Proyecto

Jocelyn Trujillo Gutierrez

2025-01-31

Librerias que se utilizan

```
library("recount3") # BiocManager::install("recount3", update = FALSE)
library("edgeR") # BiocManager::install("edgeR", update = FALSE)
library("limma") # BiocManager::install("limma", update = FALSE)
library("pheatmap") # BiocManager::install("pheatmap", update = FALSE)
library("RColorBrewer") # install.packages("RColorBrewer")
library("ggplot2") # install.packages("ggplot2")
```

Datos utilizados

Abstract de los datos obtenidos:

Se recolectaron 30 muestras de 15 sujetos en dos momentos temporales (a nivel del llano y a gran altitud). La significancia se determinó comparando los perfiles de expresión a gran altitud con los del llano.

Diseño general: 15 voluntarios viajaron desde un punto de partida a 1400 m hasta una zona a 5300 m en un período de 4 días.

```
## Todos los proyectos con datos de humano en recount3
human_projects <- available_projects()

## Obtencion de datos del proyecto SRP131480
rse_gene_SRP131480 <- create_rse(
    subset(
        human_projects,
        project == "SRP131480" & project_type == "data_sources"
    )
)

assay(rse_gene_SRP131480, "counts") <- compute_read_counts(rse_gene_SRP131480)
```

Una vez tenemos los datos, debemos explorarlos de manera general

```
# Explorar el objeto rse_SRP131480
rse_gene_SRP131480
```

```
## class: RangedSummarizedExperiment
## dim: 63856 30
```

```
## metadata(8): time_created recount3_version ... annotation recount3_url
## assays(2): raw_counts counts
## rownames(63856): ENSG00000278704.1 ENSG00000277400.1 ...
##   ENSG00000182484.15_PAR_Y ENSG00000227159.8_PAR_Y
## rowData names(10): source type ... havana_gene tag
## colnames(30): SRR6514110 SRR6514111 ... SRR6514138 SRR6514139
## colData names(175): rail_id external_id ...
##   recount_pred.curated.cell_line BigWigURL
```

Como podemos observar este objeto contiene 30 muestras y 63856 genes.

Con el proposito de facilitarnos la utilizacion de los datos, se modificara un poco el objeto rse_gene_SRP131480

```
rse_gene_SRP131480 <- expand_sra_attributes(rse_gene_SRP131480)

# Se verfica que todos los datos se vean bien
colData(rse_gene_SRP131480)[
    ,
    grepl("^sra_attribute", colnames(colData(rse_gene_SRP131480)))
]
```

```
## DataFrame with 30 rows and 4 columns
##            sra_attribute.altitude sra_attribute.lls_score
##                       <character>             <character>
## SRR6514110                  Plain                       0
## SRR6514111                  Plain                       0
## SRR6514112                  Plain                       0
## SRR6514113                  Plain                       0
## SRR6514114                  Plain                       0
## ...                           ...                     ...
## SRR6514135 High altitude exposure                       7
## SRR6514136 High altitude exposure                       6
## SRR6514137 High altitude exposure                       3
## SRR6514138 High altitude exposure                       1
## SRR6514139 High altitude exposure                       1
##            sra_attribute.source_name sra_attribute.tissue
##                          <character>           <character>
## SRR6514110                     blood                 blood
## SRR6514111                     blood                 blood
## SRR6514112                     blood                 blood
## SRR6514113                     blood                 blood
## SRR6514114                     blood                 blood
## ...                              ...                   ...
## SRR6514135                     blood                 blood
## SRR6514136                     blood                 blood
## SRR6514137                     blood                 blood
## SRR6514138                     blood                 blood
## SRR6514139                     blood                 blood
```

Nuestros datos no muestran ningun problema aparente por lo que procederemos a analizar un poco la diferencia entre nuestros datos de acuerdo a la fraccion del total de fragmentos asignados por 'featureCounts' que se unieron a un gen en especifico

```
rse_gene_SRP131480$assigned_gene_prop <-
    rse_gene_SRP131480$recount_qc.gene_fc_count_all.assigned /
    rse_gene_SRP131480$recount_qc.gene_fc_count_all.total
summary(rse_gene_SRP131480$assigned_gene_prop)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.3561  0.4015  0.4119  0.4195  0.4258  0.5469
```

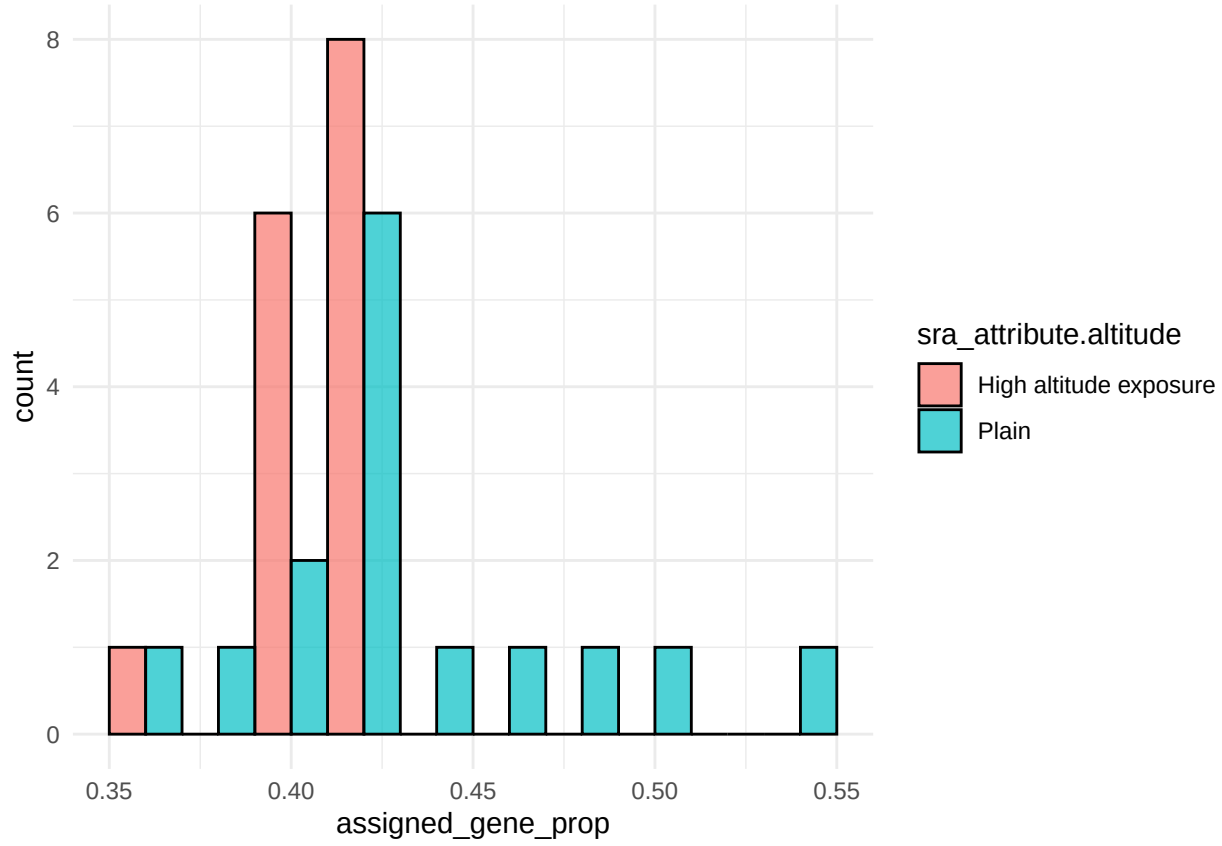
Ahora procederemos a visualizar estos mismos resultados pero separando las muestras de mayor altitud a las que se encuentran en plano.

```
with(colData(rse_gene_SRP131480), tapply(assigned_gene_prop, sra_attribute.altitude, summary))
```

```
## $`High altitude exposure`
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.3561  0.3993  0.4110  0.4068  0.4154  0.4286
##
## $Plain
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.3582  0.4101  0.4219  0.4322  0.4455  0.5469
```

Como nos muestran los resultados, podemos observar que hay una pequeña disminucion entre los valores de las muestras de mayor altitud en comparacion con las de plano. Para poder observar mejor este punto:

```
# Graficar
ggplot(as.data.frame(colData(rse_gene_SRP131480)), aes(x = assigned_gene_prop,
                                                       fill = sra_attribute.altitude)) +
  geom_histogram(binwidth = 0.02, color = "black", alpha = 0.7, position = "dodge") +
  theme_minimal()
```



Ahora procederemos a filtrar nuestros datos para poder realizar un analisis de expresion diferencial.

```
# Guardar los datos originales por si acaso
rse_unfiltered <- rse_gene_SRP131480

# Extraer la matriz de conteos
counts_matrix <- assays(rse_gene_SRP131480)$counts

# Extraer la información de los grupos
group <- as.factor(rse_gene_SRP131480$sra_attribute.altitude)

# Aplicar el filtro de expresión
keep_genes <- filterByExpr(counts_matrix, group=group)

# Filtrar con los genes seleccionados
rse_gene_SRP131480 <- rse_gene_SRP131480[keep_genes, ]
```

```
# Ver cuántos genes quedaron después del filtrado
dim(rse_gene_SRP131480)
```

```
## [1] 14752    30
```

```
## Porcentaje de genes que retuvimos
round(nrow(rse_gene_SRP131480) / nrow(rse_unfiltered) * 100, 2)
```

```
## [1] 23.1
```

Despues de aplicar el filtro de expresion, se redujo el numero de genes de 63856 a 14752, lo que es una disminucion de mas del 76%. El siguiente paso es aplicar la normalizacion de los datos.

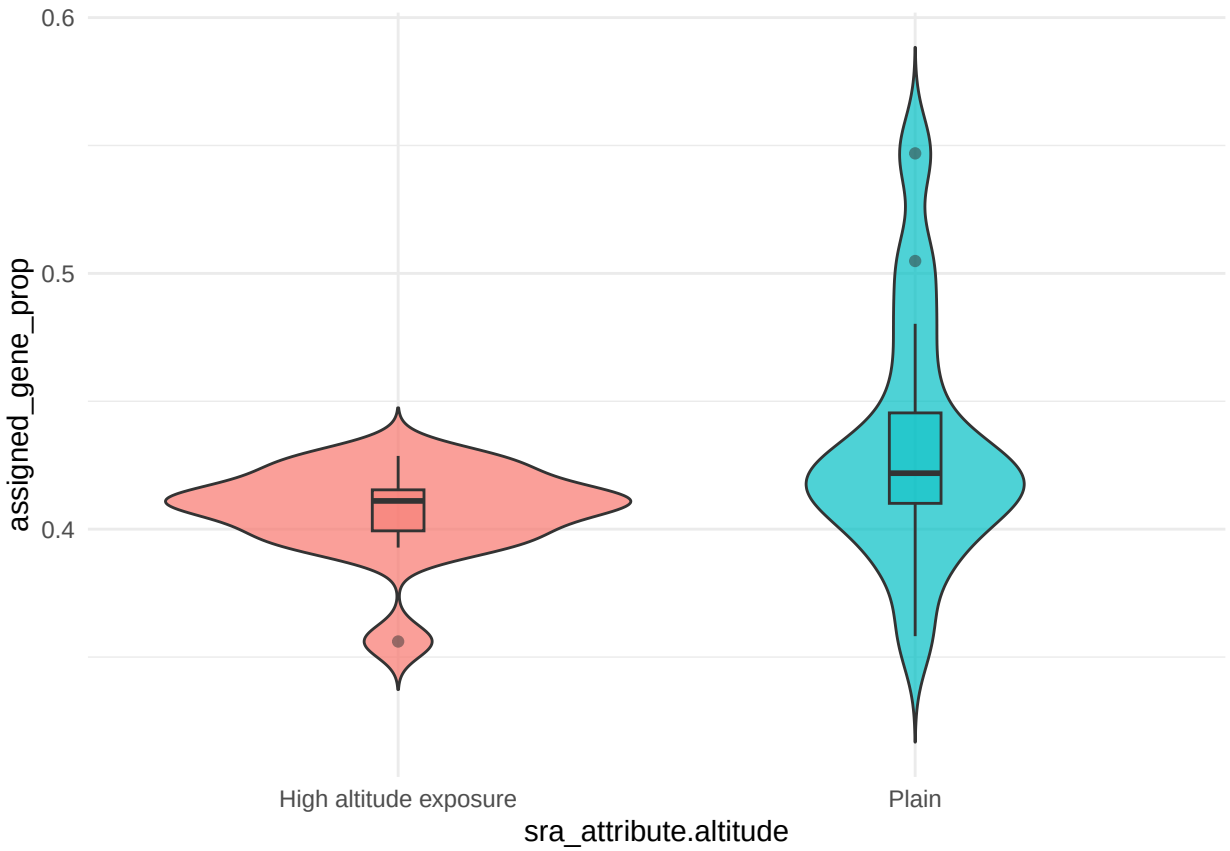
Normalizacion de los datos

```
dge <- DGEList(
    counts = assay(rse_gene_SRP131480, "counts"),
    genes = rowData(rse_gene_SRP131480)
)
dge <- calcNormFactors(dge)
```

Analisis de expresion diferencial

Primero exploramos nuestros datos para revisar que no haya otros problemas con las muestras y para explorar la relación entre nuestras variables.

```
ggplot(as.data.frame(colData(rse_gene_SRP131480)),
       aes(x = sra_attribute.altitude, y = assigned_gene_prop,
           fill = sra_attribute.altitude)) +
  geom_violin(trim = FALSE, alpha = 0.7) +
  geom_boxplot(width = 0.1, alpha = 0.5) +
  theme_bw(base_size = 20) +
  theme_minimal() +
  theme(legend.position = "none")
```



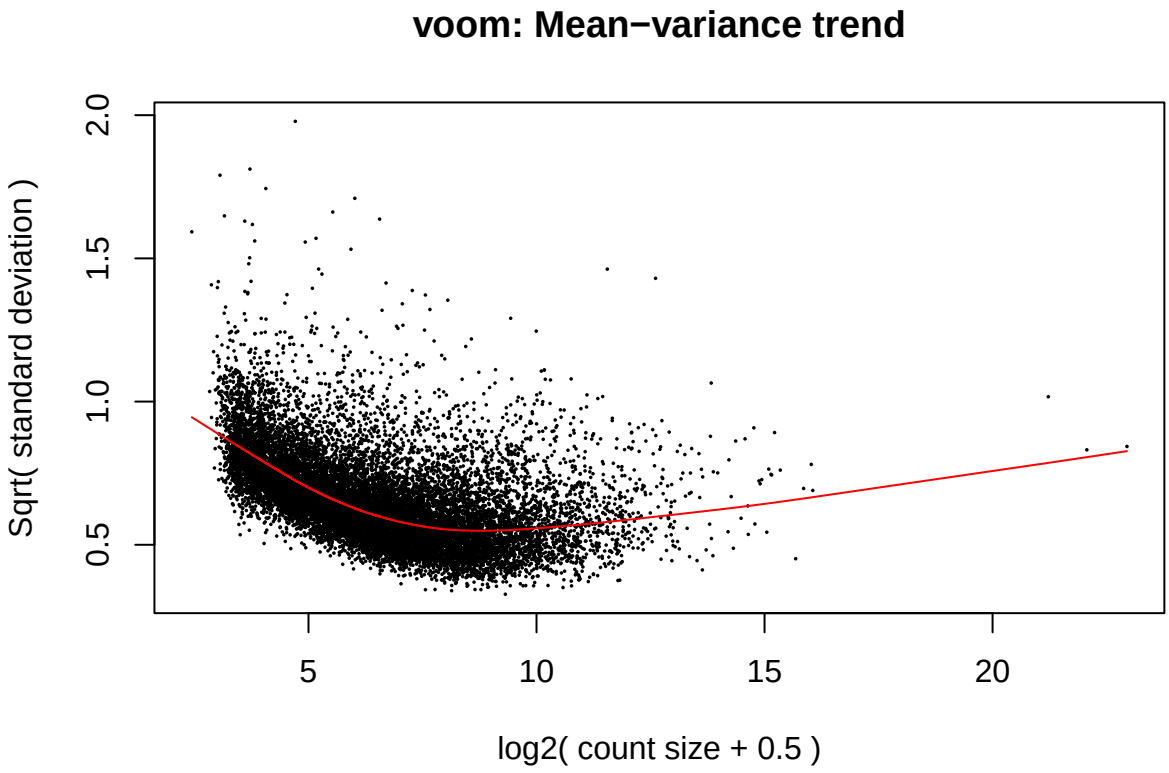
Ahora probaremos con un modelo estadistico

```
# Crear el modelo
mod <- model.matrix(~ sra_attribute.altitude + assigned_gene_prop,
                    data = colData(rse_gene_SRP131480))
# Visualizar sus columnas
colnames(mod)
```

```
## [1] "(Intercept)"                 "sra_attribute.altitudePlain"
## [3] "assigned_gene_prop"
```

Ya teniendo el modelo estadístico, usamos 'limma' para realizar el análisis de expresión diferencial

```
vGene <- voom(dge, mod, plot = TRUE)
```



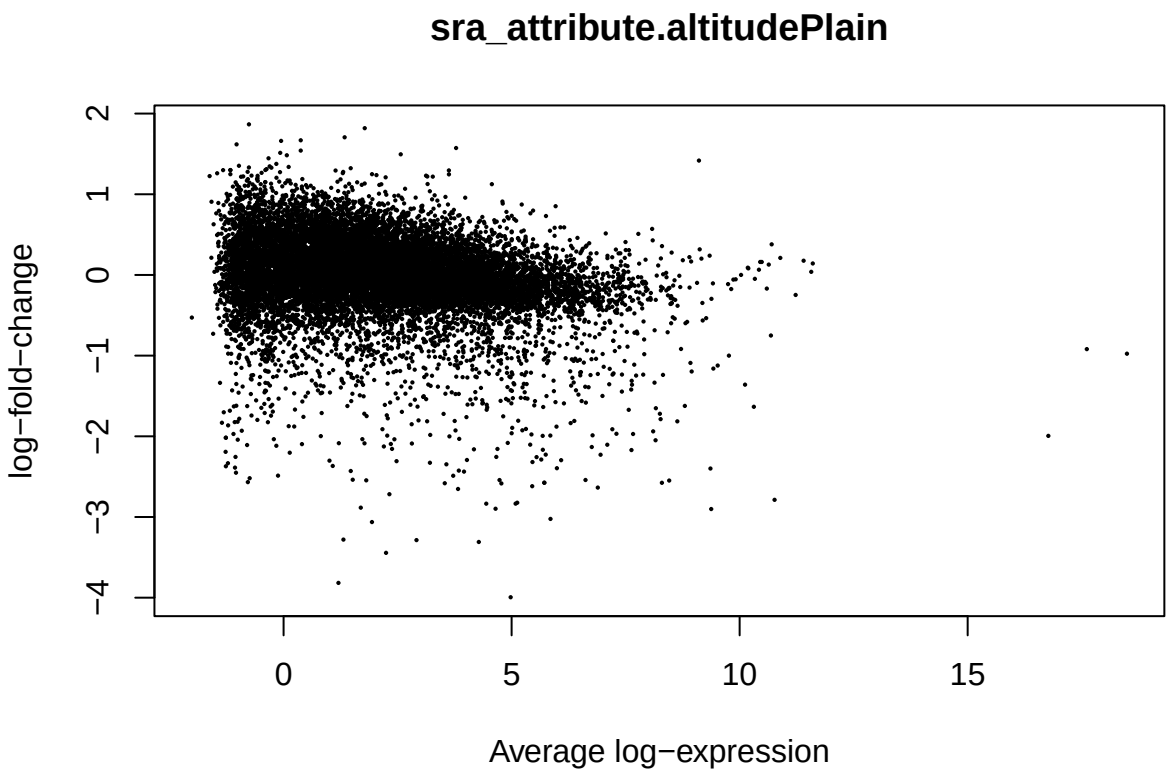
Ahora buscamos los genes diferencialmente expresados con un p_value < 0.05

```
eb_results <- eBayes(lmFit(vGene))

de_results <- topTable(
    eb_results,
    coef = 2,
    number = nrow(rse_gene_SRP131480),
    sort.by = "none"
)

## Genes diferencialmente expresados entre gran altitud y plano con FDR < 5%
table(de_results$adj.P.Val < 0.05)
```

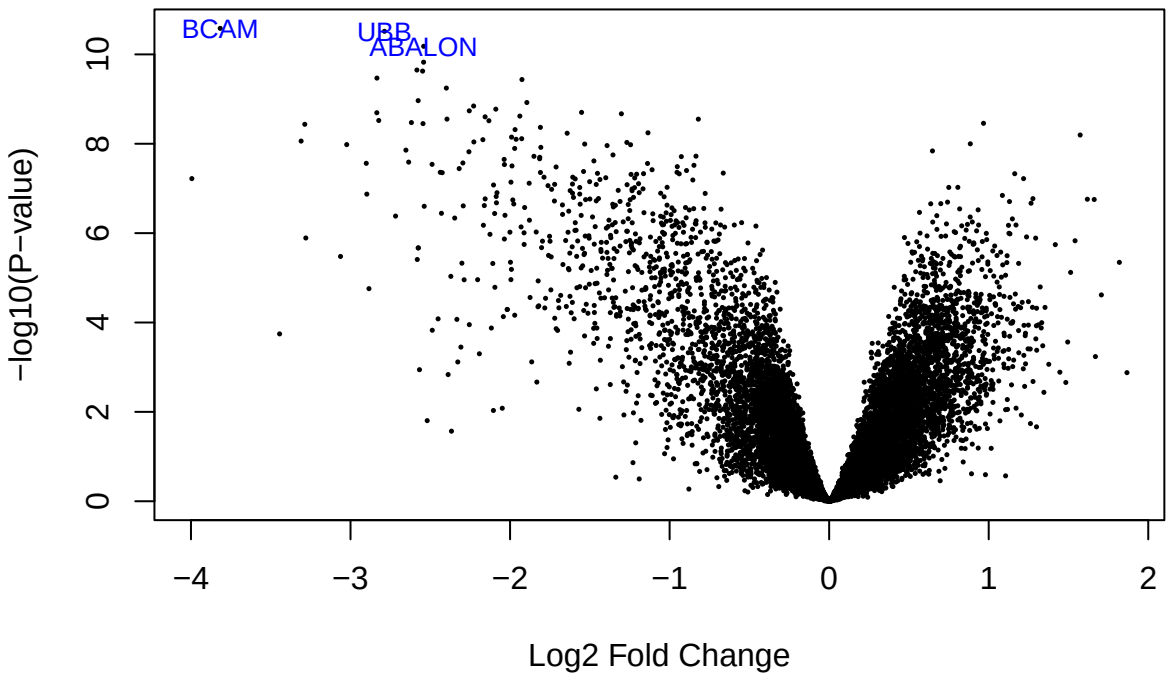
```
##
## FALSE  TRUE
## 10588  4164
```

De los 14752 genes que se analizaron, 4164 resultaron ser diferencialmente expresados con un p_value $<$ 0.05. Para visualizar estos resultados:

```
plotMA(eb_results, coef = 2)
```




```
volcanoplot(eb_results, coef = 2, highlight = 3, names = de_results$gene_name)
```



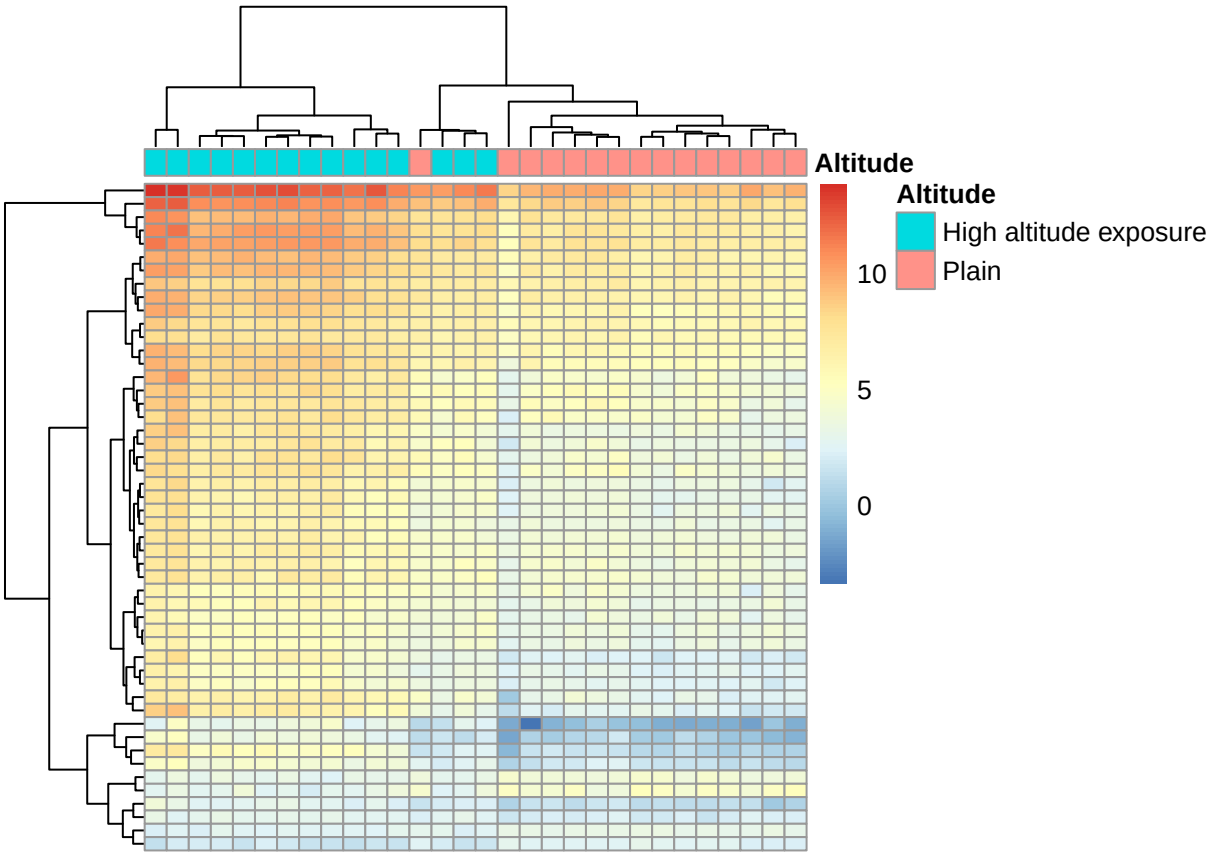

```
# Ver los genes que se encuentran en la parte superior derecha del gráfico
de_results[de_results$gene_name %in% c("BCAM", "UBB", "ABALON"), ]
```

```
##                     source type bp_length phase            gene_id
## ENSG00000170315.13 HAVANA gene      1621    NA ENSG00000170315.13
## ENSG00000187244.10 HAVANA gene      5024    NA ENSG00000187244.10
## ENSG00000281376.1  HAVANA gene      1903    NA  ENSG00000281376.1
##                          gene_type gene_name level          havana_gene  tag
## ENSG00000170315.13 protein_coding       UBB     1 OTTHUMG00000058987.6 <NA>
## ENSG00000187244.10 protein_coding      BCAM     1 OTTHUMG00000180838.3 <NA>
## ENSG00000281376.1       antisense    ABALON     2 OTTHUMG00000189574.1 <NA>
##                         logFC   AveExpr         t      P.Value    adj.P.Val
## ENSG00000170315.13 -2.787211 10.765904 -9.830086 3.042496e-11 2.244145e-07
## ENSG00000187244.10 -3.816741  1.202814 -9.891163 2.613806e-11 2.244145e-07
## ENSG00000281376.1  -2.541453  6.620134 -9.522095 6.592652e-11 3.241827e-07
##                           B
## ENSG00000170315.13 15.22456
## ENSG00000187244.10 15.02391
## ENSG00000281376.1  14.81650
```

Visualizar los genes diferencialmente expresados

De los datos normalizados por limma-voom, revisaremos aquellos top 50 genes diferencialmente expresados.

```
## Extraer valores de los genes de interés
exprs_heatmap <- vGene$E[rank(de_results$adj.P.Val) <= 50, ]

## Crear una tabla con información de las muestras
df <- as.data.frame(colData(rse_gene_SRP131480)[, "sra_attribute.altitude", drop = FALSE])
colnames(df) <- "Altitude"

## Crear el heatmap
pheatmap(
    exprs_heatmap,
    cluster_rows = TRUE,
    cluster_cols = TRUE,
    show_rownames = FALSE,
    show_colnames = FALSE,
    annotation_col = df
)
```



Como podemos observar se clusterizan casi perfectamente por altitud nuestras muestras, lo que nos indica que la altitud es un factor importante en la expresion de los genes. Otra forma para visualizarlo es:

```
# Convertir la columna Altitude a factor
df$Altitude <- factor(df$Altitude)

# Asignar colores basados en los niveles de Altitude
col.group <- df$Altitude
levels(col.group) <- brewer.pal(nlevels(col.group), "Set1")
```

```r
col.group <- as.character(col.group)

# Realizar el MDS y graficar
plotMDS(vGene$E, labels = df$Altitude, col = col.group, pch = 19, cex = 0.7)
```