

Project_logboek_Genexpressie

Mark van de Streek

2023-03-24

Contents

Gen expressie Logboek	1
Het onderzoek	1
Verdeling van de groepen	2
Inladen van de Data	2
Eerste kijk op data	3
Classificatie van variabelen	3
Verdeling van de data	4
Normalisatie van de data en de afstand tussen de samples	5
Multi-dimensionele schaling (MDS)	7
Differentieel tot expressie gebrachte genen (DEG's)	8
Wegfilteren van de nullen in de dataset	9
Fold Change Value (FC)	10
DEG analyse met behulp van Bioconductor Packages	11

Gen expressie Logboek

In dit logboek wordt een onderzoek uitgelicht naar 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 er gekeken naar genen die aanwezig zijn in leukocyten (witte bloedcellen).

Het onderzoek

Er is gekeken naar de activiteit van genen van de volgende groepen mensen:

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

Om mensen te verkrijgen voor het onderzoek werd er om te beginnen een online vragenlijst gemaakt. Vervolgens werd er voor deze vragenlijst reclame gemaakt online, op bijvoorbeeld Reddit en Twitter. 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 voor het onderzoek werden gebruikt. Onderstaand de aantallen in een kleine 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, rassen en ook naar mensen die kunnen lezen en schrijven in het Engels. Het onderzoek vond plaats in Los Angeles.

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 varen, 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 geclassificeerd 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 konden nemen aan het onderzoek. Voor de traditionele rokers groep werd een periode van 1 jaar gebruikt voor deelname.

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.

Inladen van de Data

Onderstaand wordt de data van het bestand ingeladen. 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.

```
data <- read.table("data/GSE169757_Partek_Hum_BC_RNA_Jan2018_UnfilteredCounts_82.txt",
                    header = T, sep = "\t")

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/RtmpycqdCs/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 kolommen 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

```

Eerste kijk op data

Nu de data goed is ingeladen met alle juiste namen kunnen we kijken hoe het precies er 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)]))
```

Chromosome	Start	havana_gene	Control	Smoker	Control_1
4	67439992	—	2	2.602	5
6	5.3e+07	—	466	73.82	357.3
19	58345178	OTTHUMG00000183507.2	324.4	96.5	258.9
19	58347751	OTTHUMG00000183508.1	286.5	81.92	181.3

```

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 bijhorende 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.

Verdeling van de data

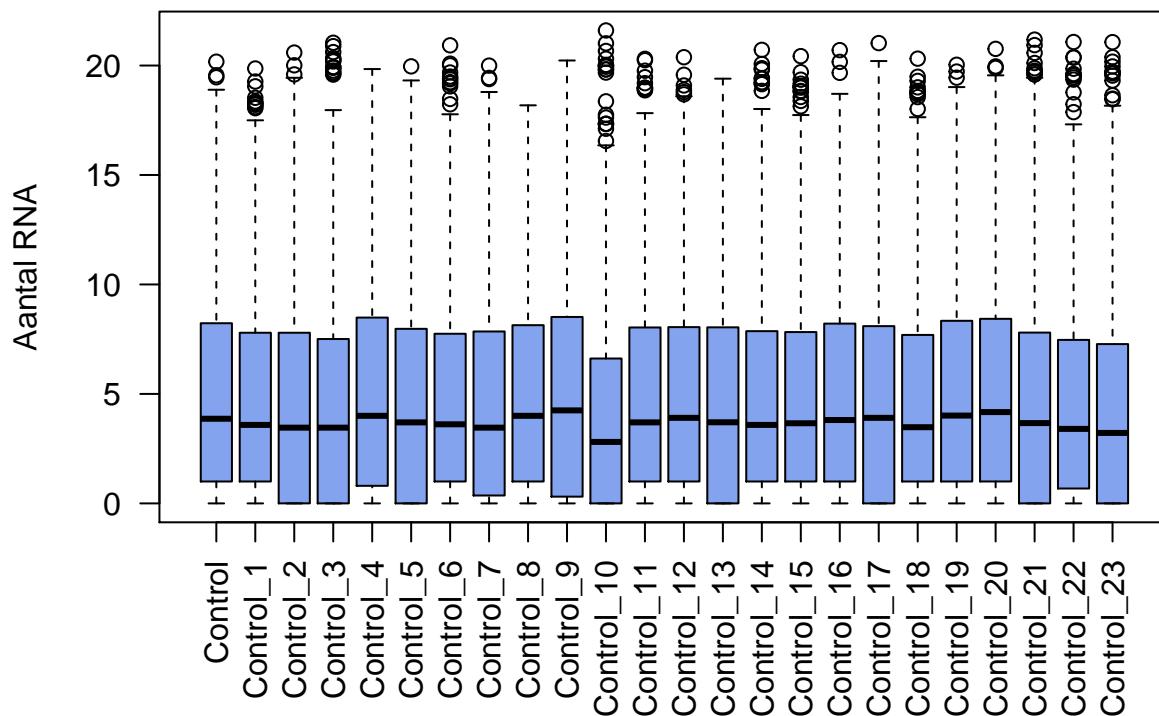
De groepen zijn nu geclasseerd en we kunnen dus kijken hoe de data is verdeeld. We kunnen het duidelijk weergeven op twee manieren. Allereerst een boxplot. Als de boxen van de figuren allemaal rond een lijn zitten, is de data goed dicht bij elkaar. Echter zal het met zo'n grote dataset nooit helemaal duidelijk verdeeld zijn. Er zijn bijvoorbeeld bij elke sample heel veel nullen aanwezig. Deze nullen 'trekken' de andere waren behoorlijk naar beneden. De boxplot heeft 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 controle groep, met getransformeerde data.

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

log2 transformatie van de verdeling van de control groep

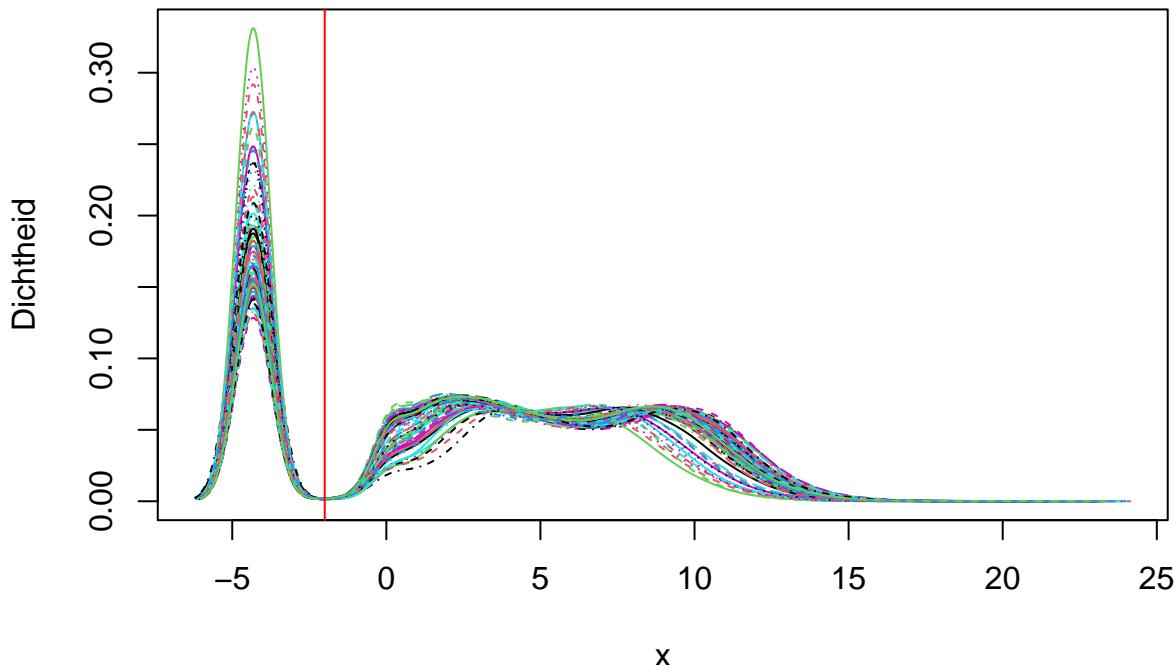


Zoals je kunt zien op het figuur liggen alle boxen behoorlijk dicht bij elkaar. Je zou dus zeggen dat deze data goed is verdeeld. Echter is dit nog maar een simpele log2 transformatie en kunnen we dus nog verder kijken naar normalisatie.

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.

```
plotDensity(log2(data[9:ncol(data)] + 0.05),
            main = "Dichtheidsplot van alle data samples",
            ylab = "Dichtheid")
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 log2 van 0.05 is ~ -4.3 . Deze piek zijn dus alle waarden met nul en kunnen genegeerd worden (daarom is er om die reden ook een rode lijn geplaatst).

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.

Normalisatie van de data en de afstand tussen de samples

Om zoveel mogelijk ruis uit de data te halen is het mogelijk om te normaliseren. Onderstaand zal er een simpele normalisatie worden uitgevoerd met het DESeq2 pakket. De meest basale vorm van normalisatie wordt toegepast.

```
(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
## class: DESeqDataSet
## dim: 38921 82
## metadata(1): version
## assays(1): counts
## rownames: NULL
## rowData names(0):
## colnames(82): Control Control_1 ... Vaper_35 Vaper_36
## colData names(1): samples
# uitvoeren van de normalisatie
rld.dds <- vst(ddsMat)
```

```
# waarden ophalen
rld <- assay(rld.dds)
```

Nu de data genormaliseerd is kunnen we de afstand berekenen tussen de samples om aan te tonen hoever de data bij elkaar is. Deze afstanden kunnen we weergeven in een heatmap.

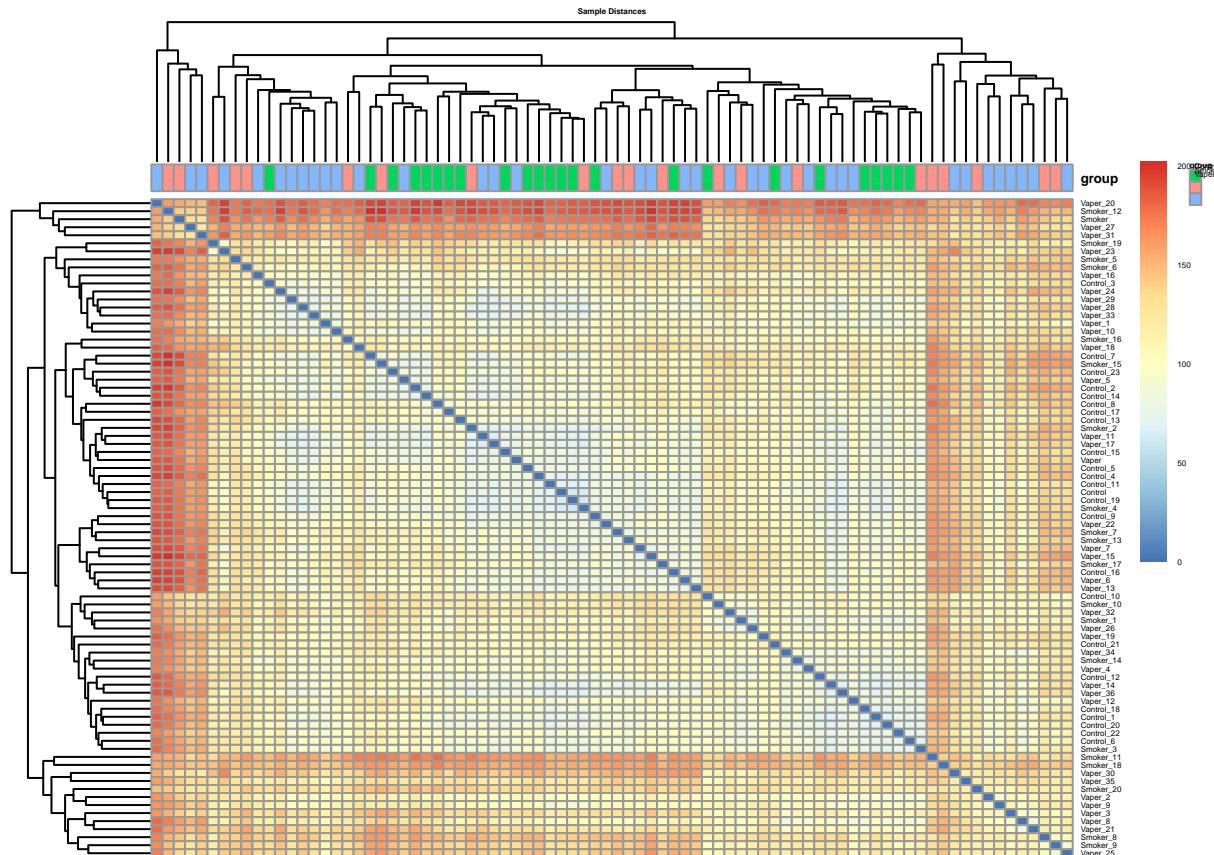
```
sampledists <- dist(t(rld))
# We use the 'pheatmap' library (install with install.packages('pheatmap'))
# Convert the 'dist' object into a matrix for creating a heatmap
sampleDistMatrix <- as.matrix(sampledists)

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)])

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 mooi zien dat de eerste vier samples verder weg liggen dan de andere samples. Het is echter lastig om hier meer over te vertellen, omdat er zoveel samples aanwezig zijn.

Multi-dimensionale schaling (MDS)

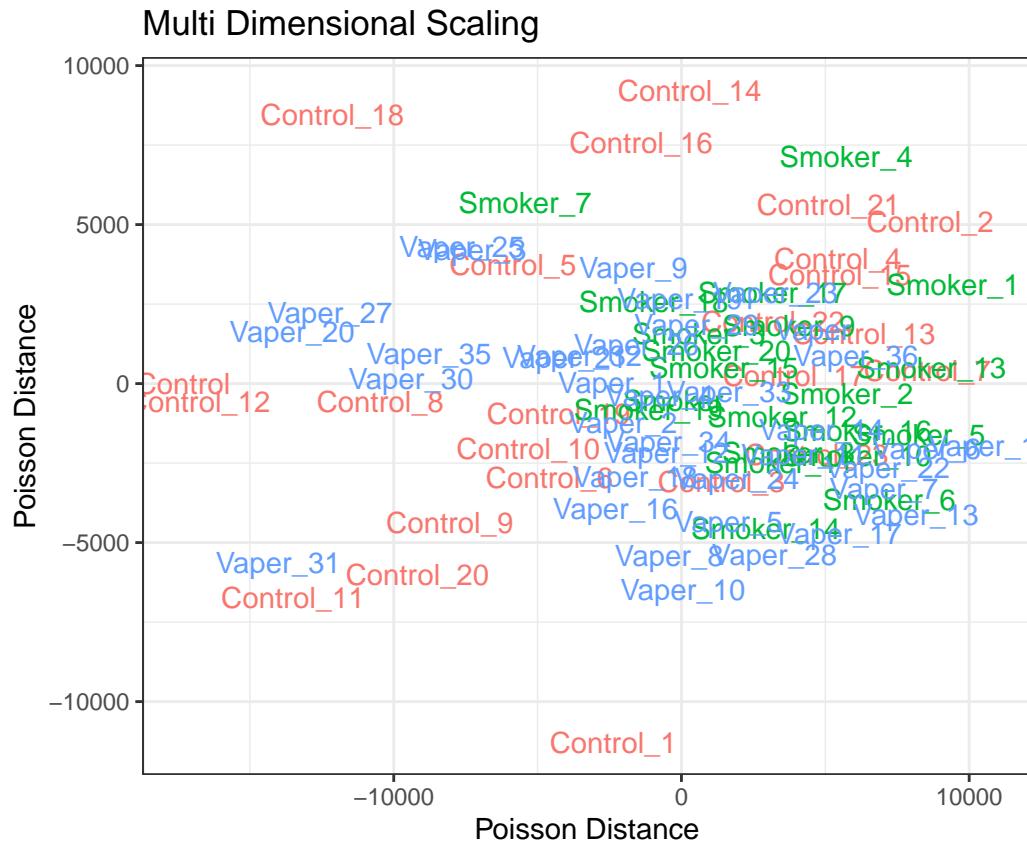
Op de data kunnen we ook multi-dimensionale schaling (MDS) toepassen. Dit is vergelijkbaar met de heatmap, maar hierbij worden de afstanden op een andere manier berekend. Ook worden ze 2d weergegeven. Bij MDS wordt Poisson afstandsberekening toegepast. Er wordt gekeken welke genen bij elkaar horen. De genen die bij elkaar horen, vormen een groepje in de grafiek.

```
# Note: uses the raw-count data, PoissonDistance performs normalization
# set by the 'type' parameter (uses DESeq)
dds <- assay(ddsMat)
poisd <- PoissonDistance( t(dds), type = "deseq")
# Extract the matrix with distances
samplePoisDistMatrix <- as.matrix(poisd$dd)
# Calculate the MDS and get the X- and Y-coordinates
mdsPoisData <- data.frame( cmdscale(samplePoisDistMatrix) )

# And set some better readable names for the columns
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') +
  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 rokers groep wel een aantal samples die ook in de groep vallen, maar dat zijn er niet veel. Ook vertoond de rokers groep behoorlijke individuele afwijkingen.

Differentieel tot expressie gebrachte genen (DEG's)

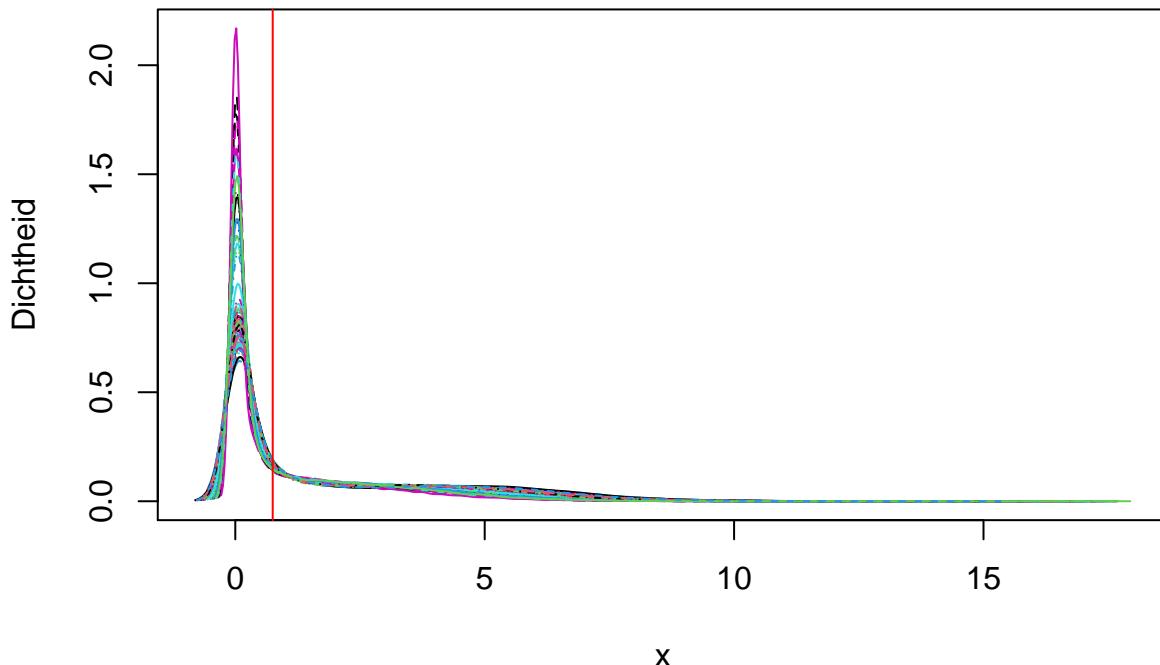
Onderstaand voeren we een aantal ‘handmatige’ stappen uit die een package simpel kan doen. Dit doen we om te kijken waar we nou precies naar op zoek zijn. Als laatste wordt wel een package gebruikt om de DEG’s op te sporen.

Het uiteindelijke doel is om genen te vinden die afwijken van de ‘normale’ genen. Het is ook belangrijk om genen te vinden die ook significant verschillen van de normaal. Hiervoor gaan we testen uitvoeren. 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. We kunnen altijd later nog makkelijk samples verwijderen en we kunnen daarom voor de zekerheid zoveel mogelijk visualisaties hebben om dit goed te onderbouwen.

We gaan de data normaliseren door middel van de 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)
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



de log₂ functie kan geen ‘overige’ waarden transformeren. De eerste negen kolommen worden dus niet meegegeven, alleen de ‘echte’ data.

Als we het vergelijken met het dichtheidsplot van de andere genormaliseerde data zou je zeggen dat dit velen malen beter is. De data lijkt nu ook velen malen beter verdeeld. Echter is dit maar één soort grafiek. We kunnen dus aan de hand van deze grafiek niet te veel conclusies trekken.

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. Voor nu gaan we onze eigen manier bedenken/onderzoeken. In het onderstaande blok wordt de data gefilterd en zullen de verschillen aangevoerd 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: 14896 regels.
##          Aantal regels weggefilterd: 24025
```

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

Fold Change Value (FC)

De FC waarde is de log2 waarde van de experiment/controle ratio. Deze waarde geeft aan simpelweg aan of een bepaald gen een hogere expressie heeft. Of een lagere expressie. Om te verkomen 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. Hiervoor wordt de FPM-data gebruikt (zie hierboven). Doordat de eerder genormaliseerde data een iets andere volgorde heeft, worden de kolommen voor de groepen

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

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))])

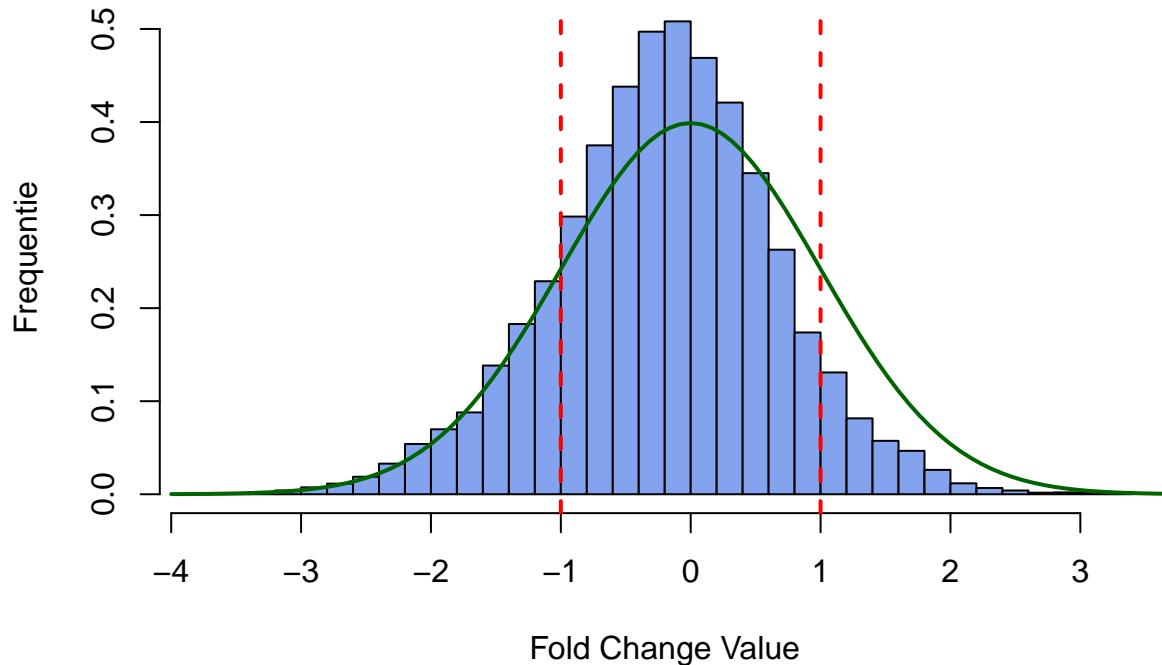
FC <- vaper.means - control.means

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 mooi normaal verdeeld. Dit betekend dat we dus zeker een aantal DEG's kunnen vinden.

DEG analyse met behulp van Bioconductor Packages

Nu er duidelijk is welke stappen we naar op zoek zijn, kunnen we beginnen met het zoeken naar de DEG's.

```
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
```

```

resultsNames(dds)

## [1] "Intercept"                  "group_Smoker_vs_Control"
## [3] "group_Vaper_vs_Control"

res_control_vs_smoker <- results(dds, name = "group_Smoker_vs_Control")
res_control_vs_vaper <- results(dds, name = "group_Vaper_vs_Control")

res_control_vs_vaper <- lfcShrink(dds, coef = "group_Vaper_vs_Control", type = "apeglm")

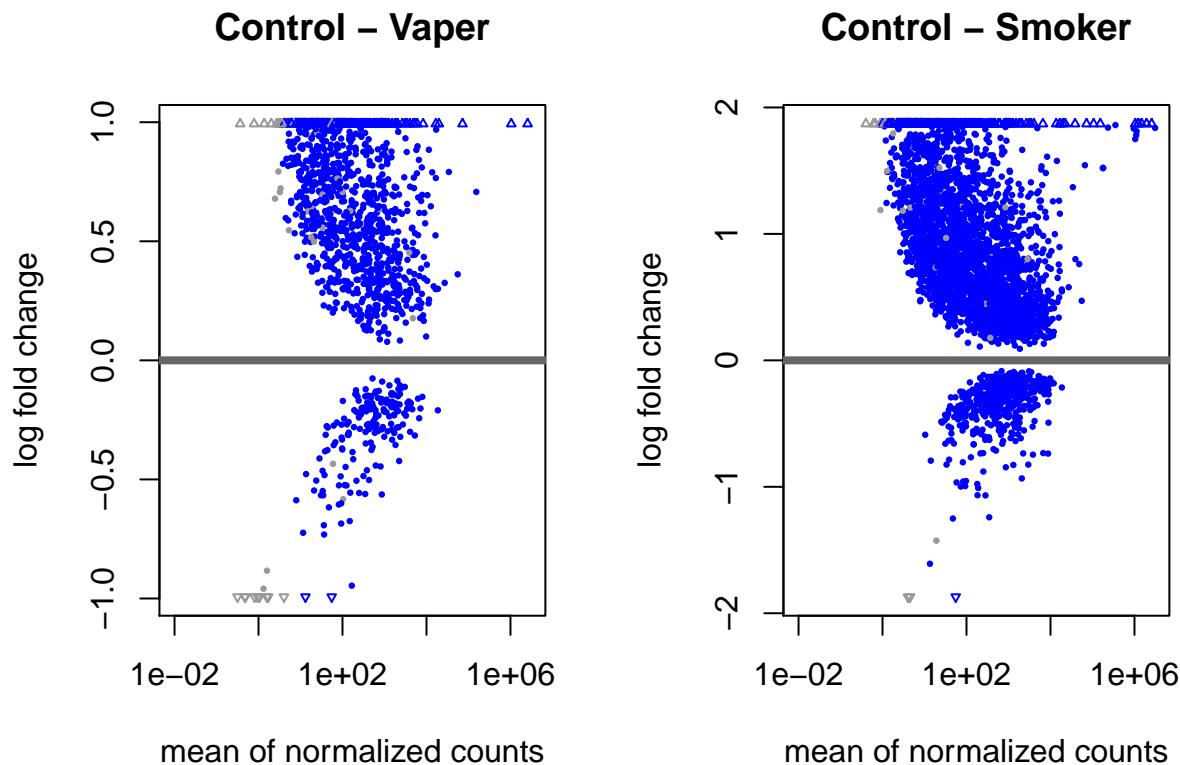
## using 'apeglm' for LFC shrinkage. If used in published research, please cite:
##      Zhu, A., Ibrahim, J.G., Love, M.I. (2018) Heavy-tailed prior distributions for
##      sequence count data: removing the noise and preserving large differences.
##      Bioinformatics. https://doi.org/10.1093/bioinformatics/bty895

res_control_vs_smoker <- 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

par(mfrow=c(1,2))
plotMA(res_control_vs_vaper, main = "Control - Vaper")
plotMA(res_control_vs_smoker, main = "Control - Smoker")

```



```

summary(res_control_vs_smoker)

##
## 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_control_vs_vaper)

##
## 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
```