Specific Analysis complete

Laura Plutowski and Franziska Heinkele

26 Juni 2019

Load data:

```
library(readr)
  Untreated = readRDS(paste0(wd,"/data/NCI_TPW_gep_untreated.rds"))
  Treated = readRDS(paste0(wd,"/data/NCI_TPW_gep_treated.rds"))
  Metadata = read_tsv(paste0(wd,"/data/NCI_TPW_metadata.tsv"))
  Drug_annotation = read.table(paste0(wd,"/data/drug_annotation.tsv"),
                               header = TRUE, sep ="\t", stringsAsFactors = TRUE)
```

SPECIFIC ANALYSIS

From now on, we decided to focus our analysis on Vorinostat.

```
Drug_annotation$Mechanism[grep(Drug_annotation$Drug, pattern= "vorinostat")]
```

```
## [1] HDAC inhibitor
## 12 Levels: DNA alkyltransferase inhibitor ...
```

The annotation data include information about vorinostats working mechanism as a **HDAC inhibitor** (HDAC = histone deacetylases). Normally, cells maintain a dynamic balance between acetylated and deacetylated histones.

- **Acetylated form**: Histones loose their positive charge when they are acetylated, what means that the negatively charged DNA is less bound to the histone proteins and the genes are more accessible. This form is promoted by the HDAC-Inhibitors, including vorinostat.
- **Deacetylated form**: Histones have a positive charge which causes a strong DNA binding. Genes can be hardly transcribed, including tumor-suppressorgenes. This form is promoted by the HDACs.

Vorinostats application as an anti-cancer medicine might origin from a better access to tumor-suppressorgenes in the acetylated form.

```
Drug_annotation$target[grep(Drug_annotation$Drug, pattern= "vorinostat")]
```

```
## [1] HDAC1|HDAC10|HDAC11|HDAC2|HDAC3|HDAC5|HDAC6|HDAC8|HDAC9
## 15 Levels: ABL1|ABL2|BLK|EPHA2|FGR|FRK|FYN|HCK|KIT|LCK|LYN|PDGFRB|SRC|SRMS|STAT5B|YES1 ...
```

We see, that the targets of vorinostat are exclusively HDACs. In further analyses, we want to check their gene expression change under vorinostat- treatment since we can imagine that an increased inhibition might result in a higher production of HDACs to maintain the balance.

```
Drug_annotation$indication[grep(Drug_annotation$Drug, pattern= "vorinostat")]
```

```
## [1] cutaneous T-cell lymphoma (CTCL)
## 15 Levels:  ...
```

Until now, the T-cell lymphoma is the only cancer-type which indicates vorinostat-ingestion.

Table of content

1. General visualizations * 1.1. Pie chart of tissue types * 1.2.Density plot of general gene expression

2. Creation of Vorinostat-FC-matrix

3. Finding biomarkers * 3.1. Biomarkers received from two-sided T-test * 3.2. Biomarkers received from FC values + 3.2.1. General Biomarkers + 3.2.2. Up and Downregulation of biomarkers * 3.3. Comparison of the differently received biomarkers

4. Visualizations of biomarkers (#anchor) * 4.1. Density plot of biomarker gene expression * 4.2. Barplots of up- and downregulation of biomarkers * 4.3. Visualization of biomarkers with Vulcanoplot

5.Are the targets for vorinostat (HDACs) part of our biomarkers?

6.PCA with general-biomarker-matrix

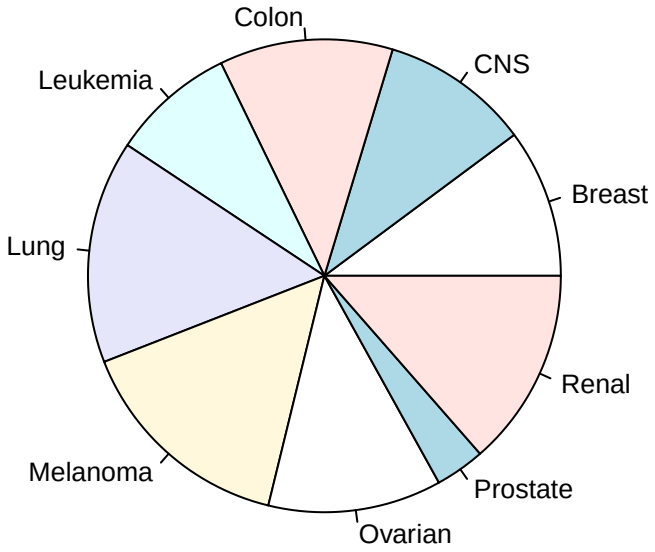
7.Find biomarkers only for leukemia-celllines

1. General visualizations

1.1. Pie chart of tissue types

Which tissue types are we dealing with when we work with the vorinostat-treated celllines? We use tissue information from Metadata and add it to the Treated-matrix, then we select those columns which belong to Vorinostattreatment:

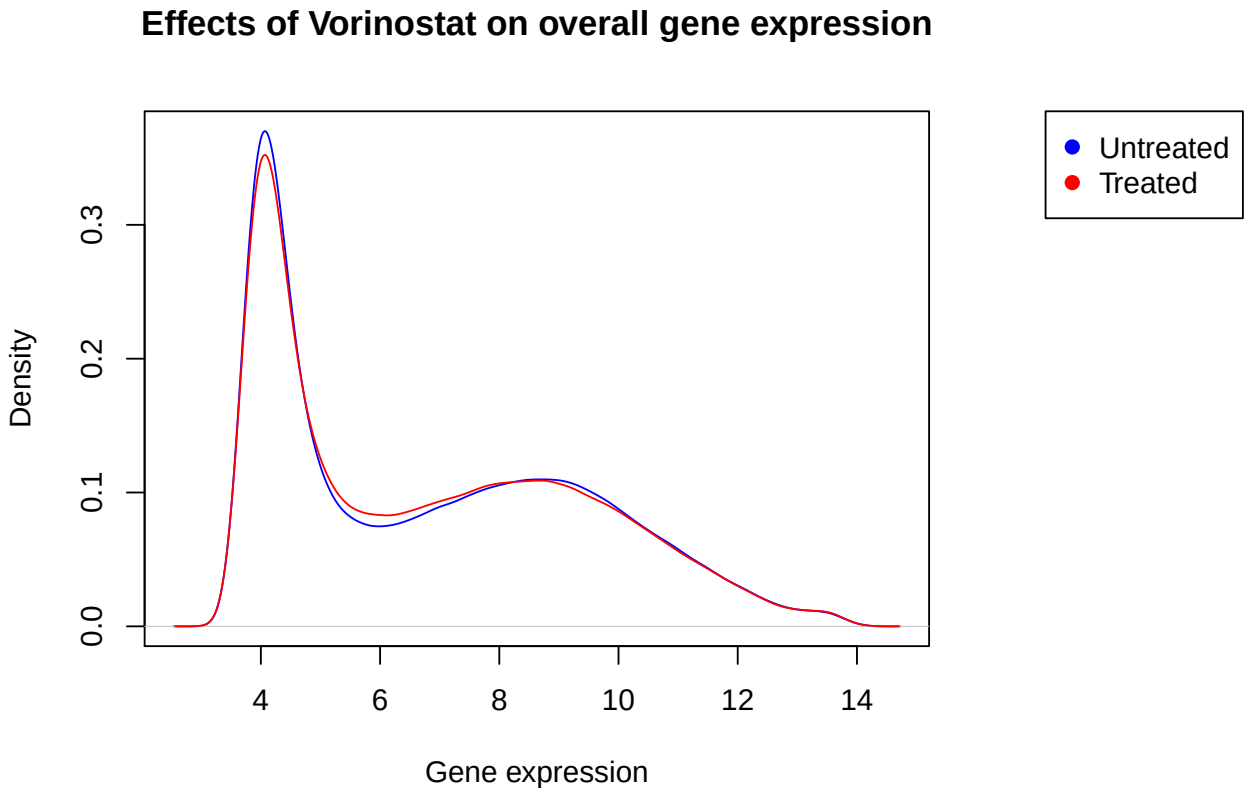
```
Metadata <- as.matrix(Metadata)
Metadatatissue <- Metadata[1:819,"tissue"]
Treatedwithtissue <- rbind(Treated, Metadatatissue)

    TreatedVorinostatcolumns <- grep(pattern = "vorinostat",colnames(Treatedwithtissue))
    TreatedVorinostattissue <- Treatedwithtissue[,TreatedVorinostatcolumns]

#Create a pie chart:

palette(rainbow(9))
pie(table(TreatedVorinostattissue["Metadatatissue",]),
    main="Tissue types of Vorinostat-treated samples",
    radius=1,
    cex=0.8)
```



1.2.Density plot of general gene expression

This plot is done to compare the gene expression of all provided genes with and without vorinostat treatment:

```
TreatedVorinostat <- Treated[,TreatedVorinostatcolumns]

    UntreatedVorinostatcolumns <- grep(pattern = "vorinostat",colnames(Untreated))
UntreatedVorinostat <- Untreated[,UntreatedVorinostatcolumns]

par(mar=c(5, 4, 5, 9))
plot(density(UntreatedVorinostat), col="blue" ,xlab = "Gene expression",
     main = "Effects of Vorinostat on overall gene expression")
lines(density(TreatedVorinostat), col = "red")
legend("topright", inset = c(-0.4,0), legend=c("Untreated", "Treated") , xpd = TRUE,
       pch=19, col = c("blue", "red"))
```



Altough Vorinostats working mechanism as a HDAC inhibitor should lead to a better access to genes, we do not see a higher expression in terms of all genes. We are interested in how this plot changes when we plot only our biomarkers, so we are going to repeat the density plot later when we identified them.

2. Create a Vorinostat-FC-matrix

For the soecific analysis, it is laborious to work with the whole dataset which includes data from all 15 drugs. Consequently, we reduce the data-sets by selecting those celllines that belong to vorinostat:

```
UntreatedVorinostatcolumns <- grep(pattern = "vorinostat",colnames(Untreated))
TreatedVorinostatcolumns <- grep(pattern = "vorinostat",colnames(Treated))
```

We want to keep all genes but only the selected celllines:

```
UntreatedVorinostat <- Untreated[,UntreatedVorinostatcolumns]
TreatedVorinostat <- Treated[,TreatedVorinostatcolumns]

FC <- TreatedVorinostat - UntreatedVorinostat
```

We can check, if FC data are approximately normal distributed by using a qq-plot:

```
qqnorm(FC)
qqline(FC, col= "red")
```

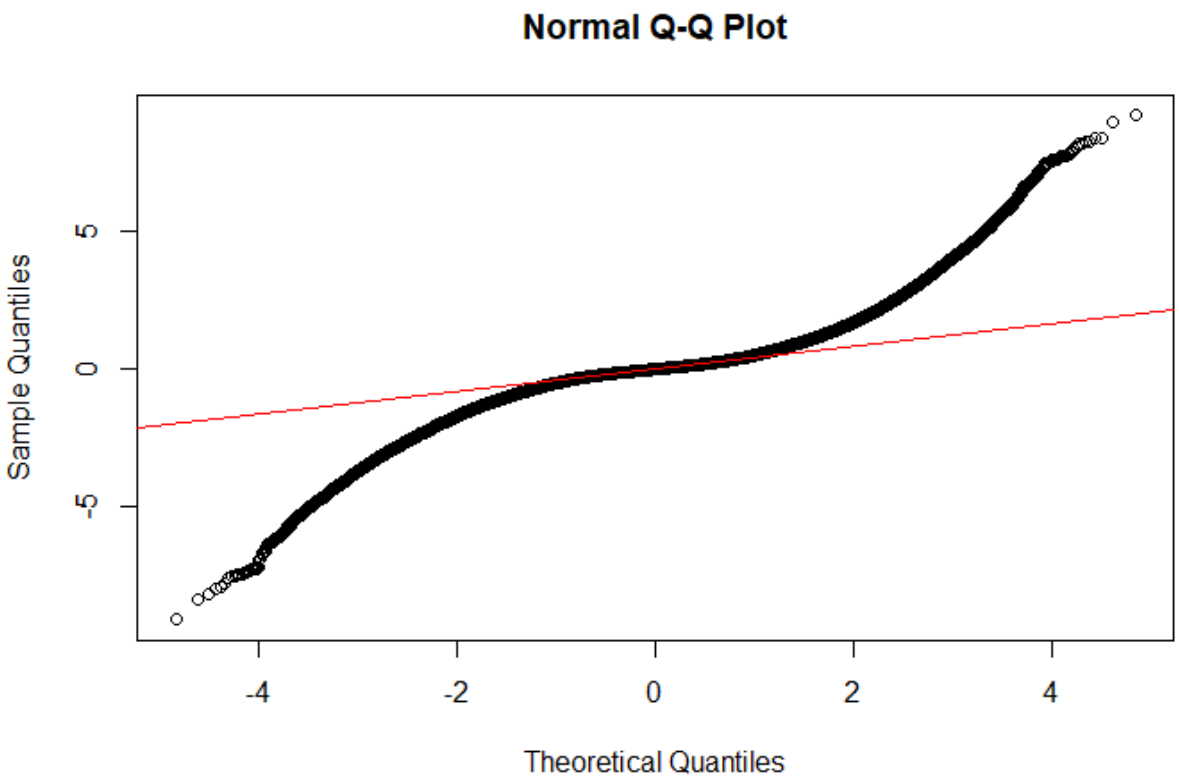

Normal Q-Q Plot
Sample Quantiles
5
0
-5
-4
-2
0
2
4
Theoretical Quantiles

Figure 1:

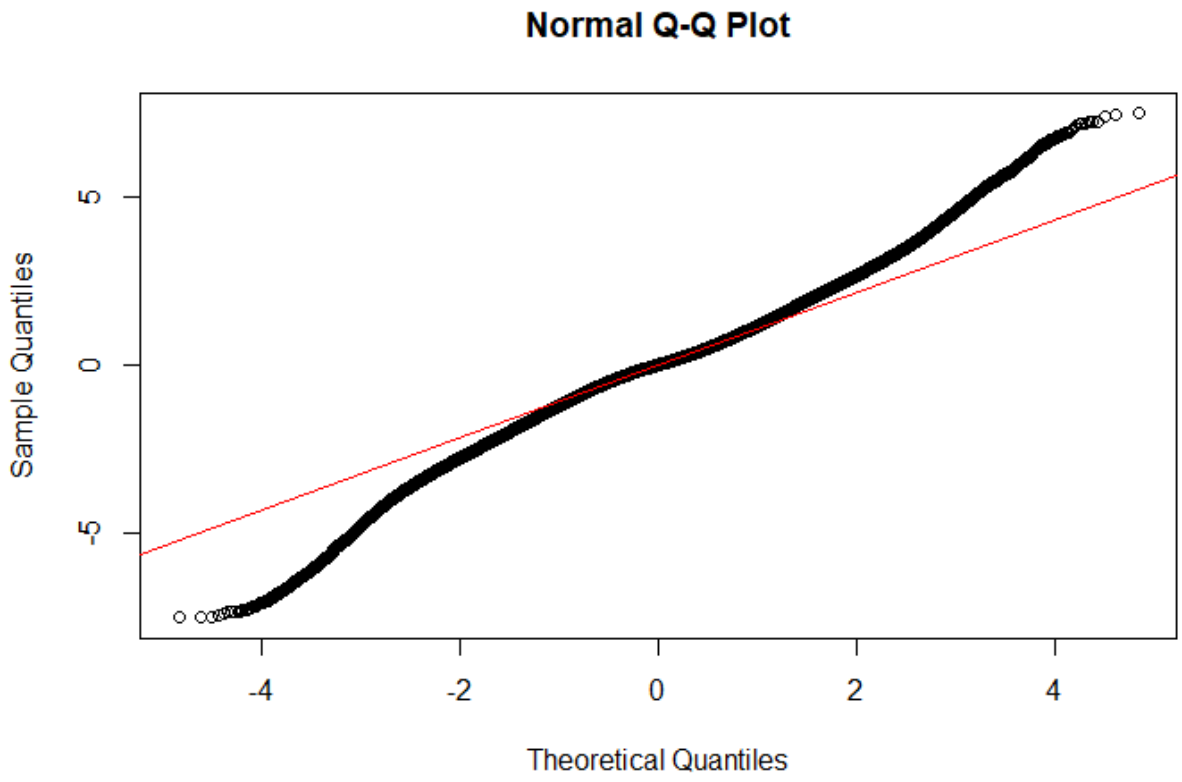


Figure 2:

We see, that we have a heavy tailed distribution.

FC data is normalized with a package (install.packages("BBmisc")):

```
library(BBmisc)
FCnorm <- normalize(FC, method= "scale")
```

```
qqnorm(FCnorm)
qqline(FCnorm, col= "red")
```

After normalization the plot looks more linear than before, altough it is still heavy-tailed.

3. Finding biomarkers

3.1. Biomarkers received from two-sided T-test

The two-sided T-test was performed to find those genes with the smallest p-values, since they change most significantly under drug treatment.

For the t-test, we need to check if genes and celllines are in the same order in both matrices:

```r
identical(rownames(UntreatedVorinostat), rownames(TreatedVorinostat))
```

```
## [1] TRUE
```

-> genes are in the same order

```r
identical(colnames(UntreatedVorinostat), colnames(TreatedVorinostat))
```

```
## [1] FALSE
```

The colnames differ since the the matrices have a different drug dosis We remove the information about the dosis (replace it with nothing):

```r
Untreatedcolnames <- sub(pattern = "_0nM_24h", "", colnames(UntreatedVorinostat))
Treatedcolnames <- sub(pattern = "_5000nM_24h", "", colnames(TreatedVorinostat))
```

Check, if the colnames are equal now:

```r
identical(Untreatedcolnames, Treatedcolnames)
```

```
## [1] TRUE
```

Since the colnames are equal now, matrices do not differ regarding their order of celllines, the dosis is the only difference, thus we can use both matrices to perform a two-sided t-test.

We need to formulate hypotheses:

- **H0 hypothesis**: Gene expression does not change significantly by drug treatment.
- **H1 hypothesis**: Gene expression changes significantly by drug treatment.

For the t-test we create a common matrix for treated and untreated vorinostat data:

```r
VorinostatTotal <- cbind(UntreatedVorinostat, TreatedVorinostat)
```

Define variables for better readability:

```r
 col_untreated = grep ('_0nM',colnames(VorinostatTotal))

 col_treated = grep ('_5000nM',colnames(VorinostatTotal))
```

Perform the t-test by applying the t-test-function on every gene (indicated by 1):

```r
t.test.Vorinostat = apply(VorinostatTotal, 1, function(x) t.test(x[col_untreated],
                                                                 x[col_treated],
                                                                 paired = TRUE,
                                                                 alternative = "two.sided"))
```

To get the p-values:

```
pValues <- apply(VorinostatTotal, 1, function(x) t.test(x[col_untreated],
                                                        x[col_treated],
                                                        paired = TRUE,
                                                        alternative = "two.sided")$p.value)
```

-> gives a mean p-value for each gene.

Next we sort the p-values:

```
sortedpValues <- sort(pValues, decreasing = FALSE)
sortedpValues <- as.matrix(sortedpValues)
```

Now we can add the p-Values as a new column to each gene:

```
VorinostatwithpValues <- cbind(VorinostatTotal, pValues)
```

Select those rows with smallest p.Values to create a matrix with biomarker-genes:

```
VorinostatwithpValues <-  as.data.frame(VorinostatwithpValues)
Biomarkermatrix2sidedtest <- VorinostatwithpValues[VorinostatwithpValues$pValues
                                                   <=  sortedpValues[100,],]
```

Sort biomarkers according to increasing p-Values (most significant p-Values on top of the data frame):

```
Biomarker2sidedtestsorted <- Biomarkermatrix2sidedtest[order
                                                       (Biomarkermatrix2sidedtest$pValues),]
```

Those 100 genes with the smallest p-Value are most significant:

```
biomarkers <- sortedpValues[1:100,]
biomarkers <- as.matrix(biomarkers)
biomarkers.genes = row.names(biomarkers)
```

3.2. Biomarkers received from FC values

3.3.1.General biomarkers

```
FC <- TreatedVorinostat - UntreatedVorinostat

FCVorinostatabs= abs(FC)
FCVorinostatmean <- apply(FCVorinostatabs, 1, mean)

#sort the values to get the 100 largest values:
sortedgeneralbiomarker <- sort(FCVorinostatmean, decreasing = TRUE)
sortedgeneralbiomarker <- as.matrix(sortedgeneralbiomarker)

#select the top 100 general biomarkers:
top100generalbiomarkers = sortedgeneralbiomarker[1:100,]
top100generalbiomarkers <- as.matrix(top100generalbiomarkers)
head(top100generalbiomarkers)
```

```
##                     [,1]
## DHRS2           4.546618
## ABAT            3.415063
## SERPINI1        3.229245
## MIR612///NEAT1 3.170173
## HBA2///HBA1     2.709310
## CLU             2.702957
```

Create a vector with gene names:

```
generalbiomarkergenes = row.names(top100generalbiomarkers)
```

Now we have determined biomarkers via the fold change. However, since we have worked only with absolute values, we cannot say whether our genes are up- or downregulated.

3.2.2. Up- and Downregulation of biomarkers

We determine rowmeans of FC values but unlike before, we do not use absolute values to calculate the averages. Consequently, we get a general overview which genes are mostly up- or downregulated.

```
FC_meanrow= rowMeans(FC)

# look at the absolute values:
FC_meanrow = abs(FC_meanrow)

# sort the FC
sortedFC <- sort(FC_meanrow, decreasing = TRUE)
sortedFC <- as.matrix(sortedFC)

# take the first 100 values for comparison
FC100<- sortedFC[1:100,]
FC100 <- as.matrix(FC100)

# create a vector with genenames
biomarkergenes_noabs = row.names(FC100)
# check if it has the correct length
length(biomarkergenes_noabs)
```

```
## [1] 100
```

We want to compare the top biomarkers with and without using absolute values to determine how much they differ:

```
setequal(biomarkergenes_noabs,generalbiomarkergenes)
```

```
## [1] FALSE
```

How many of the biomarkers are different?

```r
# biomarkers in biomarkergenes_noabs but not in generalbiomarkergenes
diff1= setdiff(biomarkergenes_noabs,generalbiomarkergenes)
length(diff1)
```

```
## [1] 5
```

```r
diff1
```

```
## [1] "TXNDC15"                "AAMDC"
## [3] "LOC100287852///FAM136A" "MEPCE"
## [5] "GINS3"
```

```r
# biomarkers in generalbiomarkergenes but not in biomarkergenes_noabs
diff2= setdiff(generalbiomarkergenes,biomarkergenes_noabs)
length(diff2)
```

```
## [1] 5
```

```r
diff2
```

```
## [1] "AKR1C3"           "TP53"             "ALDOC"
## [4] "EMP3"             "SLC2A14///SLC2A3"
```

We see that only 5 genes differ in the top biomarkers. Nevertheless, it is better to trust the absolute values, as they indicate the largest change in gene expression. So we avoid, that up- or down-regulating effects cancel each other out.

In order to be able to work better with the biomarkers, we creat a matrix with the information whether the genes are up or down regulated.

```r
FCVorinostatmean_noabs= rowMeans(FC)

#Matrix with information about general up- or downregulation:
FCVorinostatmean_noabs_matrix <- as.matrix(FCVorinostatmean_noabs)

#loop, which adds those biomarkergenes to a vector, which are mostly upregulated:
i=1
Upregulated <- c()
while(i<=100) {
  if(FCVorinostatmean_noabs_matrix[generalbiomarkergenes[i],1] >0) {
    Upregulated <- union(Upregulated, generalbiomarkergenes[i])
  }
  i=i+1}

#Create a vector that includes for each gene sequentially if it is up- or downregulated:
i=1
Generalchange <- c()
while(i<=100) {
  if(FCVorinostatmean_noabs_matrix[generalbiomarkergenes[i],1] >0) {
    Generalchange <- append(Generalchange, "Up")
  } else {
    Generalchange <- append(Generalchange, "Down")
  }
  i=i+1}
```

Bind the information about up/downregulation as a new column to the biomarkermatrix:

```
top100generalbiomarkers_withUporDown <- cbind(top100generalbiomarkers, Generalchange)
# rename
colnames(top100generalbiomarkers_withUporDown)[1] <- "FCmean"
```

3.3. Comparison of the differently received biomarkers

Now we have determined biomarkers via the 2 sided T-test and want to investigate their FC values now since our biomarkers should have the largest change in gene expression. Do we have exactly the same biomarkers?

```
setequal(generalbiomarkergenes,biomarkers.genes)
```

```
## [1] FALSE
```

How many of the biomarkers are different?

```
diff= setdiff(generalbiomarkergenes,biomarkers.genes)
length(diff)
```

```
## [1] 88
```

We want to compare p.Values and FC values in a new matrix which contains both values:

```
pV_FC= cbind(FC_meanrow,sortedpValues)
head(pV_FC)
```

```
##        FC_meanrow
## A1CF   0.12282564 6.133401e-30
## A2M    0.09263300 1.240342e-29
## A4GALT 0.16032849 2.030467e-29
## A4GNT  0.05774256 4.011786e-29
## AAAS   0.10233892 7.143975e-29
## AACS   0.03026825 2.159588e-28
```

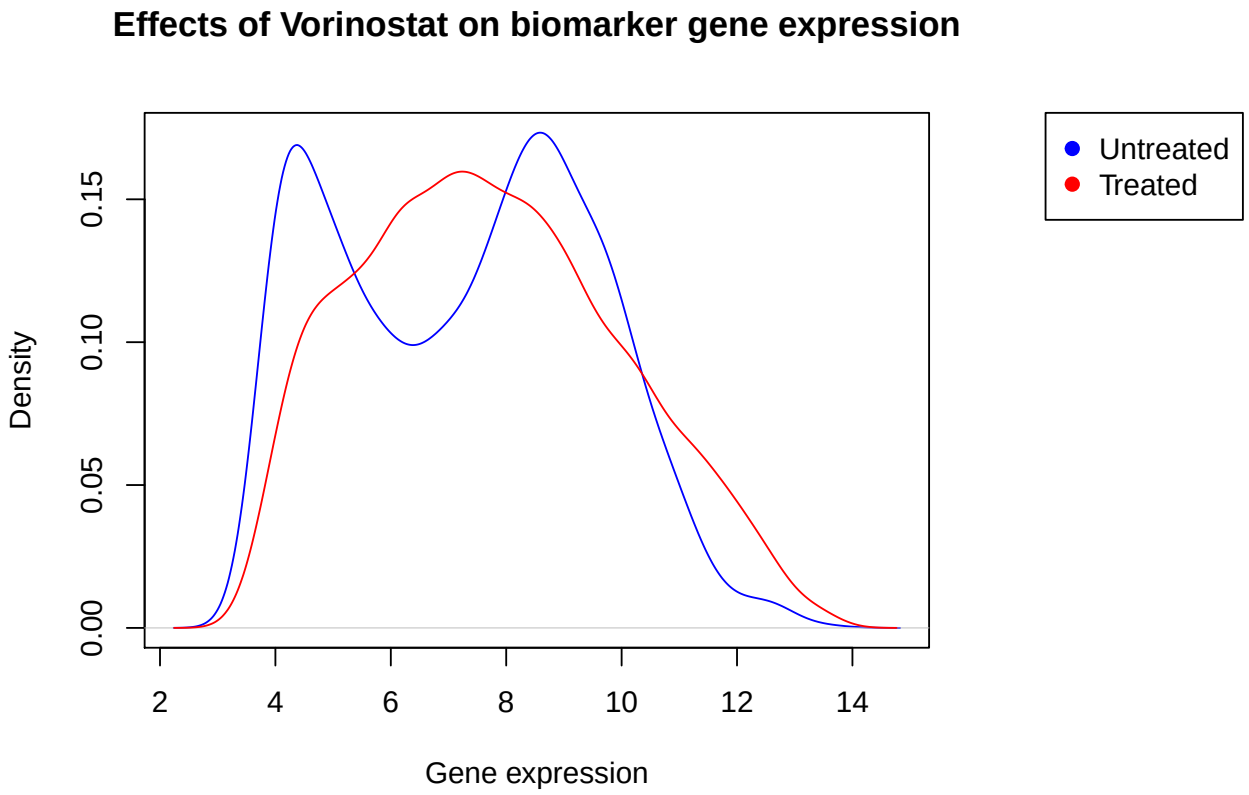
Now we know that p.Value and FC value do not deliver the same biomarkers. In fact, 88 out of 100 biomarkers are different. If we compare the values directly, we see that a low p.value does not correlate with a high FC value. This is because small values are also assigned small p.values by the T-test and the result of the test is therefore falsified. For this reason, we cannot trust the biomarkers from the two-sided T-test and will trust the biomarkers we received via the **fold change** values.

4. Visualizations of biomarkers

4.1. Density plot of biomarker gene expression

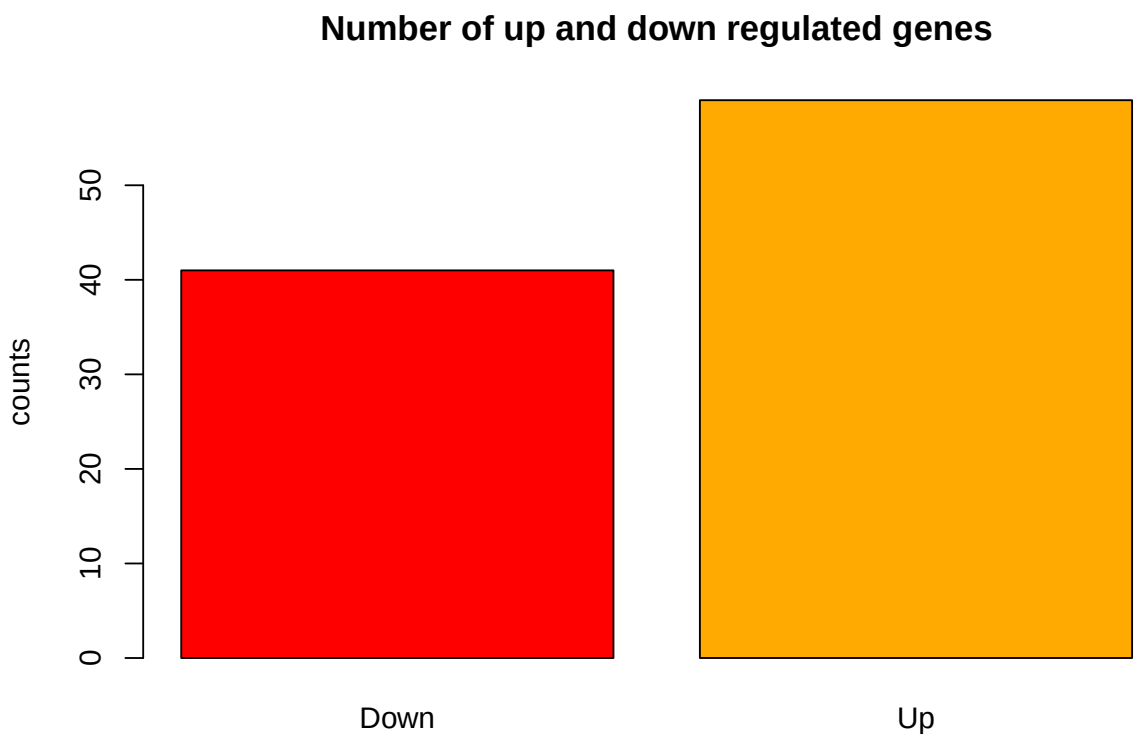
Now we can repeat the the **gene-expression density plot** from the beginning of our specific analysis with our biomarkers. We can see, that in terms of our identified genes Vorinostat leads to a clearly altered expression:

```
Treated_Vorinostat_biomarker <- Treated[generalbiomarkergenes,TreatedVorinostatcolumns]
Untreated_Vorinostat_biomarker <- Untreated[generalbiomarkergenes,UntreatedVorinostatcolumns]

par(mar=c(5, 4, 5, 9))
plot(density(Untreated_Vorinostat_biomarker),col="blue" ,xlab = "Gene expression",
     main = "Effects of Vorinostat on biomarker gene expression")

lines(density(Treated_Vorinostat_biomarker), col = "red")

legend("topright", inset = c(-0.4,0), legend=c("Untreated", "Treated") , xpd = TRUE,
       pch=19, col = c("blue", "red"))
```



4.2. Barplots of up- and downregulation of biomarkers

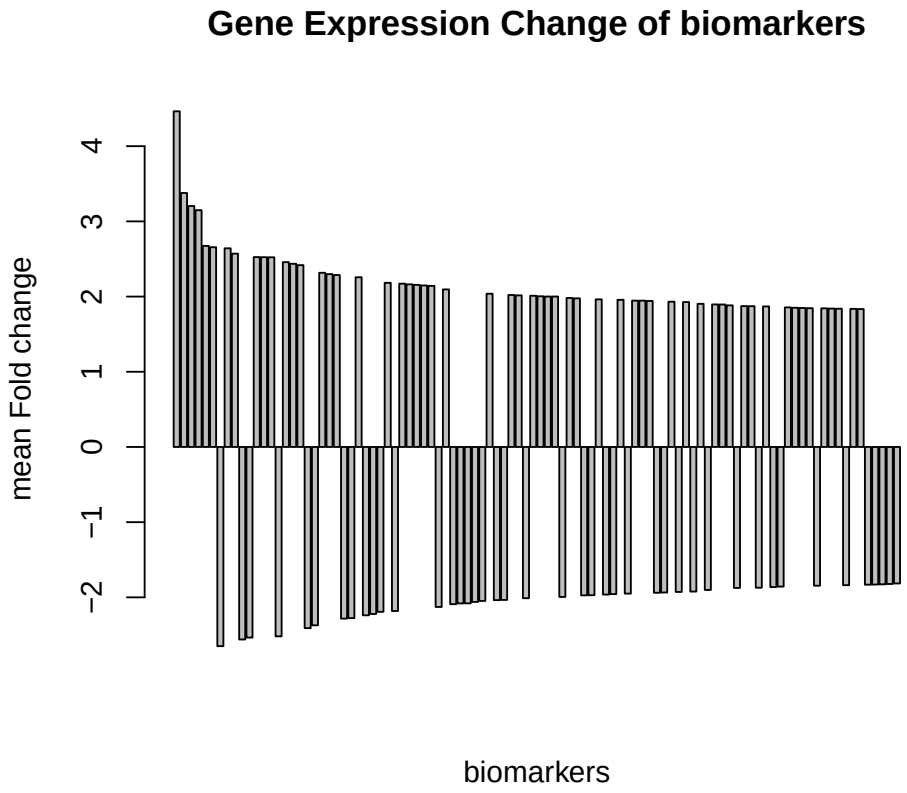
```
top100generalbiomarkers_withUporDown=as.data.frame(top100generalbiomarkers_withUporDown)
change=top100generalbiomarkers_withUporDown$Generalchange
col=palette(rainbow(2))
barplot(table(change), ylab="counts", main="Number of up and down regulated genes", col=col)
```



We see that we have approximately the same number of up and down regulated genes.

Another option to visualize the up- and downregulation of the biomarkers:

```
FC_meanrow= rowMeans(FC)
FC_abs= abs(FC_meanrow)

FC_both= cbind(FC_meanrow,FC_abs)
FC_both=as.data.frame(FC_both)

# order this matrix
FC_both_sorted <- FC_both[order(FC_both$FC_abs, decreasing = TRUE),]

# FC values of biomarkers
# take the first 100 of the sorted matrix, should be the same as un biomarkers_FC
biomarkers_FC_values = FC_both_sorted[1:100,]
# remove the absolute values
biomarkers_FC_values <- subset( biomarkers_FC_values, select = -FC_abs)
biomarkers_FC_values = as.data.frame(biomarkers_FC_values)

par(mar=c(5, 4, 5, 9))
barplot( height= biomarkers_FC_values$FC_meanrow,
         main= "Gene Expression Change of biomarkers",
         ylab="mean Fold change", xlab="biomarkers")
```



We see that most values are very similar to each other and that we have approximately the same number of genes regulated up and down as we already have visualized above.

4.3.Visualization of biomarkers with Vulcanoplot

First we perform a one-sample t.test with normalized FC data:

```
T.testFC <- apply(FC, 1, t.test)
```

Save the results in a data frame:

```
TResults <- lapply(T.testFC, function(.tres) {

  data.frame(t.value=.tres[1],dfs=.tres[2],conf.int1=.tres$conf.int[1],conf.int2=
               .tres$conf.int[2],p.value=.tres[3])
})

Tfinalresults <- do.call(rbind, TResults)
```

For the vulcano plot, a package needs to be installed: if (!requireNamespace(‘BiocManager’, quietly = TRUE)) install.packages(‘BiocManager’) BiocManager::install(‘EnhancedVolcano’)

```
library(EnhancedVolcano)
```

Since we are interested in general biomarkers, we use the mean FC of the columns:

```
log2FC <- apply(FC, 1, mean)
```

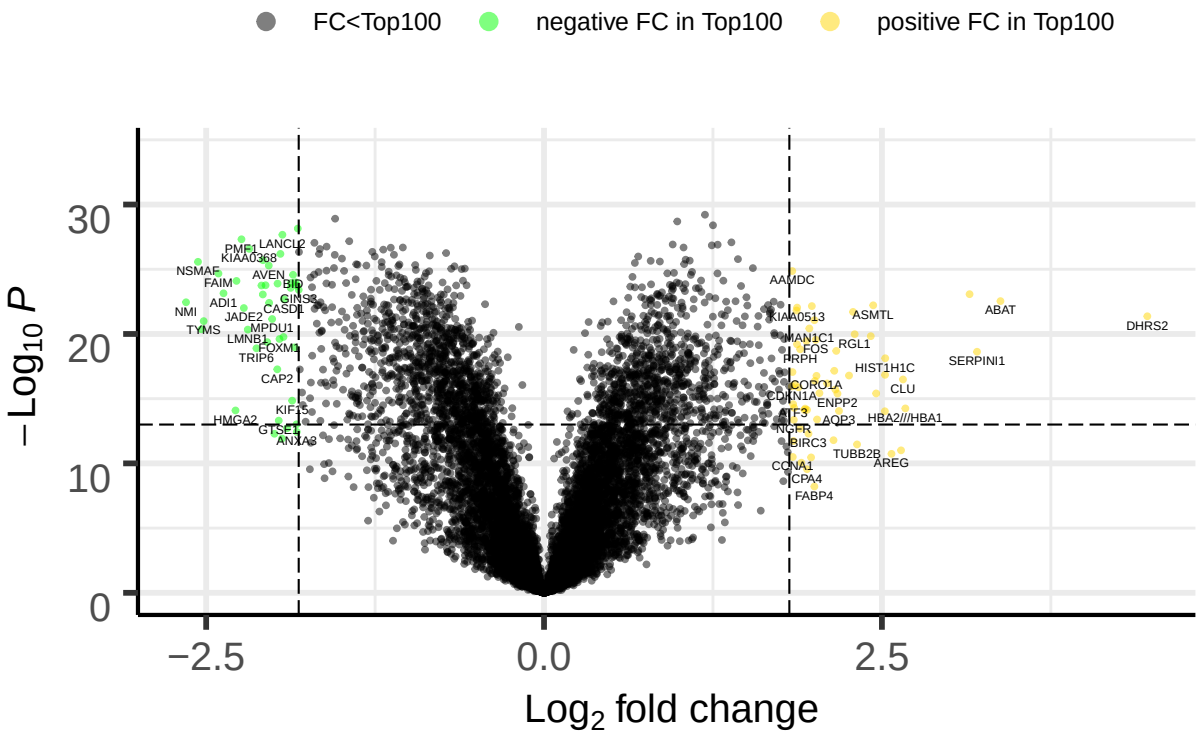
Add the FCvalues as a new column to the data frame containing the results of the t.test:

```
FCandpvalues <- cbind(Tfinalresults, log2FC)
```

Create a Vulcanoplot:

First, all genes are assigned to the color black. Subsequently, those genes with a FC equal or larger than the smallest FC of the top 100 FC-values are colored in gold. Those, which exceed the same limit in the negative range are assigned to the color green.

```
keyvals <- rep('black', nrow(FCandpvalues))
names(keyvals) <- rep('FC<Top100', nrow(FCandpvalues))

keyvals[which(FCandpvalues$log2FC >= min(FC100))] <- 'gold'
names(keyvals)[which(FCandpvalues$log2FC >= min(FC100))] <- 'positive FC in Top100'

keyvals[which(FCandpvalues$log2FC <= -min(FC100))] <- 'green'
names(keyvals)[which(FCandpvalues$log2FC <= -min(FC100))] <- 'negative FC in Top100'

EnhancedVolcano(FCandpvalues,
                lab = rownames(FCandpvalues),
                x = 'log2FC',
                y = 'p.value',
                title = 'Significance versus fold change of all genes',
                selectLab = biomarkergenes_noabs,
                transcriptLabSize = 1.8,
                colOverride = keyvals,
                pCutoff = 10e-14,
                FCcutoff = min(FC100)
              )
```



Now we want to further investigate our biomarkers.

5. Are the targets for vorinostat (HDACs) part of our biomarkers?

```
# search for HDAC in biomarkers
target <- grep(pattern = "HDAC",colnames(top100generalbiomarkers))
length(target)
```

```
## [1] 0
```

In drug annotation, HDAC genes are mentioned as target of vorinostat, but these genes are not in our defined biomarkers, so we want to check their FC values:

We create a vector with targetnames as strings, because otherwise we have a problem with spacer |

```
target_vorinostat=c("HDAC1","HDAC10","HDAC11","HDAC2","HDAC3","HDAC5","HDAC6",
                    "HDAC8","HDAC9")

### find FC values of targets

# creat FC data
FC_meanrow = as.data.frame(FC_meanrow)
genes = row.names(FC_meanrow)
FC_new=cbind(FC_meanrow,genes)
```

```
FC_target = as.data.frame(filter(FC_new, genes %in% target_vorinostat))
FC_target
```

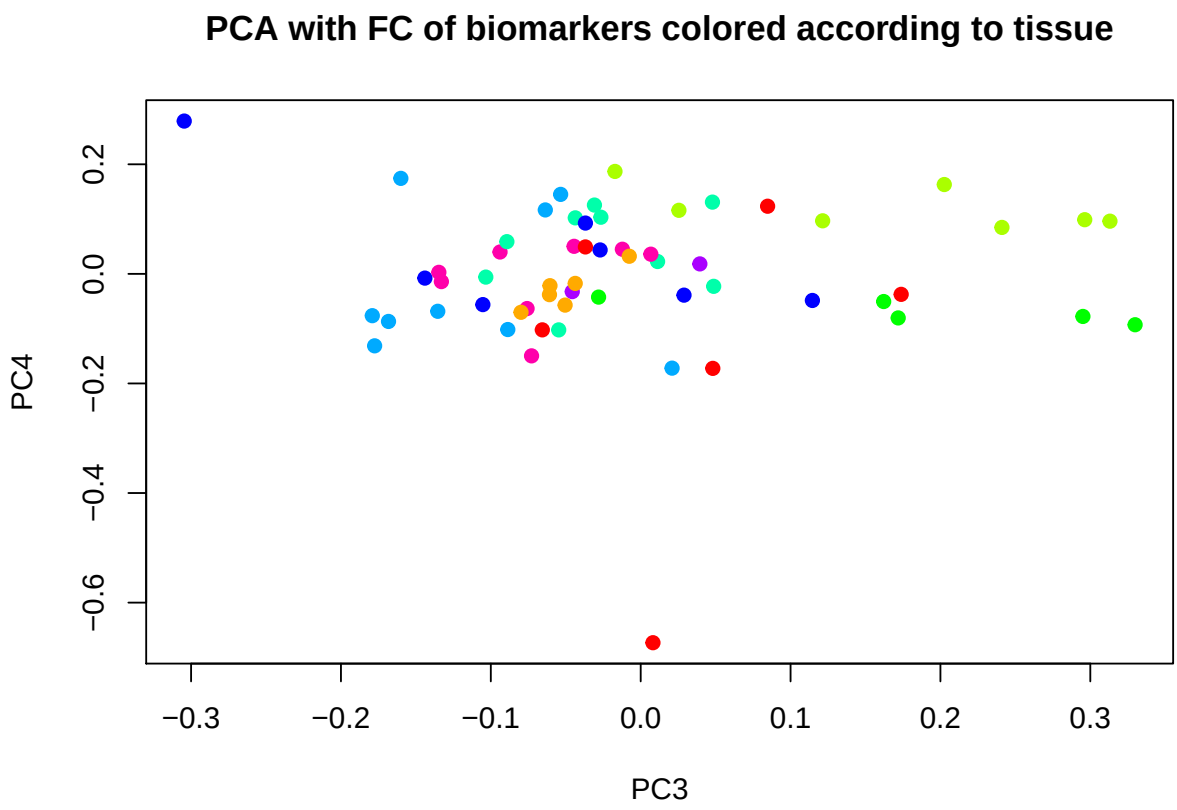
```
##    FC_meanrow  genes
## 1  0.28492740  HDAC1
## 2  0.07724018 HDAC11
## 3 -0.06489651  HDAC2
## 4  1.04792610  HDAC3
## 5  1.22216759  HDAC5
## 6  0.28711822  HDAC6
## 7 -0.34452399  HDAC9
```

The changes in gene expression are low. Most of the HDAC genes are up regulated insted of down regulated. The cell might want to compensate the inhibition of the enzymes by their increased production.

6. PCA with general-biomarker-matrix

Select only those genes from the FC matrix which were identified as biomarkers:

```
FCbiomarkers <- FC[generalbiomarkergenes,]

#Execute the PCA:
FCbiomarkers.pca = prcomp(FCbiomarkers, center=T, scale. = T)

#For coloring according to tissue type we use information from Metadata:
Metadata <- as.data.frame(Metadata)

#We only need the tissueinformation of those samples treated with Vorinostat.
#Since Metadata includes them twice, once with and once without drug treatment,
#we select only the Treated samplenames:
Vorinostatsamples <- grep(Metadata$sample, pattern= "vorinostat_5000nM")
Metadatavorinostattissue <- Metadata[Vorinostatsamples,"tissue"]

#Add the tissueinformation as a new row to the Biomarkermatrix:
FCbiomarkerswithtissue <- rbind(FCbiomarkers, Metadatavorinostattissue)

#save tissue information as factors so it can be used for coloring:
vorinostattissuefactor <- as.factor(FCbiomarkerswithtissue["Metadatavorinostattissue",])

#9 different tissue types:
palette(rainbow(9))

#PC 3 & 4 group the tissues best but a clear grouping is not shown as well:
plot(FCbiomarkers.pca$rotation[, 3], FCbiomarkers.pca$rotation[, 4],
     col = vorinostattissuefactor, pch = 19, xlab = "PC3", ylab = "PC4",
     main = "PCA with FC of biomarkers colored according to tissue")

#Add a legend to the plot:
levels <- as.factor(levels(vorinostattissuefactor))
legend("topright", inset = c(-0.4,0),levels(Metadatavorinostattissue),
       xpd = TRUE, pch=19, col = levels)
```




```
par(mar=c(5, 4, 5, 9))
```

7. Find biomarkers only for leukemia-celllines

Since we found in the beginning, that Vorinostat is used against T-cell lymphomas, we are interested in how Vorinostat influences especially the leukemia-celllines.

```
#We only need the tissueinformation of those samples treated with Vorinostat
#Since Metadata includes them twice, once with and once without drug treatment,
#we select only the Treated samplenames:
Vorinostatsamples <- grep(Metadata$sample, pattern= "vorinostat_5000nM")
Metadatavorinostattissue <- Metadata[Vorinostatsamples,"tissue"]

#Add the tissueinformation as a new row to the FCVorinostat-matrix:
FCwithtissue <- rbind(FC, Metadatavorinostattissue)

#select only those samples/celllines which belong to Leukemia:
FCwithtissue <- as.data.frame(FCwithtissue)
Leukemiasamples <- colnames(FCwithtissue[Metadatavorinostattissue== "Leukemia"])
FCLeukemia <- FCwithtissue[,Leukemiasamples]

#Now we do not need the tissue information any longer:
FCLeukemia <- FCLeukemia[-13300,]
```

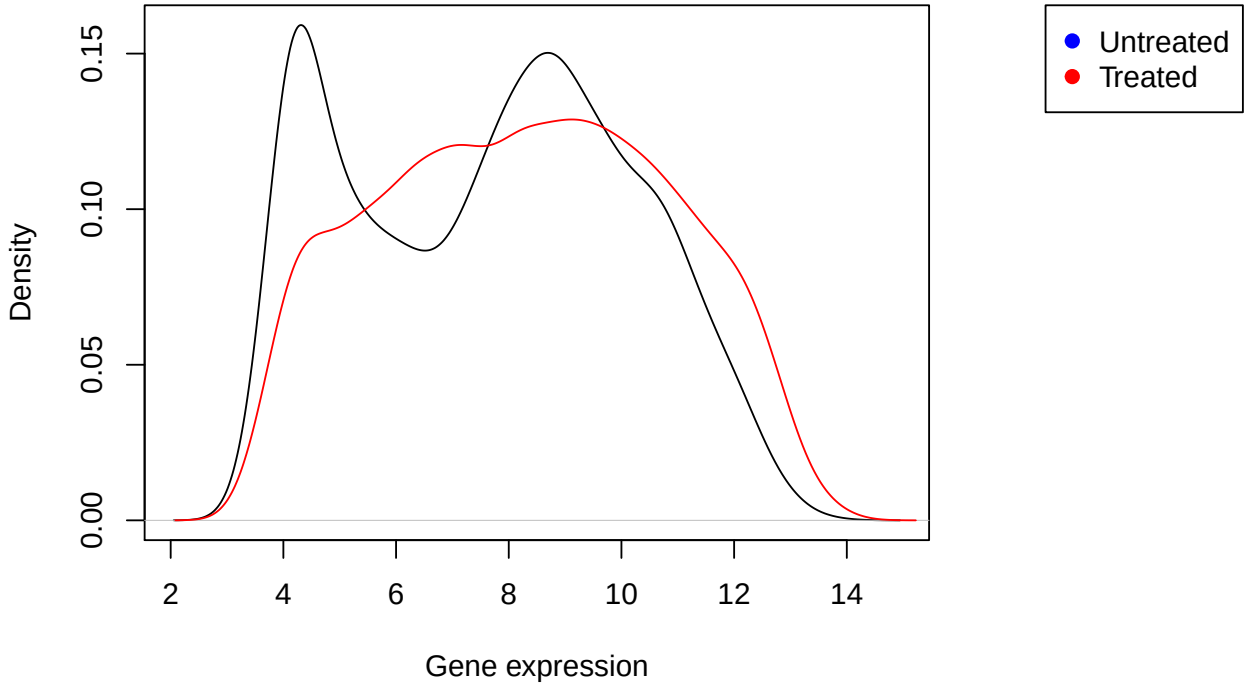
```r
#We want the FC-mean for each gene:
FCLeukemiaabs <- abs(FCLeukemia)
FCLeukemiameans <- apply(FCLeukemiaabs,1 ,mean)

#Sort the biomarkers that those with the biggest FC are on top:
sortedFCleukemiamean <- sort(FCLeukemiameans, decreasing = TRUE)
sortedFCleukemiamean <- as.matrix(sortedFCleukemiamean)

# take the first 100 as Leukemia-biomarkers:
biomarkersLeukemia = sortedFCleukemiamean[1:100,]
biomarkersLeukemia <- as.matrix(biomarkersLeukemia)

# create a vector with gene names of Leukemia-biomarkers:
biomarkersLeukemiaGenes = row.names(biomarkersLeukemia)


#Comparison with general biomarkers:
intersect(biomarkersLeukemiaGenes, generalbiomarkergenes)
```

```
##  [1] "DHRS2"                 "MIR612///NEAT1"
##  [3] "ID2B///ID2"            "PSMB10"
##  [5] "HIST1H2BJ///HIST1H2BG" "SERPINI1"
##  [7] "ADI1"                  "CRISPLD2"
##  [9] "TUFT1"                 "CDKN1A"
## [11] "NMI"                   "NEU1"
## [13] "AKR1C3"                "TNFSF9"
## [15] "UTP20"                 "FKBP1B"
## [17] "SYT11"                 "RCN1"
## [19] "NRGN"                  "HIST1H2AE"
## [21] "ASMTL"                 "NSMAF"
## [23] "MAN1C1"                "ZMYND11"
## [25] "ENPP2"                 "LANCL2"
## [27] "RGL1"                  "PMF1"
## [29] "ABAT"                  "PHF2"
## [31] "TMEM35B///ZMYM6"       "KIAA0368"
## [33] "CLU"                   "BID"
## [35] "MPDU1"                 "JADE2"
## [37] "TUBB4A"                "TUBB2B"
```

```r
biomarkersLeukemiaGenes[biomarkersLeukemiaGenes == generalbiomarkergenes]
```

```
## [1] "DHRS2"
```

```r
Treated_Leukemiabiomarker <- Treated[biomarkersLeukemiaGenes ,TreatedVorinostatcolumns]
Untreated_Leukemiabiomarker <- Untreated[biomarkersLeukemiaGenes,UntreatedVorinostatcolumns]

par(mar=c(5, 4, 5, 9))
plot(density(Untreated_Leukemiabiomarker) ,xlab = "Gene expression",
     main = "Effects of Vorinostat on Leukemiabiomarker gene expression")
lines(density(Treated_Leukemiabiomarker), col = "red")
legend("topright", inset = c(-0.4,0), legend=c("Untreated", "Treated") , xpd = TRUE,
       pch=19, col = c("blue", "red"))
```



The plot looks quite similar to the plot before which contained the general biomarkers. Since the biomarkers of leukemia are partly the same as the general biomarkers(38 out of 100 biomarkers), the similarity of the plots is not surprising.