypes$sample.id),]
subtypes.sort.clean <- subtypes.sort[subtypes.sort$sample.id %in% colnames(s
ubtypes10), ]
condition <- factor(subtypes.sort.clean$Cluster)
condition[151] <- "Mesenchymal"#: 150? reporting missing type
countData <- matrix(as.integer(subtypes10), ncol=151)
dds <- DESeq2::DESeqDataSetFromMatrix(countData, DataFrame(condition), ~ con
dition)

dds <- dds[rowSums(counts(dds)) >1,]
dds <- DESeq2::DESeq(dds)
```

```
## estimating size factors
```

```
## estimating dispersions
```

```
## gene-wise dispersion estimates
```

```
## mean-dispersion relationship
```

```
## -- note: fitType='parametric', but the dispersion trend was not well capt
ured by the
##    function: y = a/x + b, and a local regression fit was automatically su
bstituted.
##    specify fitType='local' or 'mean' to avoid this message next time.
```

```
## final dispersion estimates
```

```
## fitting model and testing
```

```
## -- replacing outliers and refitting for 15 genes
## -- DESeq argument 'minReplicatesForReplace' = 7
## -- original counts are preserved in counts(dds)
```

```
## estimating dispersions
```

```
## fitting model and testing
```

```
res <- DESeq2::results(dds)

hist(res$pvalue, xlab = "p-value", main = "Raw p-values")
```