

Project_logboek_Genexpressie

Mark van de Streek

2023-04-21

Contents

Gen expressie Logboek	2
Het onderzoek	2
Verdeling van de groepen	2
1. Verkenning van de data	4
1.1 Inladen van de data	4
1.2 Eerste kijk op data	5
1.3 Verdeling van de data	6
1.4 Normalisatie van de data en de afstand tussen de samples	8
2 Differentieel tot expressie gebrachte genen (DEG's)	12
2.1 Voorbewerking van de data	12
2.2 Fold Change Value (FC)	13
2.3 DEG analyse met behulp van R-paketten	14
3. Analyse en Visualisaties van de DEG's	18
3.1 Volcano plot	18
3.2 Venn Diagram	20
3.3 Pad analyse	22
Conclusie en discussie	24
Referenties	25

Gen expressie Logboek

Het doel van dit logboek is om het onderzoek wat onderstaand benoemd wordt te reproduceren. Verder wordt er gekeken of er nog meer biologische processen opvallen.

Het onderzoek gaat over het roken van sigaretten en de elektronische sigaret, ook wel de vape genoemd. In het onderzoek is gebruikt gemaakt van RNA-seq data. Er is dus gekeken naar expressie van bepaalde genen. Om precies te zijn is gekeken naar genen die aanwezig zijn in leukocyten (witte bloedcellen).

Het onderzoek

Er is gekeken naar RNA expressie van de volgende groepen mensen:

- Vapers (mensen die een elektronische sigaret roken)
- Mensen die de traditionele sigaret roken
- Mensen die geen tabaksproducten gebruiken

De mensen die traditionele sigaretten roken, worden in het onderzoek de ‘smokers’ groep genoemd. De mensen die niet roken worden ook wel de controle-groep genoemd.

Om mensen te werven voor het onderzoek werd er om te beginnen een online vragenlijst gemaakt. Vervolgens is er voor deze vragenlijst online reclame gemaakt. Denk hierbij aan Reddit en Twitter, etc. Aan de hand van de deze vragenlijst werden de mensen die geschikt leken voor het onderzoek uitgenodigd voor een gesprek. In dit gesprek werden nog een aantal vragen en checklists gemaakt, om er zeker van te zijn dat de persoon geschikt is voor het onderzoek.

Uit de resultaten bleek dat de mensen die op dit moment Vapen, maar vroeger niet rookten, een significant andere genregulatie hebben ten opzichte van mensen uit de controle groep. Ook mitochondriale genen en immuunresponsgenen waren significant ontregeld.

Verdeling van de groepen

In alle verschillende groepen waren een aantal mensen die aan het onderzoek hebben deelgenomen. Onderstaand de aantallen in een tabel.

Table 1: Aantal mensen per groep

Vapers	Traditionele rokers	Niet-rokers
37	22	23

Er werd bij de deelnemers ook gekeken naar geslacht, leeftijd, etniciteit, ras en ook of de mensen goed konden lezen/schrijven in het Engels. Het onderzoek vond plaats in Los Angeles, aan de University of Southern California.

De opzet van het onderzoek was om de verschillen tussen alle drie de groepen duidelijk weer te geven. In vele eerdere onderzoeken werden vaak deelnemers gekozen die vapen, maar vaak met een geschiedenis in andere rookgewoonten. Er namen dus mensen deel aan het onderzoek die eerder traditionele sigaretten rookten en op latere tijd zijn overgestapt naar elektronische sigaretten.

De groep die werd geëindigd als vapers in dit onderzoek heeft in de 6 maanden voor het onderzoek geen traditionele sigaret gebruikt. De onderzoekers hebben bewust voor een periode van ‘maar’ 6 maanden gekozen, omdat de elektronische sigaret nog niet zo heel lang bestaat. door het kiezen van een periode van 6 maanden, zijn er meer mensen die deel kunnen nemen aan het onderzoek.

Met behulp van bioinformatische bevindingen zijn in dit onderzoek onderscheid gemaakt tussen gezonde volwassen vapers, met en zonder geschiedenis van roken, en ‘exclusieve’ sigaretten rokers. Ook is specifiek de expressie van genen in rokers (zowel e-sigaret als traditionele rokers) vergeleken met niet-rokers.

1. Verkenning van de data

In de eerste sectie worden de verkennende data analyses uitgevoerd. Er wordt bijvoorbeeld gekeken naar de verdeling van de data, maar ook naar de kwaliteit. In het laatste gedeelte van deze sectie wordt een normalisatie uitgevoerd.

1.1 Inladen van de data

Onderstaand wordt de data van het bestand ingeladen. In het bestand staan de ongefilterde ruwe counts van de genexpressie. Helaas is er in het bestand nog niet duidelijk welke samples precies bij welke groep horen. Vandaar dat er nog een aantal stappen moeten worden uitgevoerd om de juiste namen bij de juiste samples te krijgen. Om dit probleem op te lossen kijken naar het GEO-id. Met dit ID kunnen we de juiste namen ophalen uit een object.

```
data <- read.table(  
  "data/GSE169757_Partek_Hum_BC_RNA_Jan2018_UnfilteredCounts_82.txt",  
  header = T,  
  sep = "\t")  
  
# In de eerste kolom van het bestand staat alleen een index,  
# We willen hier de gennamen hebben  
row.names(data) <- data$gene_name  
  
Sys.setenv(VROOM_CONNECTION_SIZE=50007200)  
  
# Opzoeken van het ID en defineren in een object  
gse <- getGEO("GSE169757")  
  
## Found 1 file(s)  
  
## GSE169757_series_matrix.txt.gz  
  
## Rows: 0 Columns: 83-- Column specification -----  
## Delimiter: "\t"  
## chr (83): ID_REF, GSM5213644, GSM5213645, GSM5213646, GSM5213647, GSM5213648...  
## i Use 'spec()' to retrieve the full column specification for this data.  
## i Specify the column types or set 'show_col_types = FALSE' to quiet this message.File stored at:  
## /tmp/Rtmp1JVLvx/GPL18573.soft  
  
# Defineren van de specifieke kolom met alle namen  
gse_groups_column <- gse[[1]]@phenoData@data$characteristics_ch1.1  
  
# Lege vector om de uiteindelijke namen in op te slaan  
all_modified_group_names <- c()  
  
# Het door lopen van alle kolomnamen en het opzoeken van de namen.  
# Allereerst worden de indexen verkregen van alle namen  
# Met die indexen worden de juiste namen eruit gehaald.  
for (name in names(data[, 9:ncol(data)])) {  
  name <- gsub("\\.", "-", name)  
  all_group_names <- gse_groups_column[which(gse[[1]]@phenoData@data$title %in% name)]
```

```

group_names <- strsplit(all_group_names, " : ")[[1]][2]
all_modified_group_names <- c(all_modified_group_names, group_names)
}

# Uniek maken door een getal erachter te zetten.
unique_group_names <- make.unique(all_modified_group_names, sep = "_")

# Het veranderen van de kolom namen in de data
names(data)[9:ncol(data)] <- unique_group_names

```

1.2 Eerste kijk op data

Nu de data goed is ingeladen met alle juiste namen, kunnen we kijken hoe het er precies uit ziet. Ook kunnen we kijken wat het precieze aantal rijen en kolommen zijn, ook wel de dimenties.

```
pander(head(data[1:4, c(1, 2, 8, 9, 10, 11)]))
```

Table 2: Table continues below

	Chromosome	Start	havana_gene	Control	Smoker
5S_rRNA	4	67439992	—	2	2.602
7SK	6	5.3e+07	—	466	73.82
A1BG	19	58345178	OTTHUMG00000183507.2	324.4	96.5
A1BG-AS1	19	58347751	OTTHUMG00000183508.1	286.5	81.92

	Control_1
5S_rRNA	5
7SK	357.3
A1BG	258.9
A1BG-AS1	181.3

In de dataset zijn de volgende kolommen aanwezig: chromosoom, start- en stop-positie, streng, gen symbool/gen naam, havana gen en natuurlijk de samples. Van elke sample is dus meer informatie op te zoeken dan simpelweg alleen de telwaarde. De havana gen kolom is een kolom die unieke waarden bevat die verwijzen naar een specifiek gen, dat op een specifieke locatie op het menselijk genoom ligt. De waarden zijn geannoteerd door het HAVANA-project.

```

dims <- dim(data)
sprintf("Aantal rijen: %.f en aantal kolommen: %.f", dims[1], dims[2])

## [1] "Aantal rijen: 38921 en aantal kolommen: 90"

```

We weten nu hoe de data eruit ziet en hoe groot het precies is. Vanaf kolom nummer negen beginnen de ‘samples’. De samples staan dus niet onder elkaar, maar naast elkaar.

Classificatie van variabelen Omdat we niet de hele tijd willen opzoeken welke kolomnummers bij welke groep hoeren, gaan we dit defineren. Op deze manier kunnen we makkelijk een variabel invullen om vervolgens snel alle data van de bijkomende groep te krijgen.

```
smoker <- grep("Smoker", names(data))
vaper <- grep("Vaper", names(data))
control <- grep("Control", names(data))
```

Grep() geeft de nummers van kolommen terug waar het woord in voorkomt.

1.3 Verdeling van de data

De groepen zijn nu geclasseerd en we kunnen kijken hoe de data is verdeeld. We kunnen dit duidelijk weergeven op twee manieren, een boxplot en een dichtheidsplot.

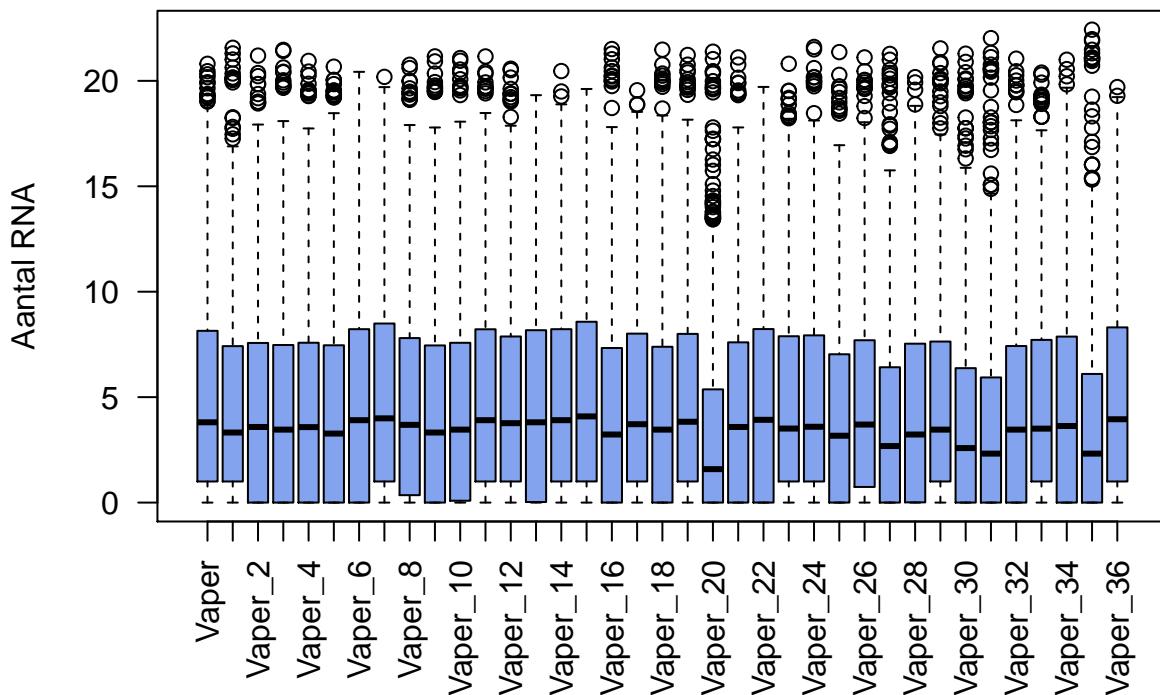
1.3.1 Boxplot Allereerst een boxplot. Als de boxen/strepen van de figuren allemaal rond één lijn zitten, is de data dicht bij elkaar. Een probleem van de ‘ruwe data’ is de hoeveelheid nullen. Deze nullen ‘trekken’ de andere waarden behoorlijk naar beneden. De boxplot heeft dan geen details meer en zegt dan niks meer.

Om dit probleem op te lossen, moeten we alle waarden met 0 ‘wegfilteren’. Dit is makkelijk te doen met een log2 transformatie. Het bereik tussen de samples wordt op deze manier kleiner. De waarden met 0 domineren minder in de data.

Onderstaand de boxplot van de vaper groep, met getransformeerde data.

```
boxplot(log2(data[vaper] + 1), las = 2, col = "#83A3EE",
        main = "log2 transformatie van de verdeling van de vaper groep",
        ylab = "Aantal RNA")
```

log2 transformatie van de verdeling van de vaper groep

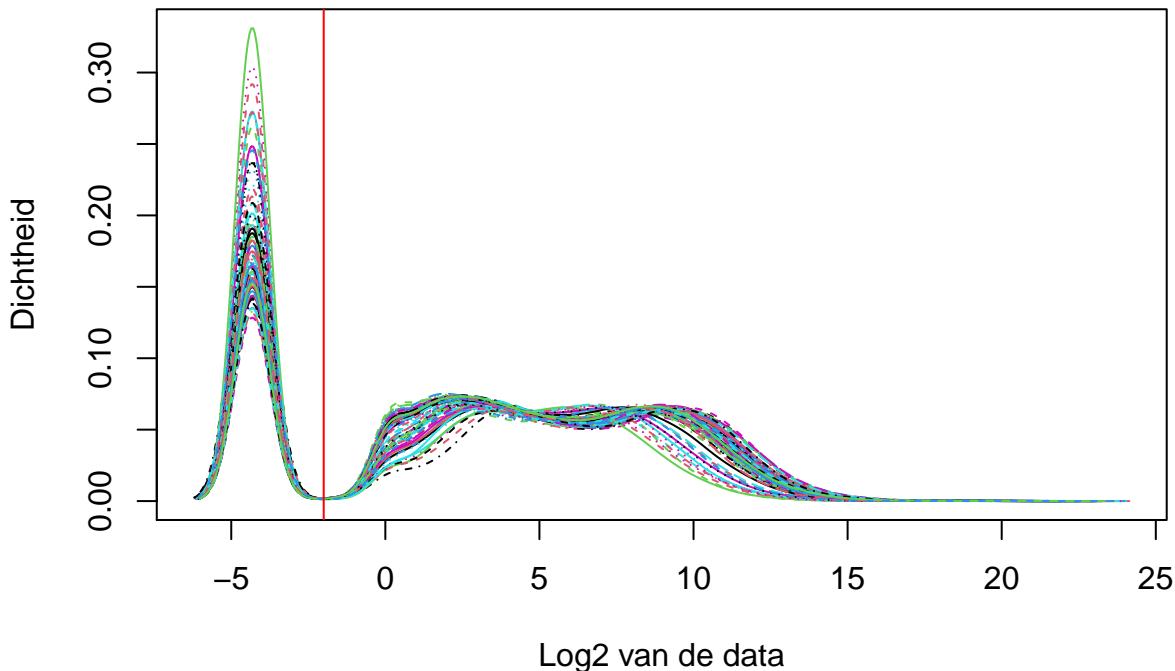


Zoals je kunt zien op het figuur liggen alle boxen behoorlijk dicht bij elkaar. Ook liggen de zwarte strepen behoorlijk op één lijn. Dit zijn de medianen. Toch zijn er wel een paar uitschieters. Deze zouden de data negatief kunnen beïnvloeden.

1.3.2 Dichtheidsplot Een andere manier om uitschieters in de data weer te geven is een dichtheids grafiek. Dit plot geeft ook de verdeling van de data weer. Als alle lijnen van de grafiek op elkaar liggen, is de data goed verdeeld. Eventuele uitschieters zijn goed zichtbaar. We willen dus in de grafiek zo min mogelijk lijnen zien die afwijken van de andere lijnen.

```
plotDensity(log2(data[9:ncol(data)] + 0.05),
main = "Dichtheidsplot van alle data samples",
ylab = "Dichtheid",
xlab = "Log2 van de data")
abline(v = -2, col = "red", lwd = 1, lty = 1)
```

Dichtheidsplot van alle data samples



Zoals je kunt zien zit er een hele groot piek links van de rode lijn. Zoals eerder opgemerkt bevat de data heel veel nullen. De data is getransformeerd. De \log_2 van 0.05 is ~ -4.3 . Deze piek zijn dus alle waarden met nul (daarom is er ook een rode lijn geplaatst). Deze waarden kunnen voor nu genegeerd worden.

In de figuur is op twee plekken een duidelijk ander verloop te zien. Aan het begin en aan het einde van de stijging is andere verloop tussen de samples. Hier is de data dus niet helemaal gelijk verdeeld. Na het normaliseren van de data is het mogelijk om eventuele afwijkende samples weg te gooien, om deze uitschieters eruit te filteren.

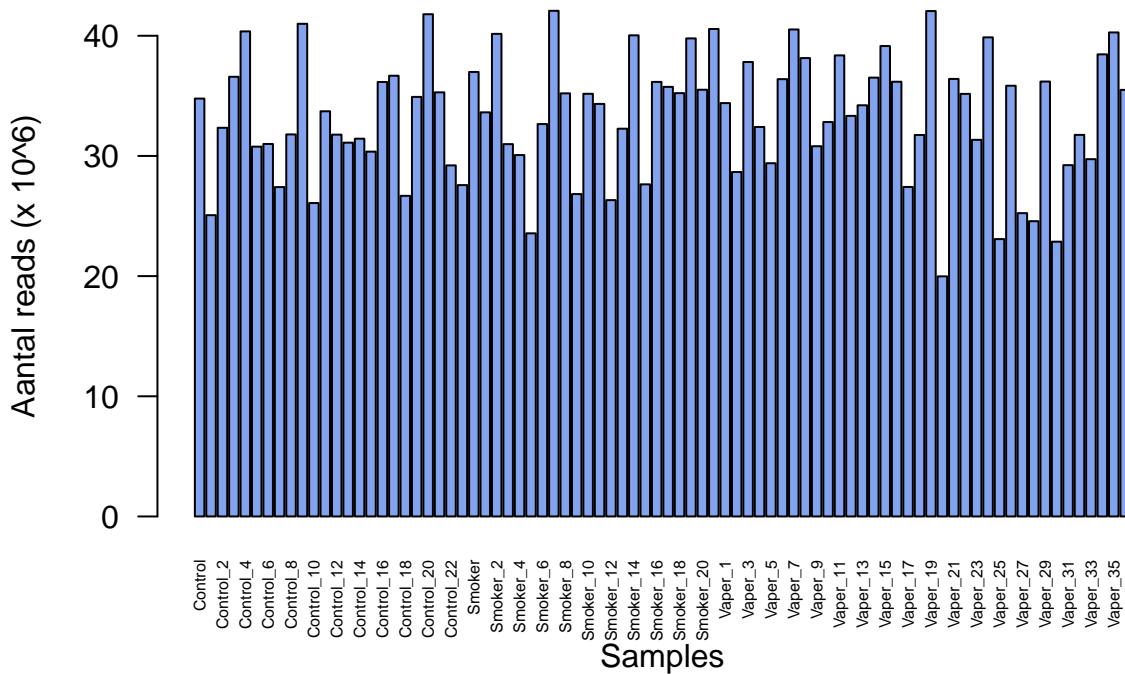
1.3.3 Staafdiagram Bij de laatste methode gaan we kijken naar de dekking van de data. Dit doen we met behulp van een staafdiagram van het aantal reads. We gebruiken de functie `cols` om alle waarden op te tellen. Het doel van dit figuur is om het aantal reads per sample weer te geven.

```

barplot(colSums(data[c(control, smoker, vaper)]) / 1e6,
        main = "Aantal reads per sample",
        xlab = "Samples",
        ylab = "Aantal reads (x 10^6)",
        las = 2,
        cex.names = .5,
        col = "#83A3EE")

```

Aantal reads per sample



Zoals zichtbaar is er voor elke sample een behoorlijke dekking. Er lijken dus geen samples aanwezig te zijn die te ‘weinig’ dekking hebben.

1.4 Normalisatie van de data en de afstand tussen de samples

Om te kijken of de data ver uit elkaar ligt, maken we twee figuren. We berekenen de afstanden tussen de datapunten en plotten dit.

Om zoveel mogelijk ruis uit de data te halen, wordt er genormaliseerd. Dit betekent dat de gegevens worden aangepast om er voor te zorgen dat de data meer gelijk is. Onderstaand zal er een simpele normalisatie worden uitgevoerd met het DESeq2 pakket. De meest basale vorm van normalisatie wordt toegepast. Er wordt onder andere genormaliseerd op basis van genlengte.

```

# Maken van het DESEQ object
ddsMat <- DESeqDataSetFromMatrix(countData = floor(data[c(control, smoker, vaper)]),
                                    colData = data.frame(samples = names(data[c(control, smoker, vaper)])),
                                    design = ~ 1)

## converting counts to integer mode

```

```

# Uitvoeren van de normalisatie
rld.dds <- vst(ddsMat)
# Waarden ophalen
rld <- assay(rld.dds)

```

1.4.1 Afstanden weergeven met heatmap Nu de data genormaliseerd is, kunnen de afstanden tussen de samples berekend worden om aan te kijken hoe dicht de data bij elkaar ligt. Een manier om de afstanden weer te geven is een heatmap. In een heatmap zijn de waarden die dicht bij elkaar liggen rood en de waarden die verder weg liggen blauw. De heatmap wordt gemaakt met de pheatmap package.

```

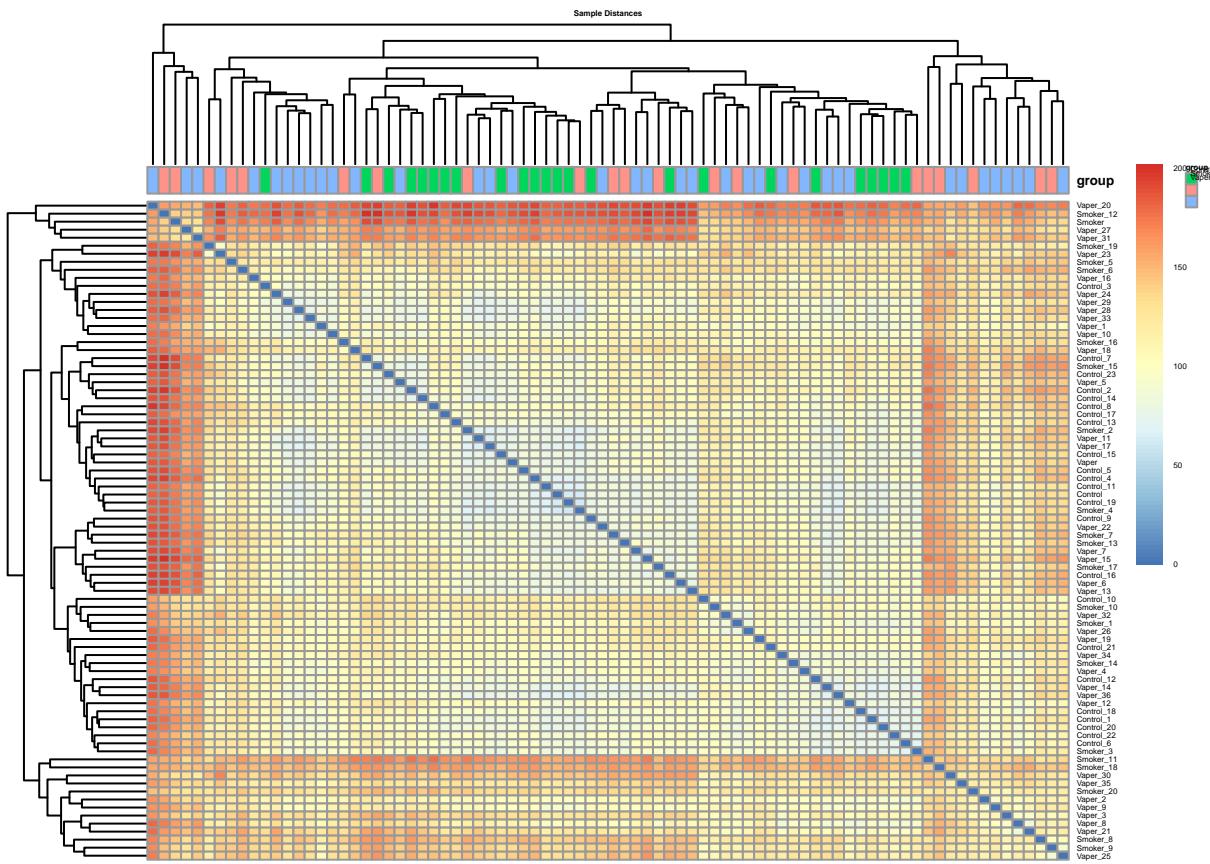
# Berekenen van de afstanden
sampledists <- dist(t(rld))
sampleDistMatrix <- as.matrix(sampledists)

# Het maken van een dataframe die gebruikt kan worden om te annoteren
group <- factor(c(rep(1, length(control)),
                     rep(2, length(smoker)),
                     rep(3, length(vaper))),
                  labels = c("Control", "Smoker", "Vaper"))

annotation <- data.frame(group = group)
rownames(annotation) <- names(data[,c(control, smoker, vaper)])

# Het maken van de heatmap
pheatmap(sampleDistMatrix, show_colnames = FALSE,
clustering_distance_rows = sampledists,
clustering_distance_cols = sampledists,
annotation_col = annotation,
cex = .5,
main = "Sample Distances")

```



Door de vele aanwezige samples is de heatmap zeer lastig af te lezen. Wel kun je zien dat de eerste vier samples verder weg liggen van de andere samples. Het is echter lastig om hier meer over te vertellen, omdat er zoveel samples aanwezig zijn.

1.4.2 Multi-dimensionale schaling (MDS) Op de data kunnen we ook multi-dimensionale schaling (MDS) toepassen. Dit is vergelijkbaar met de heatmap, maar hierbij worden de afstanden op een andere manier berekend. Bij MDS wordt Poisson afstandsberekening toegepast. Er wordt gekeken welke genen bij elkaar horen. De genen die bij elkaar horen, vormen een groepje in de grafiek.

Allereerst berekenen we de afstanden met de Poisson methode.

```
dds <- assay(ddsMat)
# Afstand berekenen
poisd <- PoissonDistance( t(dds), type = "deseq")

# De waarden ophalen uit de matrix
samplePoisDistMatrix <- as.matrix(poisd$dd)

# Het berekenen van de x- en y-coördinaten
mdsPoisData <- data.frame( cmdscale(samplePoisDistMatrix) )

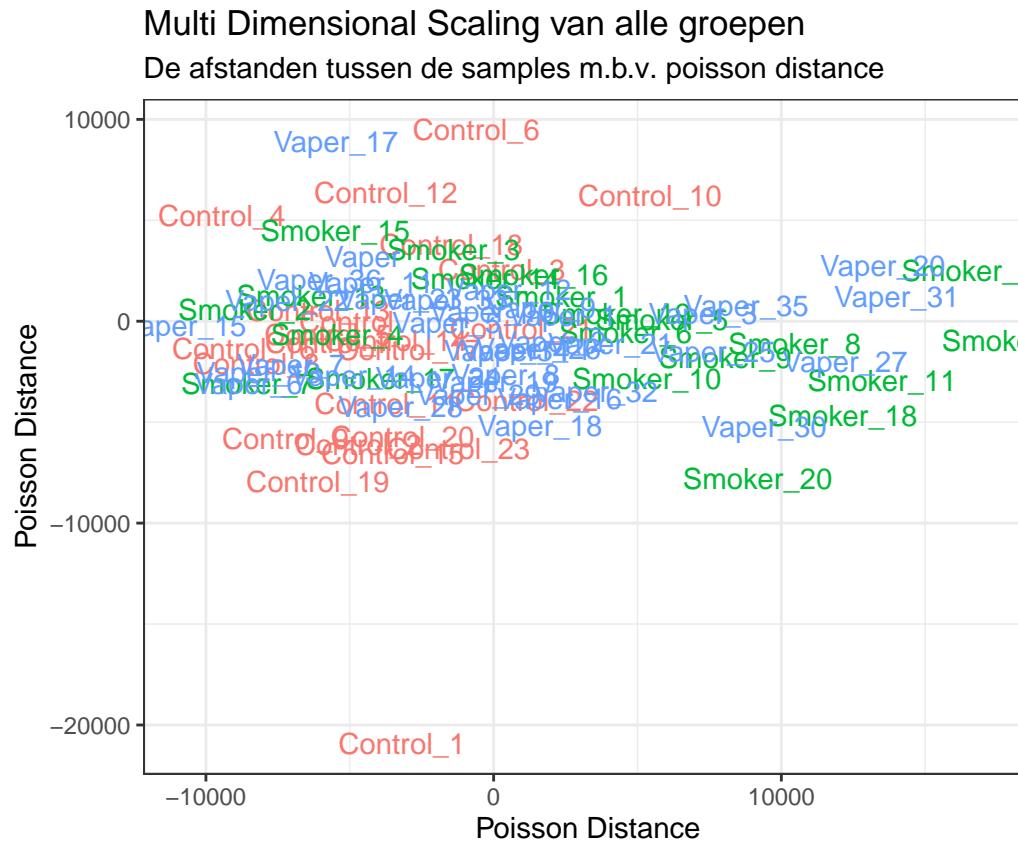
# Namen aanpassen voor het beter aflezen
names(mdsPoisData) <- c('x_coord', 'y_coord')
```

En natuurlijk het maken van de grafiek.

```

coldata <- names(data[c(control, smoker, vaper)])
ggplot(mdsPoisData, aes(x_coord, y_coord, color = group, label = coldata)) +
  geom_text(size = 4) +
  ggtitle('Multi Dimensional Scaling van alle groepen',
          subtitle = "De afstanden tussen de samples m.b.v. poisson distance") +
  labs(x = "Poisson Distance", y = "Poisson Distance") +
  theme_bw()

```



Zoals je kunt zien zijn er heel veel samples die verspreid zijn. Er zijn niet echt groepen. Toch kun je duidelijk zien dat de groepen vaper en control samen een groep vormen met blauw en groen. Er zijn van de smokers groep wel een aantal samples die ook in de groep vallen, maar dat zijn er niet veel. De smokers groep lijkt wat meer een afwijkend patroon aan te nemen richting rechts. Deze samples staan dus iets verder van de controle/vapers af.

2 Differentieel tot expressie gebrachte genen (DEG's)

In deze sectie gaan we kijken naar DEG's. Een DEG is een gen die met expressie afwijkt van normale genen. Een DEG is afhankelijk van de Log fold change waarde (LFC) en de p-waarde. Voordat we alle gegevens meteen in het pakket gooien, voeren we eerst 'handmatig' een aantal stappen uit.

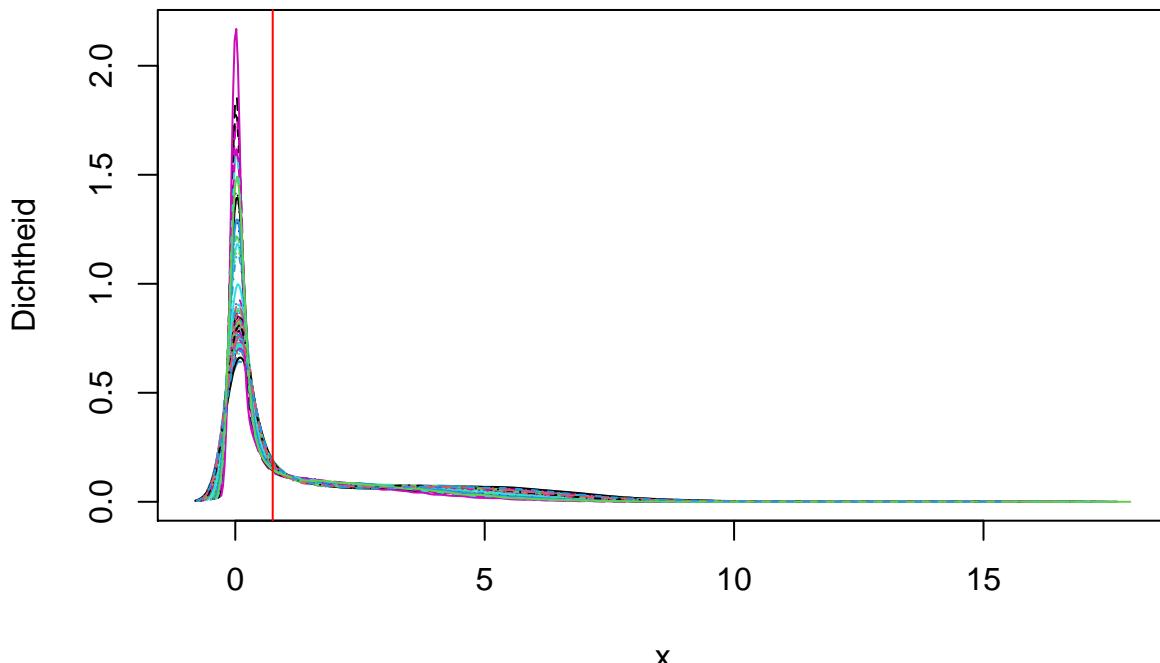
2.1 Voorbewerking van de data

Het uiteindelijke doel is om genen te vinden die afwijken in expressie van de 'controle' genen. Om dit aan te tonen, voeren we statistische testen uit. Echter zijn de testen heel erg gevoelig voor buitenliggende waarden. Om deze reden voeren we nog een normalisatie uit, maar dan op een andere manier.

2.1.1 Normalisatie op basis van reads We gaan de data normaliseren door middel van counts per miljoen. Het aantal counts wordt berekend per miljoen. Deze data transformeren we vervolgens met de log2. Eerder keken we nog naar de lengte van de genen, maar nu kijken we puur alleen naar het aantal counts.

```
counts.fpm <- log2( (data[c(control, smoker, vaper)] /  
                      (colSums(data[c(control, smoker, vaper)]) / 1e6)) + 1 )  
plotDensity(counts.fpm,  
            main = "Dichtheidsplot van alle data samples",  
            ylab = "Dichtheid")  
abline(v = .75, col = "red", lwd = 1, lty = 1)
```

Dichtheidsplot van alle data samples



In vergelijking met het vorige dichtheidsplot, lijkt deze dat nu beter verdeeld te zijn. De meeste lijnen liggen bijna allemaal op één lijn. De normalisatiemethode is echter wel een beetje 'kaal', omdat het zoals eerder vermeld alleen maar rekening houdt met het aantal reads.

2.1.2 Wegfilteren van de nullen in de dataset Zoals eerder vermeld zijn er heel veel nullen aanwezig in de dataset. Normaal gesproken gebruik je een pakket in R die deze waarden wegfilterd. Sommige paketten hebben echter deze mogelijkheid niet. Voor nu gaan we onze eigen manier bedenken/onderzoeken.

In het onderstaande blok wordt de data gefilterd en zullen de verschillen aangetoond worden.

```
new_data <- data[rowSums(data[9:ncol(data)] > 5) <= ncol(data[9:ncol(data)]) * 0.25,]
new_data_dims <- dim(new_data)

cat(sprintf("Data is gefilterd,
Oude dataset: %.f regels.
Nieuwe data: %.f regels.
Aantal regels weggefilterd: %.f",
dims[[1]], new_data_dims[[1]], dims[[1]] - new_data_dims[[1]]))

## Data is gefilterd,
## Oude dataset: 38921 regels.
## Nieuwe data: 14576 regels.
## Aantal regels weggefilterd: 24345
```

De genen moeten een expressiewaarde hoger dan 5 hebben. Dit moet gelden voor 75% van de samples.

Er zijn heel veel waarden onder de 5 aanwezig in de dataset. Daarom zijn er dus ook veel regels weggefilterd in de ‘nieuwe’ dataset.

2.2 Fold Change Value (FC)

De FC waarde is de log2 waarde van de experiment/controle ratio. Deze waarde geeft simpelweg aan of een bepaald gen een hogere of lagere expressie heeft. Om te ver komen dat de data erg wordt beïnvloed door ruis gebruik je de log2 waarde van de ratio. Dit is de Fold Change Value. Bij een FC-waarde van boven de 1, spreek je van een hogere expressie. Bij een FC-waarde tussen de 0 en 1, spreek je van een mindere expressie.

Hieronder wordt de Fold Change berekend van de genen in de dataset. Vervolgens worden deze waarden weergegeven in een barplot. Hiervoor wordt de FPM-data gebruikt (zie hierboven). Doordat de eerder genormaliseerde data een iets andere volgorde heeft, worden de kolommen voor de groepen opnieuw gedefineert.

```
# Het normaliseren van de gefilterde data
new_data.counts.fpm <- log2( (new_data[c(control, smoker, vaper)] /
                                colSums(new_data[c(control, smoker, vaper)])
                                / 1e6)) + 1 )

# Het berekenen van de ratios
vaper.means <- rowMeans(new_data.counts.fpm[grep("Vaper", names(new_data.counts.fpm))])
control.means <- rowMeans(new_data.counts.fpm[grep("Control", names(new_data.counts.fpm))])
FC <- vaper.means - control.means
```

Er wordt ook een normaalverdeling toegevoegd in het figuur. Het doel van deze barplot is om een normaalverdeling te krijgen. Je wil dat de LFC-waarde normaal verdeeld is, omdat de statistische testen zijn op aannames van de normaliteit van gegevens.

```
hist(FC,
     breaks = 40,
     main = "Histogram van de Fold Change Value (Vaper - Control)",
```

```

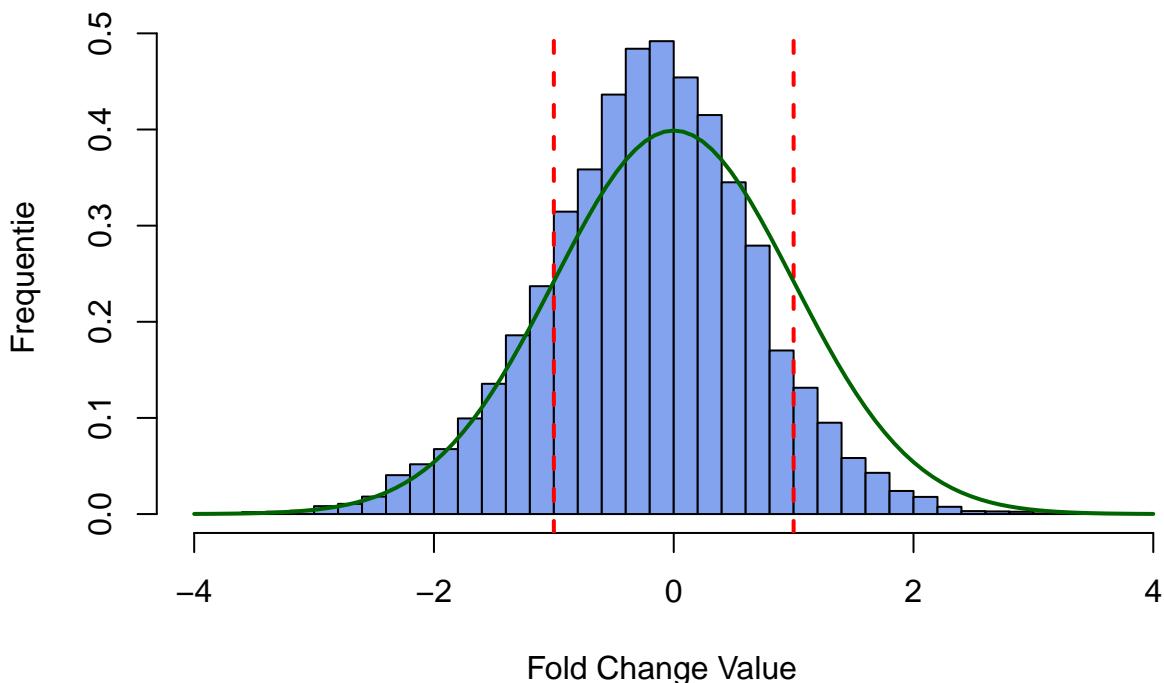
xlab = "Fold Change Value",
ylab = "Frequentie",
col = "#83A3EE",
freq = F)

# Lijn toevoegen bij -1 en 1
abline(v = -1, col = "red", lty = 2, lwd = 2)
abline(v = 1, col = "red", lty = 2, lwd = 2)

# Het plotten van de normaalverdeling.
# Allereerst datapunten maken tussen -4 en 4
x <- seq(-4, 4, by = 0.1)
curve(dnorm(x, 0, 1), from = -4, to = 4, col="darkgreen", add=T, lwd = 2)

```

Histogram van de Fold Change Value (Vaper – Control)



De groene lijn is de normaalverdeling. Zoals je kunt zien is de Fold Change redelijk normaal verdeeld op deze data.

2.3 DEG analyse met behulp van R-paketten

Nu er duidelijk is welke stappen we naar op zoek zijn, kunnen we beginnen met het zoeken naar de DEG's. We gebruiken hiervoor de DESEQ2 module. Deze module hebben we ook al eerder gebruikt voor de heatmap en mds-plot. Het annotation dataframe wat we voor de heatmap hebben gebruikt kunnen we nu weer gebruiken. Het is wel belangrijk dat de controle groep als eerste voorkomt in dit dataframe.

Om te beginnen maken we een DESeq object en voeren we hierop de testen uit. Omdat het experiment (maar) drie groepen bevat, voeren we ze allebei uit. Roker tegen controle en vaper tegen controle.

```

# Maken van het object
dds <- DESeqDataSetFromMatrix(countData = floor(data[c(control, smoker, vaper)]),
                                colData = annotation,
                                design = ~ group)

## converting counts to integer mode

dds <- DESeq(dds)

## estimating size factors

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing

## -- replacing outliers and refitting for 503 genes
## -- DESeq argument 'minReplicatesForReplace' = 7
## -- original counts are preserved in counts(dds)

## estimating dispersions

## fitting model and testing

# Deze functie geeft terug welke resultaten er mogelijk zijn met de dataset
resultsNames(dds)

```

Het object is aangemaakt en er is bekend welke testen er uitgevoerd kunnen worden. De twee resultaten kunnen dus gekoppeld worden aan objecten.

```

res.smoker.control <- results(dds, name = "group_Smoker_vs_Control")
res.vaper.control <- results(dds, name = "group_Vaper_vs_Control")

```

Voor het maken van de grafieken is het belangrijk om de ruis die komt van de LFC-waarden, wordt weggefilterd. Dit wordt gedaan met de lfcShrink functie.

```

res.smoker.control <- lfcShrink(dds, coef = "group_Smoker_vs_Control", type = "apeglm")

```

```

## using 'apeglm' for LFC shrinkage. If used in published research, please cite:
##      Zhu, A., Ibrahim, J.G., Love, M.I. (2018) Heavy-tailed prior distributions for
##      sequence count data: removing the noise and preserving large differences.
##      Bioinformatics. https://doi.org/10.1093/bioinformatics/bty895

```

```
res.vaper.control <- lfcShrink(dds, coef = "group_Vaper_vs_Control", type = "apeglm")
```

```
## using 'apeglm' for LFC shrinkage. If used in published research, please cite:  
##      Zhu, A., Ibrahim, J.G., Love, M.I. (2018) Heavy-tailed prior distributions for  
##      sequence count data: removing the noise and preserving large differences.  
##      Bioinformatics. https://doi.org/10.1093/bioinformatics/bty895
```

Met de summary functie wordt weergegeven hoeveel DEG's er zijn gevonden. Ook worden hier de percentages bijgegeven.

```
summary(res.smoker.control)
```

```
##  
## out of 38909 with nonzero total read count  
## adjusted p-value < 0.1  
## LFC > 0 (up)      : 4511, 12%  
## LFC < 0 (down)    : 1676, 4.3%  
## outliers [1]       : 0, 0%  
## low counts [2]     : 9064, 23%  
## (mean count < 1)  
## [1] see 'cooksCutoff' argument of ?results  
## [2] see 'independentFiltering' argument of ?results
```

```
summary(res.vaper.control)
```

```
##  
## out of 38909 with nonzero total read count  
## adjusted p-value < 0.1  
## LFC > 0 (up)      : 1604, 4.1%  
## LFC < 0 (down)    : 342, 0.88%  
## outliers [1]       : 0, 0%  
## low counts [2]     : 15099, 39%  
## (mean count < 4)  
## [1] see 'cooksCutoff' argument of ?results  
## [2] see 'independentFiltering' argument of ?results
```

Het is nu mogelijk om een MA plot te maken van de objecten. In een MA-plot wordt de LFC-waarde (M) afgezet tegen de gemiddelde genexpressie-waarde (A).

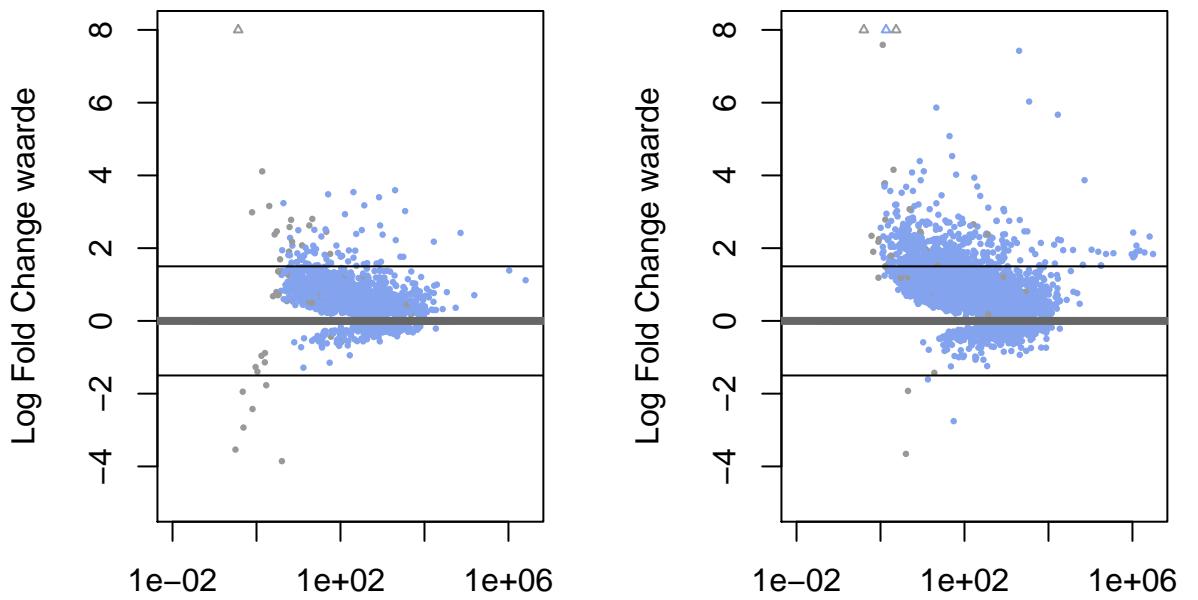
```
# Plots naast elkaar zetten met par()  
par(mfrow = c(1, 2))  
  
# Ma plots maken inclusief de LFC-lijnen bij -1.5 en 1.5  
plotMA(res.vaper.control,  
        xlab = "Gemiddelde van genormaliseerde tellingen",  
        ylab = "Log Fold Change waarde",  
        ylim = c(-5, 8),  
        colSig = "#83A3EE",  
        alpha = 0.1)  
abline(h = c(-1.5, 1.5))
```

```

plotMA(res.smoker.control,
       xlab = "Gemiddelde van genormaliseerde tellingen",
       ylab = "Log Fold Change waarde",
       ylim = c(-5, 8),
       colSig = "#83A3EE",
       alpha = 0.1)
abline(h = c(-1.5, 1.5))
mtext("DEG's van de groep vaper vs controle (links) en de groep smoker vs controle (rechts)",
      side = 3,
      line = -1,
      outer = T)

```

DEG's van de groep vaper vs controle (links) en de groep smoker vs controle (rechts)



Gemiddelde van genormaliseerde tellingen

Gemiddelde van genormaliseerde tellingen

Goed om op te merken is dat er in het originele onderzoek gebruik is gemaakt van een LFC-grens waarde van -1.5/1.5. De MA-plotten geven nu op basis van het eerder gemaakte Deseq-object, de DEG's weer. Voor deze gegevens kan geen grenswaarde worden opgegeven. Dit betekent bijvoorbeeld dat een gen met een LFC-waarde van 1.3 gekenmerkt zal worden als DEG door het Deseq-object. We zullen handmatig een lijn toevoegen bij -1.5 en 1.5 om de grafiek wat 'duidelijker' te maken. Alle blauwe stippen die onder/boven deze lijnen liggen kunnen worden beschouwd als DEG.

Zoals je kunt zien zijn er heel veel genen met een significantie die net onder de grens van 1.5 LFC liggen.

In de volgende sectie zullen we een volcano plot behandelen. In dit plot kan wel een grens opgeven voor de LFC. Op deze manier wordt er duidelijker welke genen nou echt DEG's zijn (volgens het originele onderzoek). Zonder deze grens zijn er nu 5000 DEG's bij de Rokers groep. Dit op een totaal aantal van ~ 38000 genen. Er worden standaard dus erg veel genen gekenmerkt als DEG en het is dus belangrijk om hier nog een extra 'grens' voor toe te voegen. Met deze grens kunnen de genen worden getoond die uitschieten.

3. Analyse en Visualisaties van de DEG's

Nu er een set met DEG's is, kunnen we meer figuren gaan maken. Ook kan er iets meer onderzoek worden gedaan naar deze DEG's

3.1 Volcano plot

Allereerst gaan we een volcano plot maken. Zoals hierboven beschreven kan je in een vulcano LFC-waarde weergeven. Deze waarde zal gebruikt worden als grenswaarde. DEG's die hieronder komen, zullen niet als DEG worden beschouwd. Verder is ook mogelijk om een grens waarde voor de p-waarde aan te geven. De p-waarde zegt iets over de significantie van een DEG. Het zegt dus niks over het gen, maar alleen hoe zeker het is dat het ontregeld is.

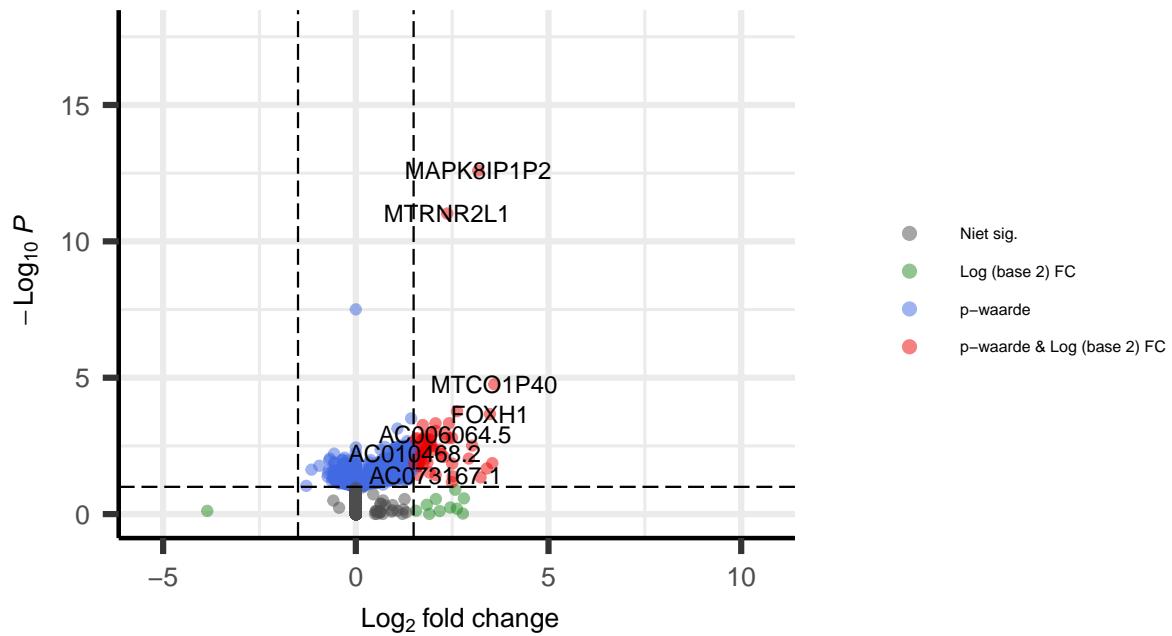
Het doel is om een vergelijkbaar aantal DEG's te krijgen die in het originele onderzoek ook aanwezig waren.

```
# Definieren van volcano plot functie
deseq.volcano <- function(res, datasetName) {
  return(EnhancedVolcano(res, x = 'log2FoldChange', y = 'padj',
    lab=rownames(res),
    title = paste(datasetName, ""),
    subtitle = bquote(italic('FDR <= 0.1 en absolute FC >= 1.5')),
    labSize = 3, pointSize = 1.5, axisLabSize=10, titleLabSize=12,
    subtitleLabSize=8, captionLabSize=10,
    legendLabels=c('Niet significant',
      'Log (base 2) FC',
      'p-waarde',
      'p-waarde & Log (base 2) FC'),
    legendPosition = 'right',
    legendLabSize = 6,
    legendIconSize = 2,
    pCutoff = 0.1, FCcutoff = 1.5))}

# Het maken van de Volcano grafieken
deseq.volcano(res = res.vaper.control,
  datasetName = "De genen van de groep vaper in vergelijking met controle")
```

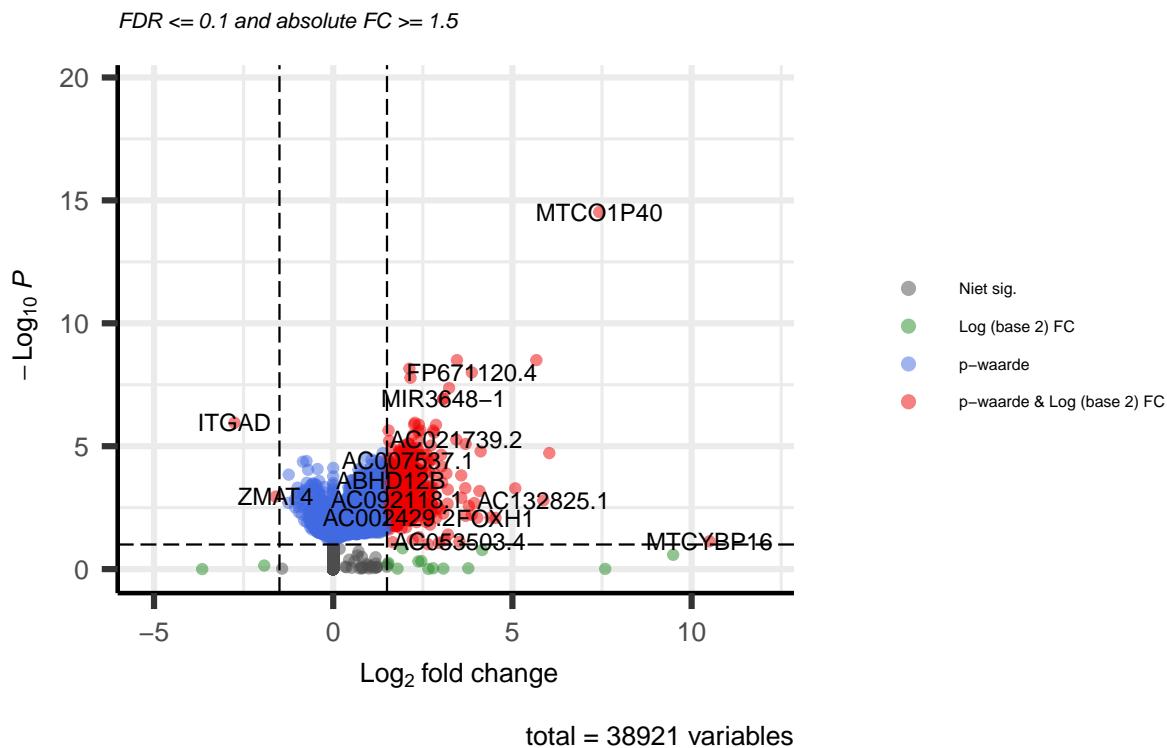
De genen van de groep vaper in vergelijking met controle

FDR <= 0.1 and absolute FC >= 1.5



```
deseq.volcano(res = res.smoker.control,  
                datasetName = "De genen van de groep smoker in vergelijking met controle")
```

De genen van de groep smoker in vergelijking met controle



Zoals beide volcano plots aantonen, zijn er veel genen die binnen de lijnen van de LFC-waardes vallen. Deze genen worden niet beschouwd als DEG. Zoals eerder beschreven werden deze genen wel zo gekenmerkt op basis van alleen de P-waarde. Deze valcano grafieken zijn daarom belangrijk om aan te tonen welke genen als DEG beschouwd kunnen worden.

Het valt verder op dat er veel DEG's aan de rechterkant van het figuur bevinden. Dit bevestigt de eerder behandelde MA-plot, waarin werd weergegeven dat de meeste DEG's een expressie waarde hebben die *hoger* ligt dan de controle groep.

3.2 Venn Diagram

In de volcano plot kan niet het absolute aantal DEG's worden afgelezen. Om deze reden maken we hieronder een venn diagram. Dit is een diagram waarin eenvoudig het aantal DEG's kan worden afgelezen. Op deze manier kunnen de verschillen tussen beide groepen ook duidelijker worden.

Verder is het in onderstaande diagram ook mogelijk om te zien hoeveel genen er overeenkomen. Dat wil zeggen, de hoeveelheid DEG's die gedeeld worden in beide groepen.

Allereerst worden onderstaand de juiste DEG's geselecteerd. Dit gebeurt op basis van p- en lfc-waarde.

```
# P-waarde (FDR) van 0.1
pval_threshold <- 0.1
# In het originele onderzoek is er geen log fold change gebruikt,
# maar een fold change. Dus om terug te gaan, log2(1.5)
lfc_threshold <- log2(1.5)

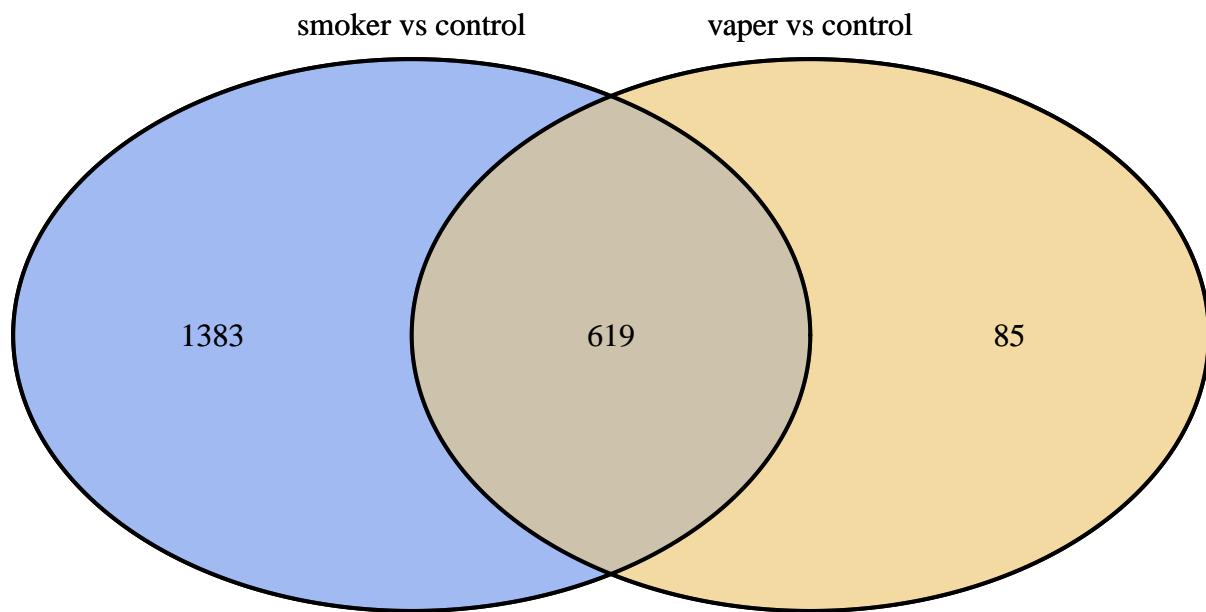
vaper.degs <- row.names(res.vaper.control[which(res.vaper.control$padj <=
pval_threshold & abs(res.vaper.control$log2FoldChange) > lfc_threshold)], ])
```

```
smoker.degs <- row.names(res.smoker.control)[which(res.smoker.control$padj <=
  pval_threshold & abs(res.smoker.control$log2FoldChange) > lfc_threshold)], ])
```

Met de juiste genen kan vervolgens een venn diagram gemaakt worden

```
venn.plot <- draw.pairwise.venn(length(vaper.degs),
  length(smoker.degs),
  length( intersect(vaper.degs, smoker.degs) ),
  category = c("vaper vs control", "smoker vs control"),
  scaled = F,
  fill = c("#eece83", "#83A3EE"),
  alpha = rep(0.5, 2),
  cat.pos = c(0, 0))

grid.draw(venn.plot)
```



De venn diagram toont aan dat 619 DEG's worden gedeeld in beide groepen. Voor de smokers groep is een aantal van 1383 DEG's uniek. Totaal bevat de groep dus 619 gedeelde + 1383 unieke = 2002 DEG's

3.3 Pad analyse

Nu er duidelijk is welke genen als DEG gekenmerkt worden, kunnen we verder kijken naar de biologische achtergrond. De DEG's worden gegroepeerd op basis van biologisch pad.

We voeren onze DEG's in in een online database genaamd DAVID. DAVID zal per gen kijken bij welk biologisch pad of functie het gen hoort. Er wordt per proces vervolgens ook een p-waarde gegeven.

Aan de hand van de p-waardes kan aangetoond worden of een proces of pad in je lichaam significant anders verloopt.

Eerst worden alle genen in het juist formaat weggeschreven naar een bestand. Vanuit dit bestand kunnen de genen in DAVID geplaatst worden.

```
vaper.degs.write <- sapply(vaper.degs, tools::file_path_sans_ext)
smoker.degs.write <- sapply(smoker.degs, tools::file_path_sans_ext)

write.table(vaper.degs.write, file = "vaper.degs.txt", row.names = F,
            quote = F, col.names = F)
write.table(smoker.degs.write, file = "smoker.degs.txt", row.names = F,
            quote = F, col.names = F)
```

DAVID gaat kijken of de opgegeven DEG's voorkomen in processen in het menselijk lichaam. Er wordt ook gegeven hoeveel genen hier betrokken bij zijn. Tot slot wordt er een p-waarde gegeven. Deze waarde geeft aan hoe significant het is, dat dit proces ontregeld is. Let op dat deze waarde niet aangeeft in welke richting dit proces is ontregeld.

De resultaten van DAVID zijn gedownload en worden onderstaand ingeladen.

```
DAVID_genes <- read.csv("data/chart_31BB3D290B901680533944143.txt",
                         sep = "\t", header = T)
```

We willen graag kijken welke paden in het lichaam ontregeld zijn. Hiervoor moeten we dus kijken naar de p-waardes. Onderstaand wordt de juiste informatie verkregen om een figuur te maken, waarin de p-waardes worden afgezet tegen de pathway titels.

Van de p-waarde wordt een -log10 genomen, omdat je anders de laagste waarde moet zoeken. Je zoekt nu naar de de staaf met de hoogste waarde.

```
DAVID.vaper.terms <- DAVID_genes$Term[DAVID_genes$Category %in% "KEGG_PATHWAY"]
DAVID.vaper.terms.split <- c()

for (term in DAVID.vaper.terms) {
  splitted <- strsplit(term, ":")[[1]][2]
  DAVID.vaper.terms.split <- c(DAVID.vaper.terms.split, splitted)
}

DAVID.p.values <- DAVID_genes$PValue[DAVID_genes$Category %in% "KEGG_PATHWAY"]
DAVID.df <- as.data.frame(DAVID.p.values, row.names = DAVID.vaper.terms.split)
```

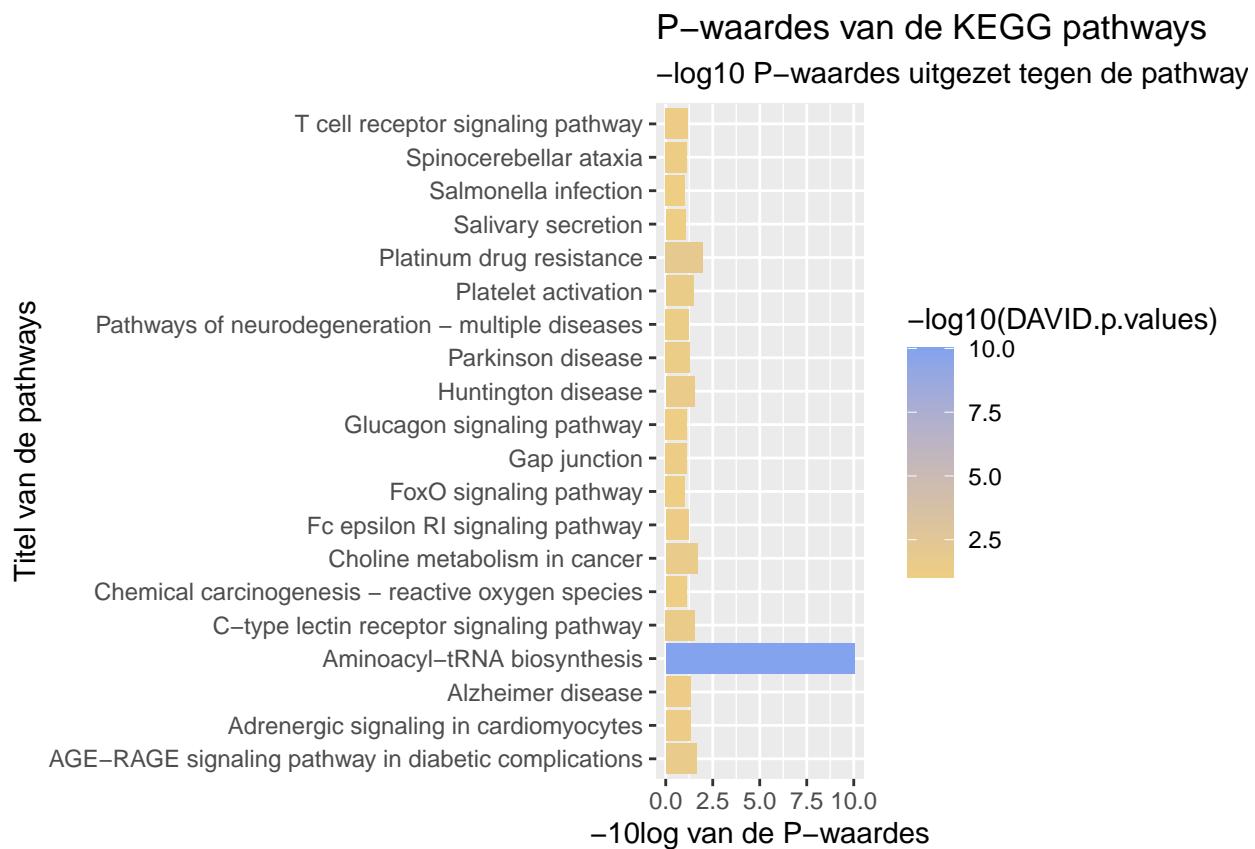
Vervolgens wordt een plot gemaakt van het dataframe.

```
ggplot(DAVID.df) + geom_bar(aes(x = rownames(DAVID.df),
                                 y = -log10(DAVID.p.values),
                                 fill = -log10(DAVID.p.values)), stat = 'identity') +
```

```

coord_flip() + scale_fill_gradient(low = "#eece83",
high = "#83A3EE") +
labs(title = "P-waardes van de KEGG pathways",
subtitle = "-log10 P-waardes uitgezet tegen de pathway titels") +
xlab("Titel van de pathways") + ylab("-10log van de P-waardes")

```



Zoals de barplot aantoon is er één proces die er echt uitschiet: Aminoacyl-tRNA biosynthesis. Dit proces gaat over het laden van de juiste aminozuren aan de juiste tRNA moleculen. Met zo'n grote $-\log_{10}$ p-waarde en dus kleine p-waarde, mag je zeggen dat dit proces significant ontregeld is. De p-waarde zegt niets over hoe het proces is ontregeld, alleen dat het zo is. We kunnen dus op deze manier niet meteen zeggen of dit proces een hogere regulatie vertoond of juist een lagere.

Je zou misschien denken dat het zeer waarschijnlijk is dat het een hogere regulatie heeft (omdat praktisch alle DEG's upregulated waren), maar het is ook mogelijk dat dit proces een inhiberend effect heeft. Dat betekent dat het juist een lagere werking heeft bij een hoge expressie.

Conclusie en discussie

Tijdens dit onderzoek zijn een aantal bevindingen gedaan. Allereerst is er geconcludeerd dat de dekking van de data goed genoeg was. Vervolgens is er gekeken naar de kwaliteit/verdeling. Hierover zal ik onderstaand nog iets verder op in gaan.

Er is uiteindelijk een mooie set met DEG's gevonden bij beide groepen. Hierom zijn er dus ook duidelijke visualisaties gemaakt. De volcano plot toonde aan dat de meeste van deze DEG's upregulated zijn. De Venn diagram toonde de absolute en overlappende DEG's tussen de beide groepen.

Het meest opvallende resultaat was toch wel de pad analyse. In deze Gene Enrichment Analyse werd aangetoond dat de synthese van tRNA significant is ontregeld in leukocyten bij vapende mensen. Deze resultaten waren *niet* terug te vinden bij de smokers groep. Zoals eerder vermeld is er al veel onderzoek gedaan naar de negatieve effecten van roken. Omdat er nog niet 'veel' onderzoek is gedaan naar velen, was het goed om dit uit te lichten.

In het refererende artikel naar dit onderzoek werd niet specifiek vermeld dat de synthese van tRNA ontregeld was. Wel vond ik teurg dat er een opvallend resultaat was in tRNA bij mitochondriaal DNA. Er werd verder niet veel verteld hoe of wat, maar alleen dat er procentueel gezien veel DEG's waren bij vapers.

Helaas was het niet meer mogelijk om verdere analyses uit te voeren naar de processen. Kort literair onderzoek toonde bijvoorbeeld aan dat er handvol onderzoeken zijn waar vergelijkbare resultaten werden gevonden. Bijvoorbeeld een onderzoek in het tijdschrift PLoS One in 2018. Dit onderzoek toonde aan dat de synthese van tRNA was verstoord in longcellen bij mensen die velen (zie referenties). Een vervolgstep voor dit onderzoek zou bijvoorbeeld kunnen zijn om aan te tonen welke genen precies ontregeld zijn. En ook vooral *hoe* deze ontregeld zijn.

Tot slot wil ik nog terug komen op de kwaliteit/verdeling controle. Door de vele samples benoemde ik eerder al dat het mogelijk was om data te verwijderen. Uiteindelijk is dit niet gebeurt bij het onderzoek. Het was misschien 'beter' geweest om nog meer te kijken naar samples. Er waren wel een paar uitschieters die wellicht verwijderd konden worden.

Referenties

Voor dit onderzoek zijn uiteraard een aantal belangrijke bronnen gebruikt. Hieronder worden de bronnen weergegeven.

1. DAVID Functional Annotation Bioinformatics Microarray Analysis. (z.d.). <https://david.ncifcrf.gov/>
2. I. Love, M., Anders, S., & Huber, W. (2023, 9 maart). Analyzing RNA-seq data with DESeq2. Geraadpleegd op 4 april 2023, van <http://bioconductor.org/packages-devel/bioc/vignettes/DESeq2/inst/doc/DESeq2.html>
3. Kempenaar, M. (2023, 14 februari). B Annotating an RNA-Seq Experiment | Analysis of Gene Expression. https://mkempenaar.github.io/gene_expression_analysis/a2-annotation.html#references
4. Lippi, G., Favaloro, E. J., Meschi, T., Mattiuzzi, C., Borghi, L., & Cervellin, G. (2013). E-Cigarettes and Cardiovascular Risk: Beyond Science and Mysticism. Seminars in Thrombosis and Hemostasis, 40(01), 060–065. <https://doi.org/10.1055/s-0033-1363468>
4. Tommasi, S., Pabustan, N., Liu, L., Chen, Y., Siegmund, K. D., & Besaratinia, A. (2021b). A novel role for vaping in mitochondrial gene dysregulation and inflammation fundamental to disease development. Scientific Reports, 11(1). <https://doi.org/10.1038/s41598-021-01965-1>