Análisis e interpretación de datos multi-ómicos

Ejercicio final de transcriptómica (curso 2024-2025)

Alumno: David Valcárcel Linares

Master Bioinfo ISCIII

Apartado 1

El dataset original (GSE37211) consta de 27 muestras paired-end depositadas en SRA. Con el fin de poder abordar las cuestiones planteadas a en el primer apartado sólo es necesario descargar 2 muestras (SRR479052 y SRR479054), es decir cuatro ficheros fastq. Además, en este repositorio se proporciona:

- Un fichero fasta con la secuencia de la referencia genómica, en este caso correspondiente al cromosoma 21 humano (ensamblaje GRCh38).
- Un fichero GTF con la anotación génica para los genes del cromosoma 21 (GRCh38.ensembl.109).

Se pide realizar un análisis de dichas muestras similar al realizado en clase, considerando los siguientes puntos:

Pregunta 1 (1.5 puntos)

Realizar control de calidad de dichas muestras con el programa FastQC, incluir plots más reseñables y comentar cada uno de los apartados. De manera complementaria se podrá realizar un análisis de contaminación con el programa FastQScreen.

En primer lugar, necesitamos instalar el entorno con los paquetes necesarios para llevar a cabo la práctica. El entorno utilizado se puede encontrar en la carpeta envs del siguiente enlace: https://github.com/dvalcarcel/transcriptomic-final-exercise/tree/main/envs

Los archivos input y ouput de las preguntas del apartado 1, se pueden encontrar en el siguiente enlace: https://github.com/dvalcarcel/transcriptomic-final-exercise/tree/main/Apartado1/input

Para instalar el entorno se tiene que ejecutar el siguiente código:

```
(base) vant@MOOVE15:~/Escritorio/transcriptomic-final-exercise$ conda env create -f
envs/transcriptomic.yaml
```

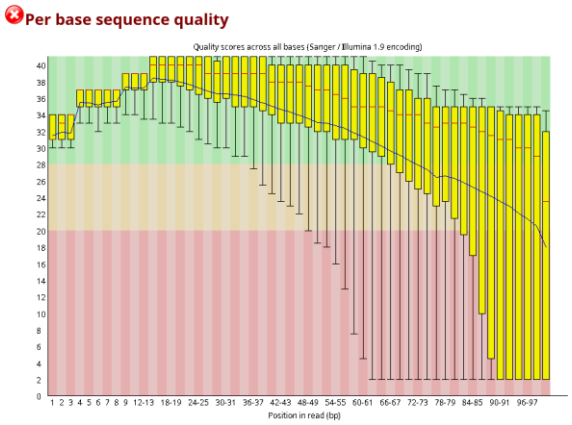
Una vez activado el entorno, para lanzar el fastqc ejecutaremos el siguiente código:

```
(transcriptomics-final-exercise) vant@MOOVE15:~/Escritorio/transcriptomic-final-exercise$
mkdir Apartado1/input/fastqc
fastqc -o Apartado1/input/fastqc/ Apartado1/input/*.fastq
```

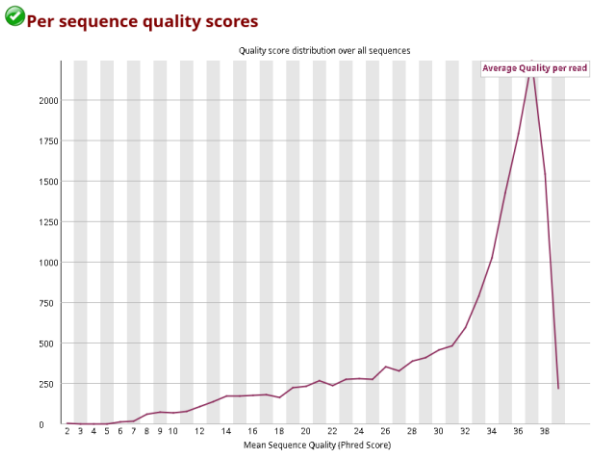
He creado un directorio de salida para los archivos generados por fastqc y he ejecutado el fastqc en todos los archivos .fastq de la carpeta input.

Fastqc SRR479052.chr21_1.fastq

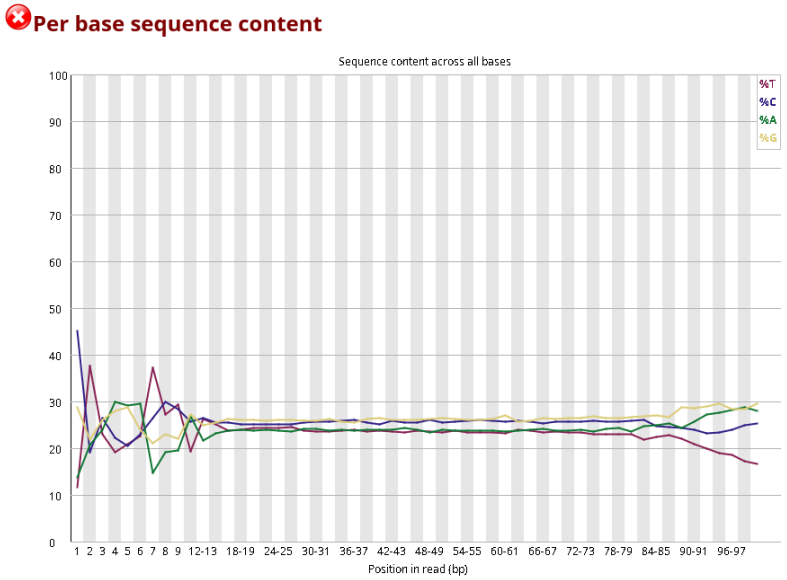
Hay un total de 15.340 de lecturas, un total de 1.5 Mbp secuenciadas, ninguna de calidad pobre, con una longitud de 101 pares de bases por lectura y un %GC de 52.



El gráfico muestra la calidad de las bases a lo largo de la secuencia. Hasta la secuencia 50, la calidad de las secuencias es alta, pero a partir de ese punto la calidad baja considerablemente. Las secuencias de illumina suelen caer al final, pero este descenso hay que tenerlo en cuenta en los siguientes pasos.

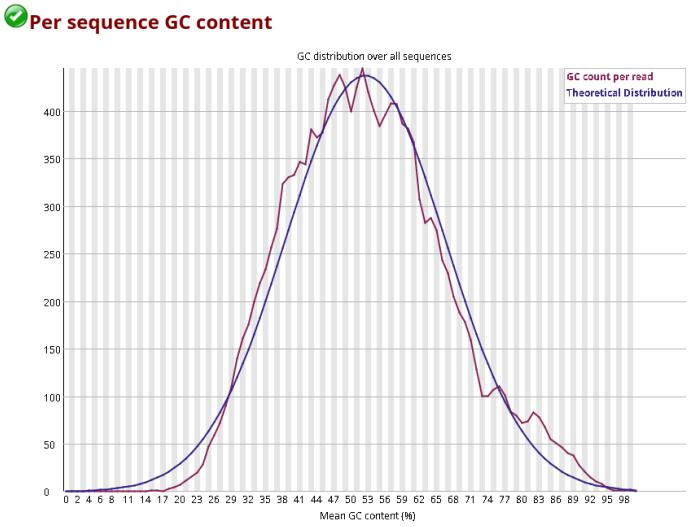


La mayoría de las lecturas presentan unos scores de calidad adecuados.

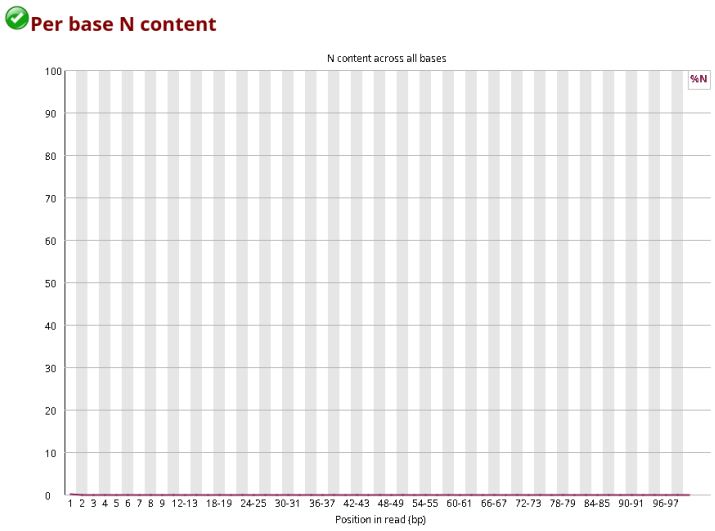


A pesar de que parece que el contenido en GC no es aceptable, se trata de una imagen típica de RNAseq y se considera que es aceptable.

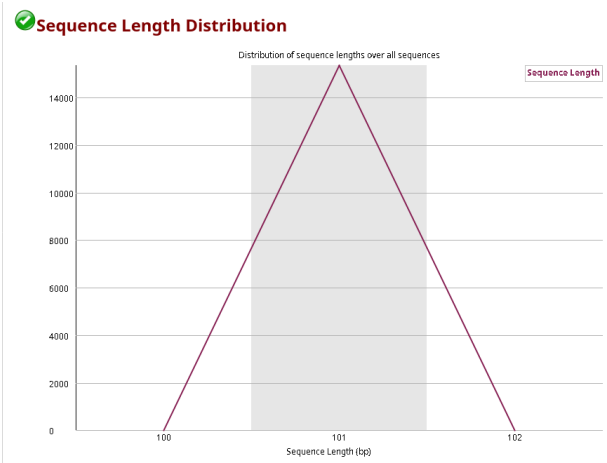
Las regiones GC son las más problemáticas porque son las más difíciles de amplificar por PCR porque son tres puentes de hidrógeno en lugar de dos.



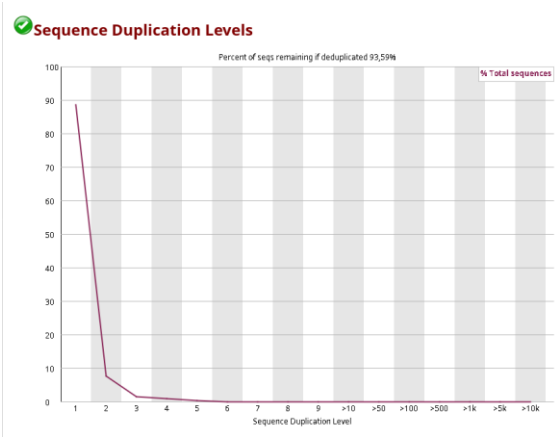
El contenido en GC por secuencia sigue una distribución normal.



No hay acumulación de bases N, por lo que no hubo problemas en la detección de nucleótidos.



Todas las lecturas tienen una longitud media de 100 a 102 pares de bases.

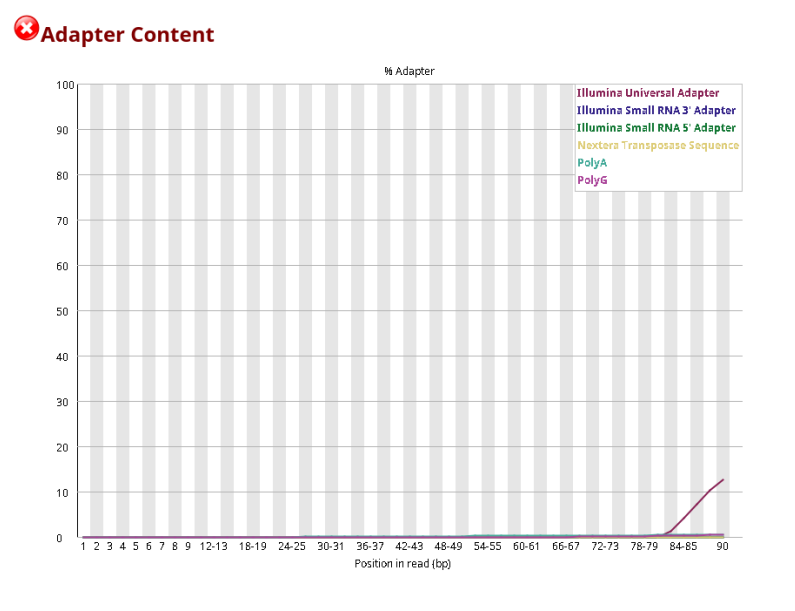


Los niveles de duplicación se encuentran en unos niveles aceptables.

Overrepresented sequences

Sequence	Count	Percentage	Possible Source
CTTTTACTTCCTCTAGATAGTCAAGTTCGACCGTCTTCTCAGCGCTCCGC	21	0.13689700130378096	No Hit

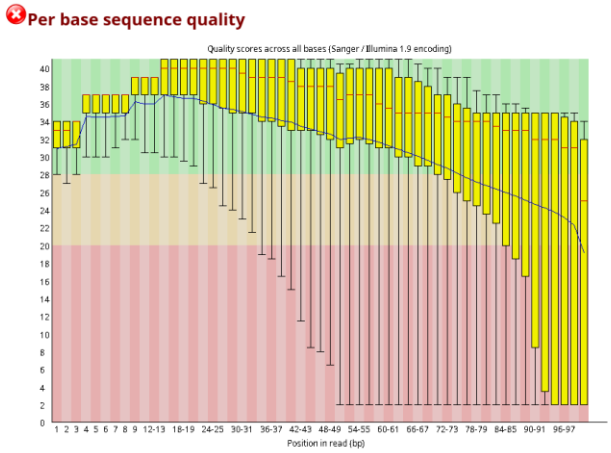
Hay una secuencia sobrerrepresentada que habrá que considerar si es un adaptador o contaminación.



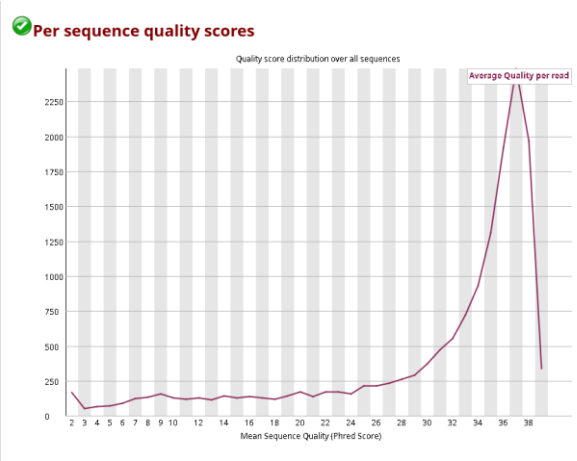
Por último, esta gráfica muestra la presencia de un adaptador universal de illumina.

Fastqc SRR479052.chr21_2.fastq

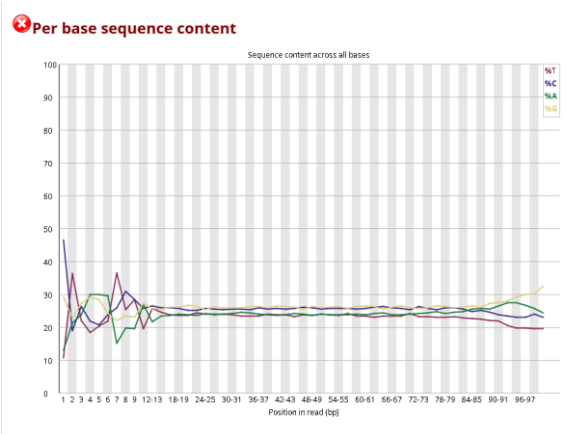
Al igual que en la primera secuencia de la muestra, hay un total de 15.340 de lecturas, un total de 1.5 Mbp secuenciadas, ninguna de calidad pobre, con una longitud de 101 pares de bases por lectura y un %GC de 52.



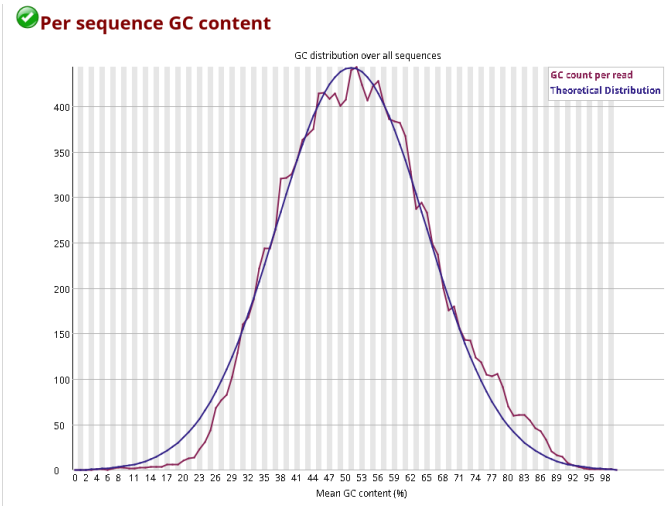
El gráfico de la calidad de las bases muestra algo muy similar a la primera secuencia. En general la calidad es alta, pero desciende mucho al final. Algo que puede ser normal en illumina, pero la caída es demasiado considerable.



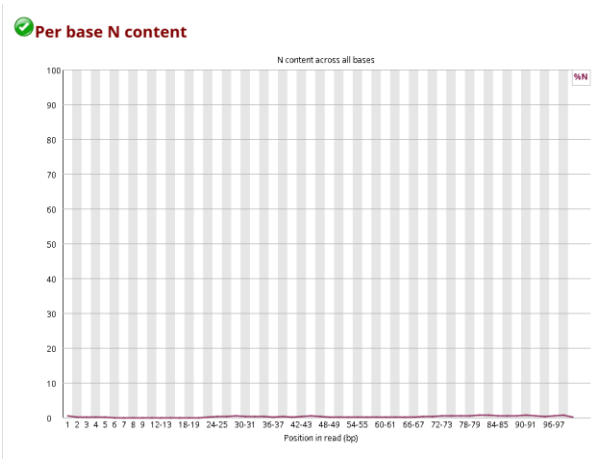
La mayoría de las lecturas presentan también unos scores de calidad adecuados.



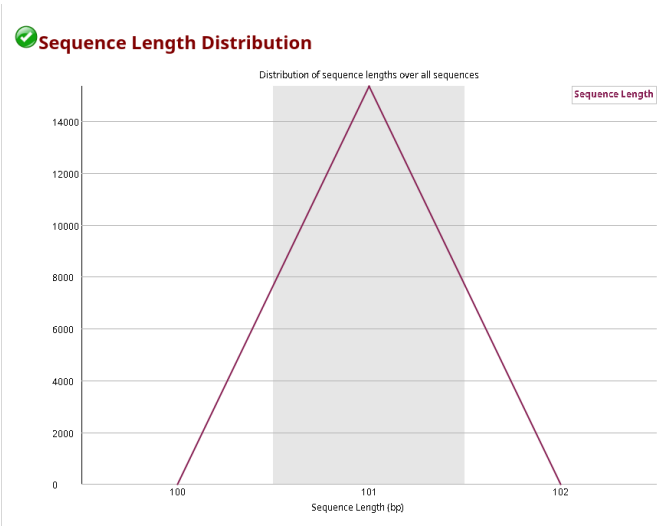
Volvemos a encontrar una imagen una imagen típica de RNAseq y consideramos que el contenido de GC por bases es aceptable.



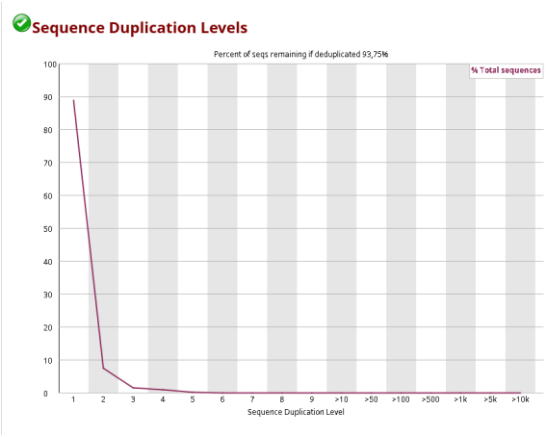
El contenido de GC por lecturas sigue una distribución normal.



Tampoco hay acumulación de bases N.



La longitud de lecturas es igual que en la muestra 1, de 100 a 102 pares de bases.

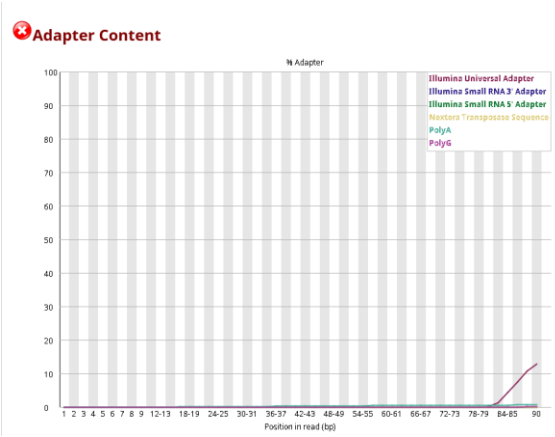


Hay unos niveles bajos de duplicación.

Overrepresented sequences

Sequence	Count	Percentage	Possible Source
CTAACACGTGCGCGAGTCGGGGGCTCGCACGAAAGCCGCCGTGGCGCAAT	20	0.1303780964797914	No Hit

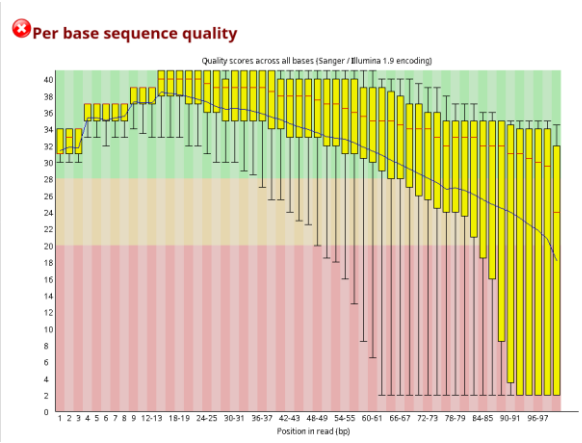
También nos encontramos con una secuencia sobrerrepresentada.



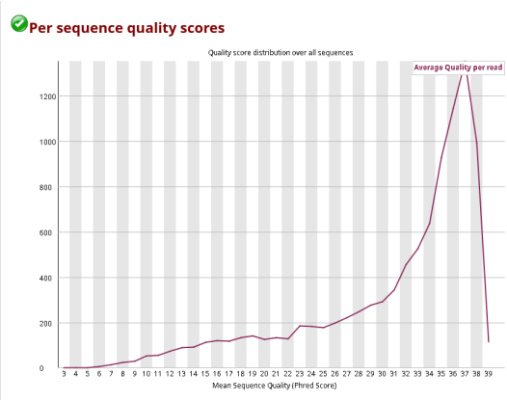
Por último, observamos la presencia de un adaptador universal de illumina.

Fastqc SRR479054.chr21_1.fastq

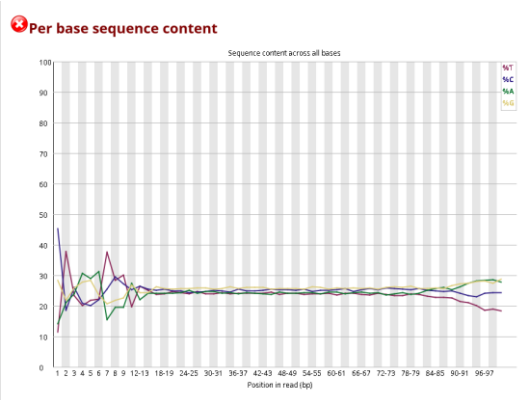
En esta muestra, hay un total de 9.746 lecturas, un total de 984.3 kbp secuenciadas, ninguna de calidad pobre, con una longitud de 101 pares de bases por lectura y un %GC de 51.



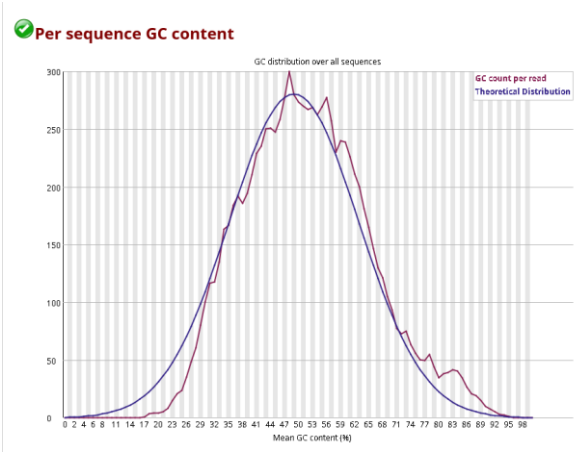
De la misma manera que las anteriores muestras, encontramos una calidad alta hasta la mitad de la secuencia, cayendo considerablemente al final.



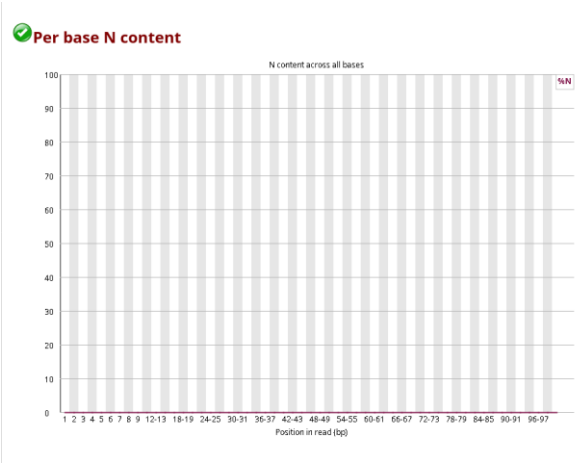
Encontramos scores de calidad elevados en la mayoría de las lecturas.



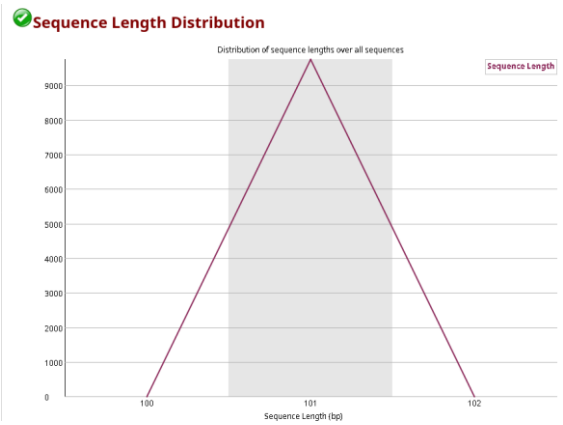
Volvemos a observar alto contenido en GC al principio, pero que consideramos aceptable al ser RNAseq.



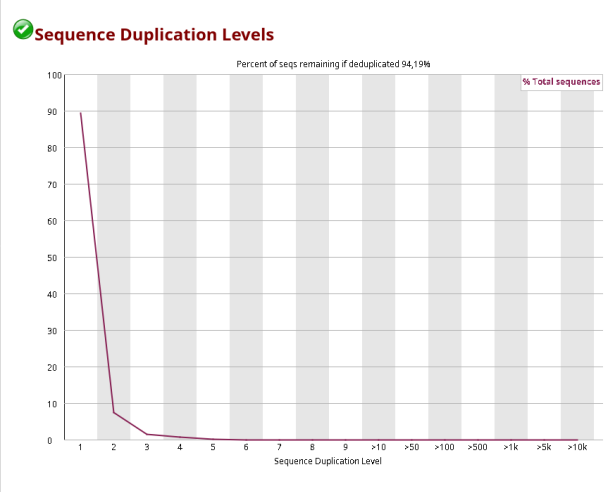
El contenido en GC sigue una distribución normal.



No hay acumulación de bases N, por lo que tampoco hubo problemas en la detección de nucleótidos en esta muestra.

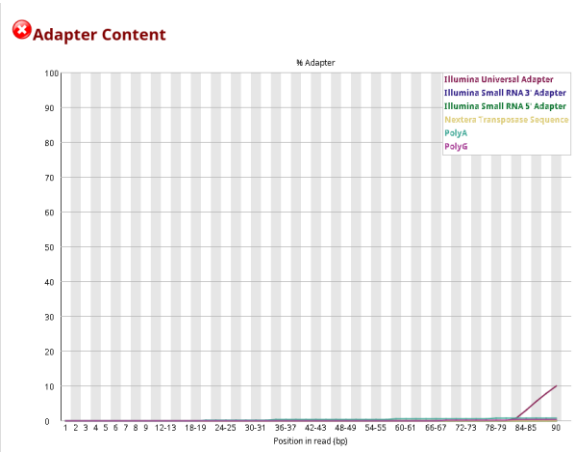


La longitud media de las secuencias es de 101 pares de bases con muy poca desviación.



Encontramos bajos niveles de duplicación.

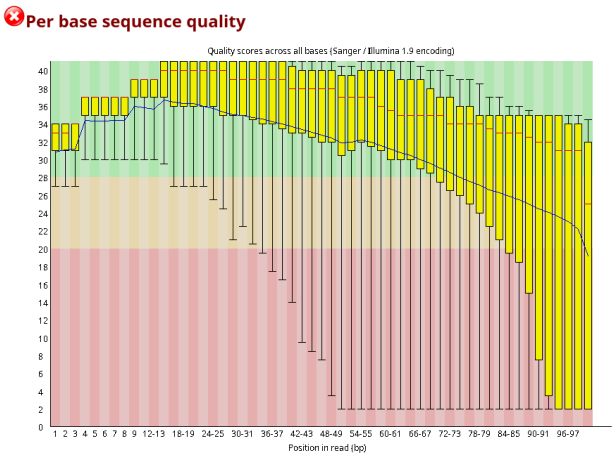
Además, no se han encontrado secuencias sobrerrepresentadas.



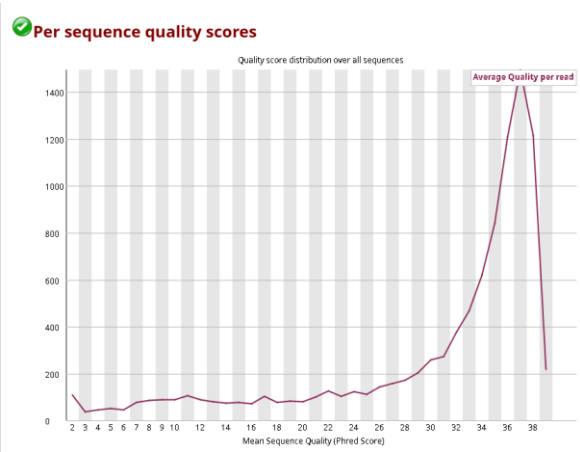
Volvemos a observar la presencia de adaptadores universales de illumina.

Fastqc SRR479054.chr21_2.fastq

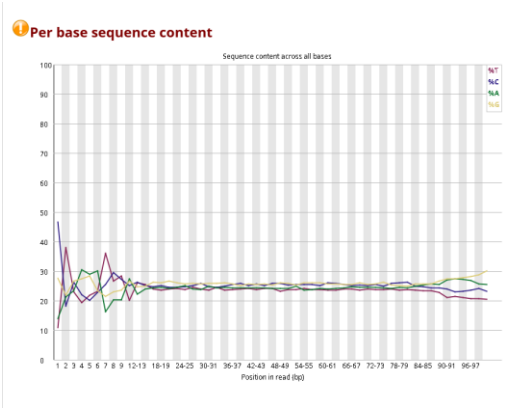
En esta muestra, hay un total de 9.746 lecturas, un total de 984.3 kbp secuenciadas, ninguna de calidad pobre, con una longitud de 101 pares de bases por lectura y un %GC de 51.



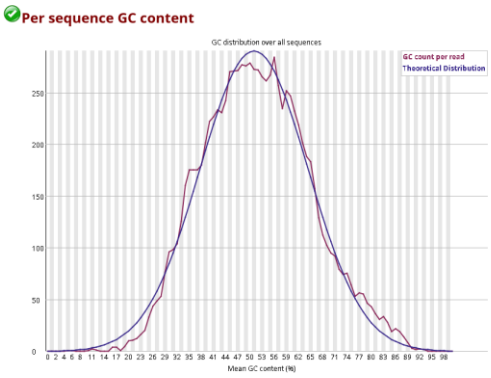
También se observa que la calidad de las lecturas por base cae al final de manera considerable.



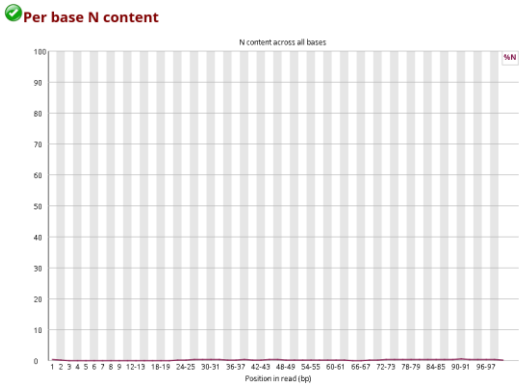
Mientras que los scores de calidad por secuencia son buenos.



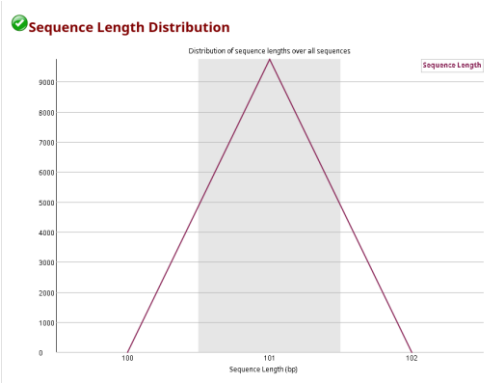
Vuelve a aparecer un contenido en GC alto al principio pero que consideraremos aceptable.



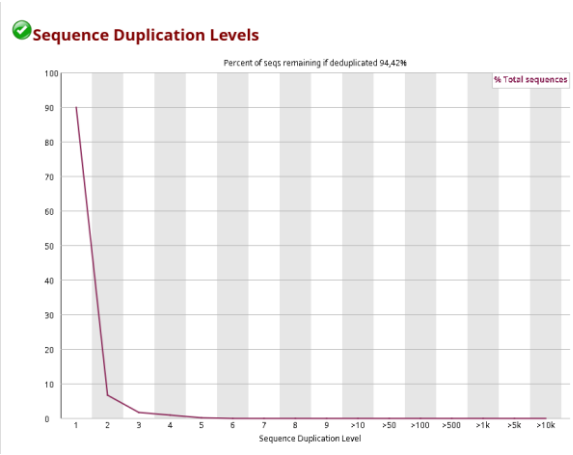
La distribución de GC por lecturas sigue la normal.



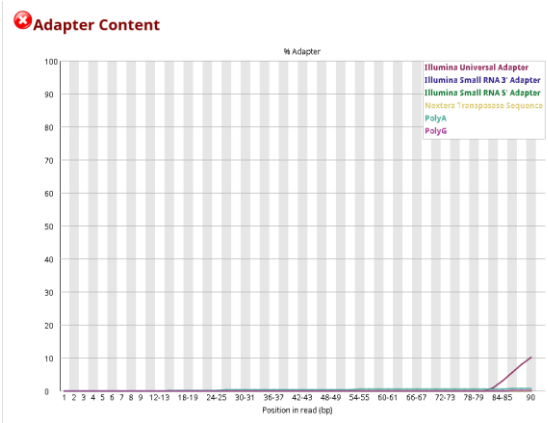
Tampoco hay contenido N, por lo que no hubo problemas de detección de nucleótidos.



Seguimos con una longitud de lecturas media de 101 pares de bases.



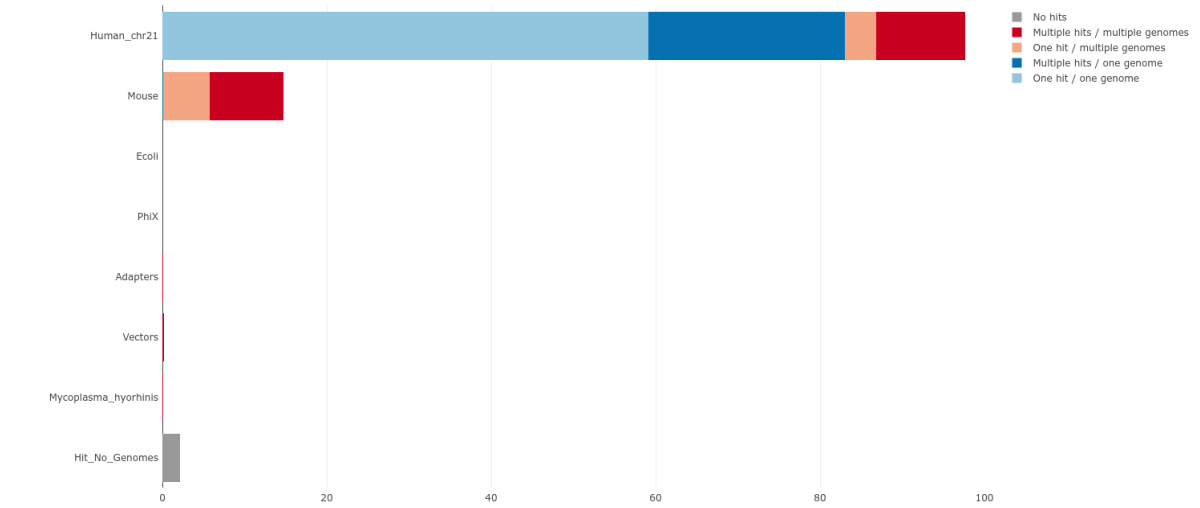
No se han encontrado niveles de duplicación ni secuencias sobrerrepresentadas.



El adaptador universal de illumina también está presente en la secuencia de esta muestra.

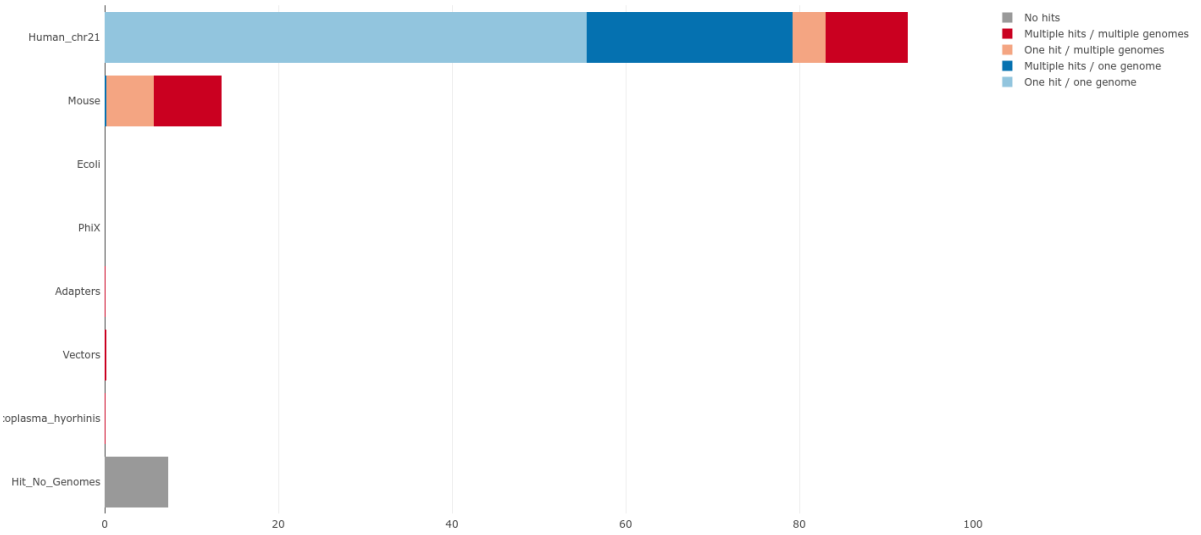
Fastq screen SRR479052.chr21_1.fastq

Para ello, he descargado el programa fastq_screen en primer lugar. Luego he descargado el genoma de mus musculus, ecoli, phix, adapters (illumina), vectors y mycoplasma hyorhinis. Posteriormente, se ha indexado con bowtie2 el fasta del chr21 de humano junto con los fasta de los anteriormente mencionados. Luego he modificado el archivo fastq_screen.conf para que tuviera en cuenta los archivos de indexado generados.



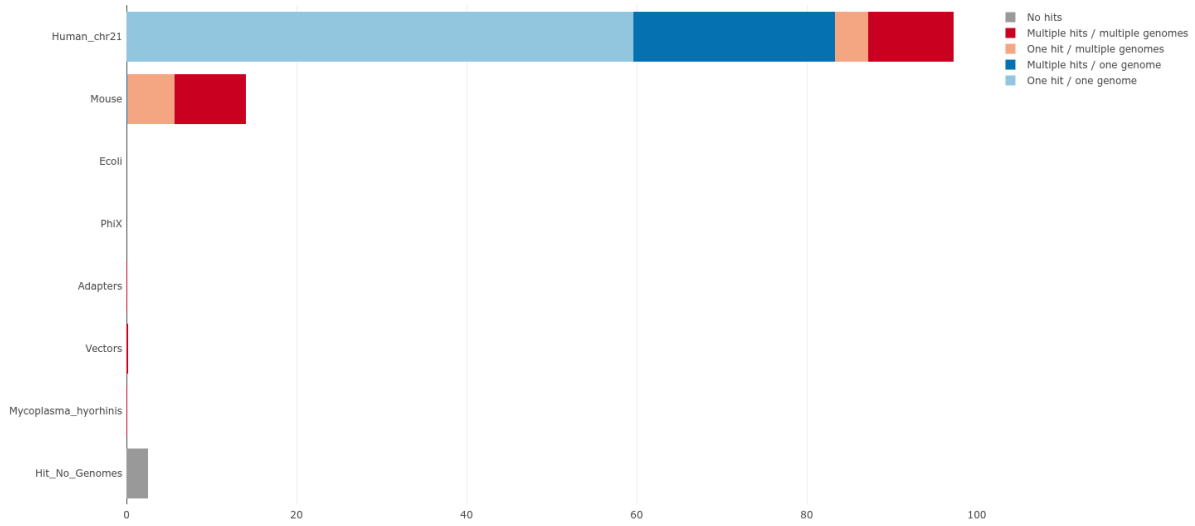
Como se puede observar, el 60% mapea de manera específica como el chr21 del genoma humano, no encontrándose fuentes de contaminación.

Fastq screen SRR479052.chr21_2.fastq



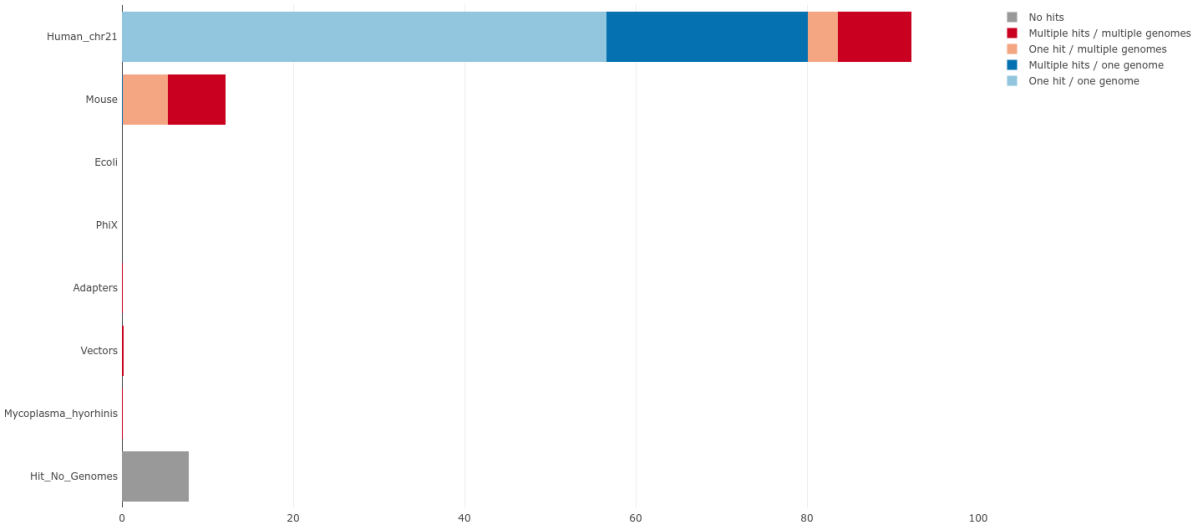
En la segunda muestra ocurre lo mismo, mapeando de manera específica con el genoma humano en un 55%, sin contaminación.

Fastq screen SRR479054.chr21_1.fastq



En la secuencia 1 de SRR479054 tampoco se ha encontrado contaminación, mapeando de manera específica con el chr21 humano en un 59%.

Fastq screen SRR479054.chr21_2.fastq



En la última secuencia de la muestra, ocurre lo mismo. No encontramos contaminación y se mapea de manera específica con el chr21 en un 56%.

Pregunta 2 (1.5 puntos)

Para poder llevar a cabo el alineamiento de las muestras en vuestros ordenadores será necesario trabajar con archivos reducidos correspondientes al cromosoma 21. Se requiere el indexado de la secuencia de este cromosoma, así como el alineamiento de las muestras a dicha referencia. Para ello se podrá utilizar el alineador HISAT2 utilizado en clase u otros (alineadores o pseudoalineadores). Comentar cada uno de los comandos y parámetros empleados, justificando su uso.

Antes de realizar el alineamiento, como en el fastqc hemos visto contenido de adaptadores universales illumina, necesitamos hacer un trimming previo.

Para ello, me he fijado en el trimming utilizado en el Cluster RNA-seq del CNIO-BU.

Por ello, utilizaré BBDuk para hacer el trimming y utilizaré el fasta de adaptadores:

```
(base) vant@MOOVE15:~/Escritorio/transcriptomic-final-exercise/Apartado1/input$ mkdir
trimming
cd trimming/
(base) vant@MOOVE15:~/Escritorio/transcriptomic-final-exercise/Apartado1/input$ wget
https://raw.githubusercontent.com/cnio-
bu/cluster_rnaseq/refs/heads/master/resources/trimming/adapters.fa
```

Luego, activamos nuestro entorno específico para ejecutar BBDuk, incluido en el paquete bbmap de bbtools:

```
(base) vant@MOOVE15:~/Escritorio/transcriptomic-final-exercise/Apartado1/input$ mkdir
trimming
cd trimming/
(transcriptomics-final-exercise) vant@MOOVE15:~/Escritorio/transcriptomic-final-
exercise/Apartado1/input/trimming$ conda activate transcriptomics-final-exercise
```

Después, para ejecutar el trimming con bbduk.sh, he realizado un script llamado trim_bbduk.sh cuyo objetivo es, coger los archivos fasta pareados a partir del nombre de ambas muestras y ejecutar el trimming. El script comentado puede encontrarse en el siguiente enlace: https://github.com/dvalcarcel/transcriptomic-final-exercise/blob/main/Apartado1/input/trim_bbduk.sh

Las estadísticas después del trimming indican:

- SRR479052.chr21: de un total de 30.680 lecturas, han sido eliminadas 1.452 lecturas (4.73%), quedando 29.228 (95.27%).
- SRR479054.chr21: de un total de 19.492 lecturas, han sido eliminadas 972 (4.99%), quedando 18.520 (95.01%).

Ahora sí, vamos a seguir, haciendo primero el indexado del cromosoma 21 con hisat2. Vamos a usar una semilla (seed 123) para asegurar la reproductibilidad. Activamos el

entorno transcriptomics-final-exercise, que permite ejecutar hisat2. Ejecutamos el siguiente código:

```
(transcriptomics-final-exercise) vant@MOOVE15:~/Escritorio/transcriptomic-final-
exercise/Apartado1/input$ hisat2-build --seed 123 -p 2
Homo_sapiens.GRCh38.dna.chromosome.21.fa GRCh38_chr21
```

Como resultado, obtenemos 8 archivos índices del genoma del cromosoma 21, necesarios para hacer el alineamiento con hisat2:

```
(transcriptomics-final-exercise) vant@MOOVE15:~/Escritorio/transcriptomic-final-
exercise/Apartado1/input$ ls *.ht2
GRCh38_chr21.1.ht2 GRCh38_chr21.2.ht2 GRCh38_chr21.3.ht2 GRCh38_chr21.4.ht2
GRCh38_chr21.5.ht2 GRCh38_chr21.6.ht2 GRCh38_chr21.7.ht2 GRCh38_chr21.8.ht2
```

Ahora procedo al realizar el alineamiento con hisat2 usando los índices del cromosoma 21 como referencia, sabiendo que se tratan de unas lecturas unstranded, es decir, sin direccionalidad. Para ello, ejecuto el siguiente código:

```
(transcriptomics-final-exercise) vant@MOOVE15:~/Escritorio/transcriptomic-final-
exercise/Apartado1/input$ mkdir -p hisat2
hisat2 --new-summary --summary-file hisat2/SRR479052.hisat2.summary --seed 123 --phred33 -p
2 -k 1 -x GRCh38_chr21 -1 trimming/SRR479052.chr21_1.trimmed.fastq -2
trimming/SRR479052.chr21_2.trimmed.fastq -S hisat2/SRR479052.sam
hisat2 --new-summary --summary-file hisat2/SRR479054.hisat2.summary --seed 123 --phred33 -p
2 -k 1 -x GRCh38_chr21 -1 trimming/SRR479054.chr21_1.trimmed.fastq -2
trimming/SRR479054.chr21_2.trimmed.fastq -S hisat2/SRR479054.sam
```

Antes de comentar el resultado, procedo a explicar el código. He creado un directorio de salida de hisat2. Genero un archivo de salida "*.summary" que me dará los resultados de las estadísticas del alineamiento tras ejecutar hisat2. Utilizo una semilla para la reproductibilidad. Uso –phred33, que es el sistema de codificación de los últimos pipelines de illumina. Empleo –p 2 para usar 2 hilos y –k 1 para buscar como máximo 1 alineación primaria a cada lectura. Con –x le indico el nombre de los archivos índice del chr21. Con –1 y –2 le proporciono los fastq paired-end y con –S la salida en formato sam.

Para la muestra SRR479052 se ha obtenido una tasa de alineamiento del 92.13%, mientras que la para muestra SRR479054, una tasa de alineamiento del 91.74%.

Ambos resultados indican unos buenos porcentajes de alineamiento con hisat2.

Ahora vamos a realizar tres pasos:

- Convertir los sam a bam: reducir el tamaño del archivo para que sea más eficiente.
- Ordenar los bam: para mejorar el rendimiento de las herramientas que lo vayan a utilizar.

- Indexar los bam ordenados: para el acceso rápido a lecturas específicas en el genoma, sin tener la necesidad de abrir el archivo bam que ocupa más.

Para ello, una vez que tenemos los sam, los convertimos a bam, su formato comprimido. Utilizamos samtools con el siguiente código:

```
(transcriptomics-final-exercise) vant@MOOVE15:~/Escritorio/transcriptomic-final-exercise/Apartado1/input$ cd hisat2
samtools view -bS SRR479052.sam > SRR479052.bam
samtools view -bS SRR479054.sam > SRR479054.bam
```

El siguiente paso es ordenar los archivos .bam como samtools, con el siguiente código:

```
(transcriptomics-final-exercise) vant@MOOVE15:~/Escritorio/transcriptomic-final-exercise/Apartado1/input$ samtools sort SRR479052.bam -o SRR479052.sorted.bam
samtools sort SRR479054.bam -o SRR479054.sorted.bam
```

Por último, realizo el indexado de los bam ordenados mediante samtools con el siguiente código:

```
(transcriptomics-final-exercise) vant@MOOVE15:~/Escritorio/transcriptomic-final-exercise/Apartado1/input$ samtools index SRR479052.sorted.bam
samtools index SRR479054.sorted.bam
```

Pregunta 3 (1.5 puntos)

Una vez generados los archivos alineados se reportarán las estadísticas de alineamiento y se procederá a la cuantificación de la expresión utilizando el archivo GTF correspondiente. Para ello se podrá utilizar HTSeq u otras herramientas de cuantificación. En cualquier caso, detallar y justificar los comandos y parámetros empleados para ello.

Una vez que he generado el archivo bam ordenado, puedo realizar la matriz de cuentas con HTseq. Este paso es necesario para luego realizar la expresión diferencial.

Para ello, ejecuto el siguiente código:

```
(transcriptomics-final-exercise) vant@MOOVE15:~/Escritorio/transcriptomic-final-exercise/Apartado1/input$ mkdir -p htseq
htseq-count --format=bam --stranded=no --mode=intersection-nonempty -a 10 --type=exon --idattr=gene_id hisat2/SRR479052.sorted.bam Homo_sapiens.GRCh38.109.chr21.gtf > htseq/SRR479052.htseq
htseq-count --format=bam --stranded=no --mode=intersection-nonempty -a 10 --type=exon --idattr=gene_id hisat2/SRR479054.sorted.bam Homo_sapiens.GRCh38.109.chr21.gtf > htseq/SRR479054.htseq
```

En cuanto al código, indicamos que el formato es bam y que no tiene direccionalidad en stranded. En el modo, elegimos intersection-nonempty (por defecto se usa union, que

es más laxo), que cuenta si la lectura se superpone con una sola característica y no con múltiples al mismo tiempo. Con –a 10 le indicamos que omita todas las lecturas con una calidad de alineación inferior a 10. En tipo le indicamos que utilice la característica exon para realizar el conteo a través del gtf. En el id del atributo, le indicamos gene_id para identificar los recuentos en la tabla de salida a partir del archivo gtf. Luego como input le damos el archivo ordenado de bam generado por hisat2 y el archivo gtf de anotaciones para el cromosoma 21. Todo lo redireccionamos en un archivo de salida con el nombre de la muestra.

Por otro lado, los resultados del conteo indican que para la muestra SRR479052, de un total de 38.236 líneas procesadas, 18.531 pares de alineación fueron procesadas, mientras que para la muestra SRR479054, de un total de 38.236 líneas, fueron procesadas 11.528 pares de alineamiento.

Apartado 2

En este repositorio se proporcionan todos los inputs del Apartado 2:

- La matriz de cuentas crudas de los 24 cultivos analizados.
- Data frame con los metadatos asociados al experimento.
- GMT para realizar un GSEA.

Pregunta 4 (3 puntos)

¿Qué genes se encuentran diferencialmente expresados entre las muestras pertenecientes al grupo tratado con OHT con respecto al control tras 24h? ¿Y en el caso de las muestras tratadas con DPN, también tras 24h? Como parte de la respuesta a esta pregunta, podéis entregar una o más tablas adjuntas donde se incluyan los genes diferencialmente expresados, indicando el criterio o los criterios que habéis seguido para filtrar los resultados, así como cualquier otro gráfico o gráficos que hayáis elaborado durante el análisis.

El script de r cumplimentado con los códigos necesarios para realizar el análisis y comentado, así como los gráficos y archivos generados, se pueden encontrar en el siguiente enlace: https://github.com/dvalcarcel/transcriptomic-final-exercise/tree/main/Apartado2/Pregunta%204

Además, para realizar la expresión diferencial o DEG con DESeq2, he tenido en cuenta las instrucciones de la vignette de este programa para RNAseq: https://bioconductor.org/packages/devel/bioc/vignettes/DESeq2/inst/doc/DESeq2.html

En primer lugar, cargamos las siguientes librerías de paquetes de R o Bioconductor. En el caso de no tenerlas, en el script he proporcionado como obtenerlas:

```
> library("readr")
> library("DESeq2")
> library("pheatmap")
> library("RColorBrewer")
> library("EnhancedVolcano")
> library("mygene")
```

En cuanto a los archivos, contamos con la matriz de cuentas crudas y los metadatos del experimento. Los cargamos en R y realizamos el pre-procesado de los datos.

Convertimos a factores las columnas de los metadatos del experimento para que lo pueda manejar DESeq2. Después comprobamos que las muestran se encuentran en ambas matrices y en el mismo orden.

Dado que en la matriz del experimento contamos con tres columnas (paciente, tratamiento y tiempo), necesitamos crear una variable llamada grupo que agrupe el tratamiento en los momentos del tiempo, para poder responder a las preguntas de este apartado.

Ahora creamos el objeto DESeq teniendo en cuenta la matriz de cuentas crudas, el metadato del experimento, ajustando el modelo por paciente y como factor de interés el grupo, es decir, el tratamiento en los distintos momentos del tiempo:

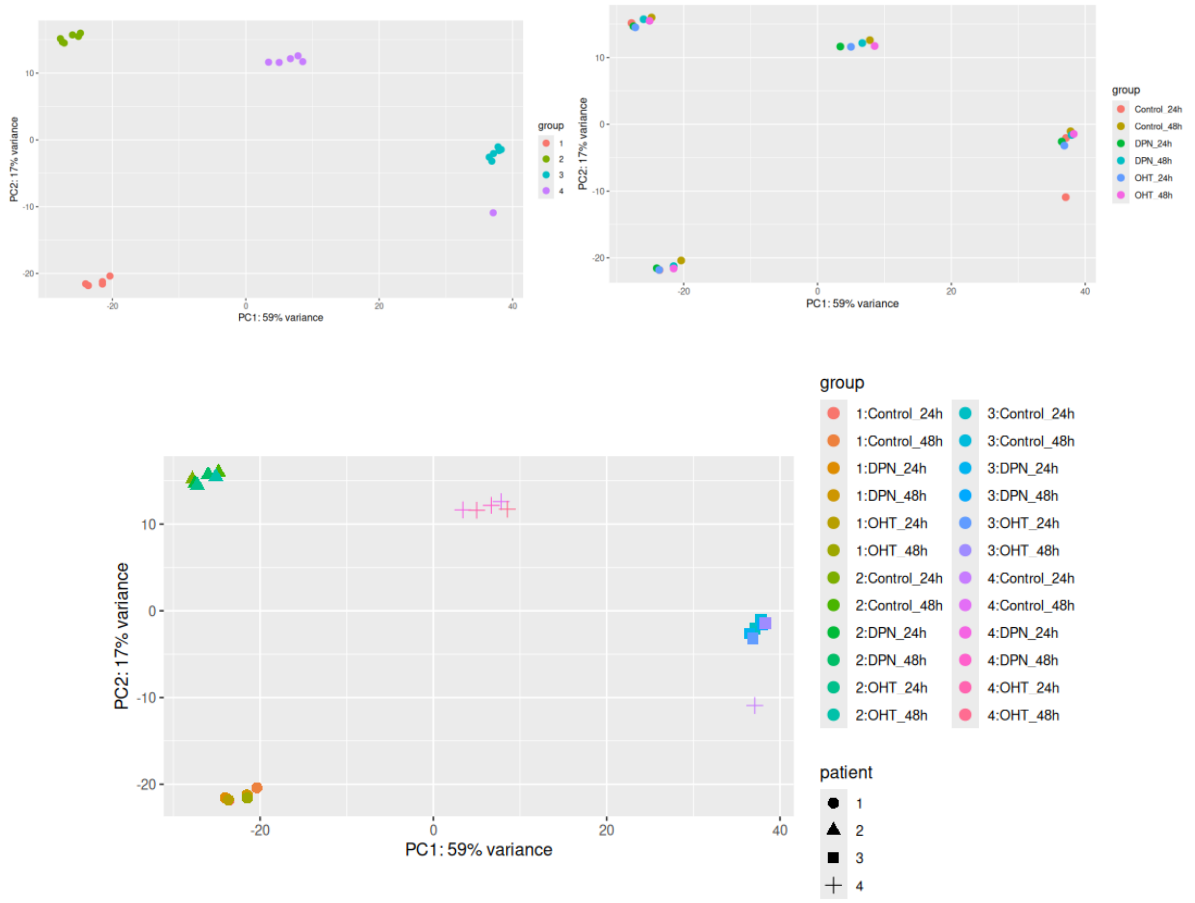
```
> dds <- DESeqDataSetFromMatrix(countData = cts,
+                               colData = coldata,
+                               design = ~ patient + group)
> dds
class: DESeqDataSet
dim: 53160 24
metadata(1): version
assays(1): counts
rownames(53160): ENSG00000000003 ENSG00000000005 ... ENSG00000257111 ENSG00000257112
rowData names(0):
colnames(24): GSM913873 GSM913874 ... GSM913895 GSM913896
colData names(4): patient agent time group
```

Se trata de un objeto de clase DESeqDataSet, especializado para realizar la expresión diferencial. Contamos con 53.160 filas que son los ID de los genes y 24 columnas, que son los ID de las muestras. Además, contamos con 4 columnas procedentes del metadato experimental (paciente, tratamiento, tiempo y grupo). Por lo tanto, contamos con conteos crudos por gen y por muestra.

Posteriormente, realizamos el análisis exploratorio de los datos. Para ello, vamos a normalizar y estabilizar la varianza de las cuentas mediante vst (variance stabilizing transformation), para luego explorar los datos con un análisis de componentes principales o PCA:

```
> vsd <- vst(dds, blind = TRUE)
> plotPCA(vsd, intgroup = "patient")
```

```
using ntop=500 top features by variance
> plotPCA(vsd, intgroup = "group")
using ntop=500 top features by variance
> pcaData <- plotPCA(vsd, intgroup=c("patient", "group"), returnData=TRUE)
using ntop=500 top features by variance
> percentVar <- round(100 * attr(pcaData, "percentVar"))
> ggplot(pcaData, aes(PC1, PC2, color=group, shape=patient)) +
+  geom_point(size=3) +
+  xlab(paste0("PC1: ",percentVar[1],"% variance")) +
+  ylab(paste0("PC2: ",percentVar[2],"% variance")) +
+  coord_fixed()
```

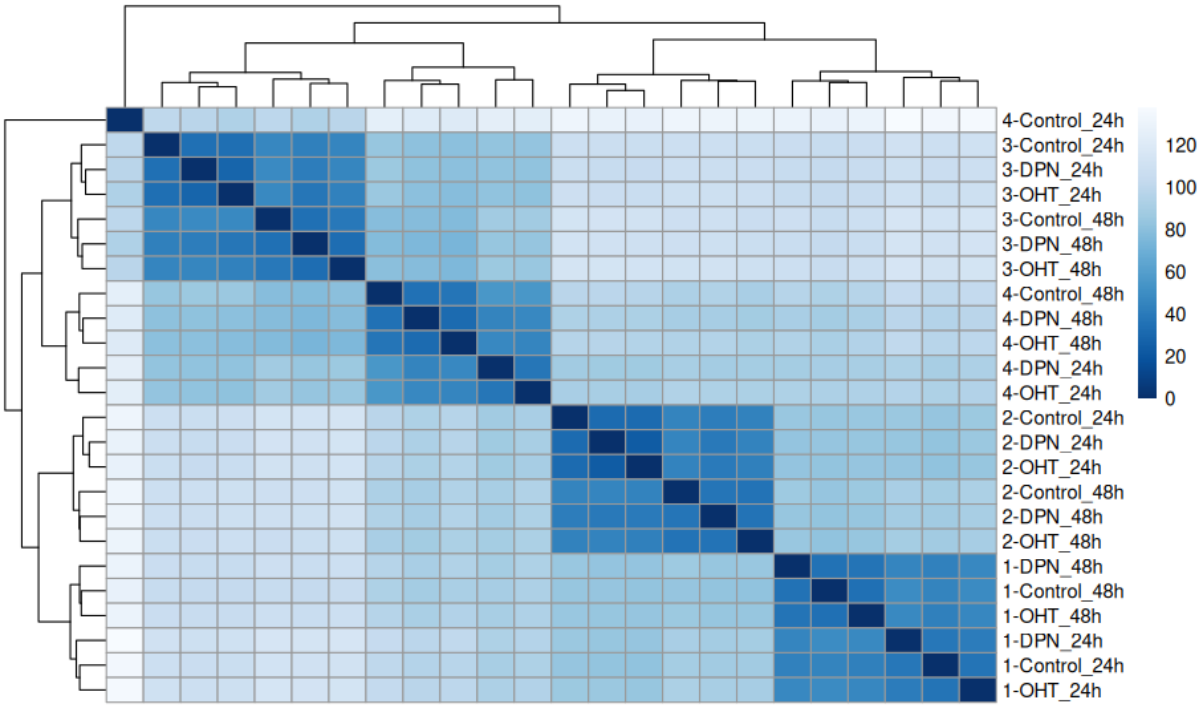


Se observa que el PC1 explica el 59% de la varianza y el PC2 el 17%. En el gráfico se puede observar que los pacientes se separan unos de otros, salvo una muestra del paciente 4. El PC1 diferencia bien los pacientes 1 y 2 del 3 y 4; mientras que el PC2 diferencia el paciente 1 y 3 del 2 y 4. En este plot, no vemos una clara diferencia entre tratamientos. Vemos la misma distribución por pacientes.

Después, se calcula la matriz de distancias a partir de las cuentas normalizadas (vst), para analizar cuanto se separa cada paciente (muestra – grupo). Lo visualizamos mediante un heatmap:

```
> #Transponemos los valores, calculamos las distancias y lo convertimos
> # a matriz:
> sampleDists <- dist(t(assay(vsd)))
> sampleDistMatrix <- as.matrix(sampleDists)
```

```
> # Ponemos los nombres del grupo en las filas y eliminamos los nombres
> # de las columnas.
> rownames(sampleDistMatrix) <- paste(vsd$patient, vsd$group, sep="-")
> colnames(sampleDistMatrix) <- NULL
> # Creamos el heatmap:
> colors <- colorRampPalette( rev(brewer.pal(9, "Blues")) )(255)
> pheatmap(sampleDistMatrix,
+          clustering_distance_rows=sampleDists,
+          clustering_distance_cols=sampleDists,
+          col=colors)
```



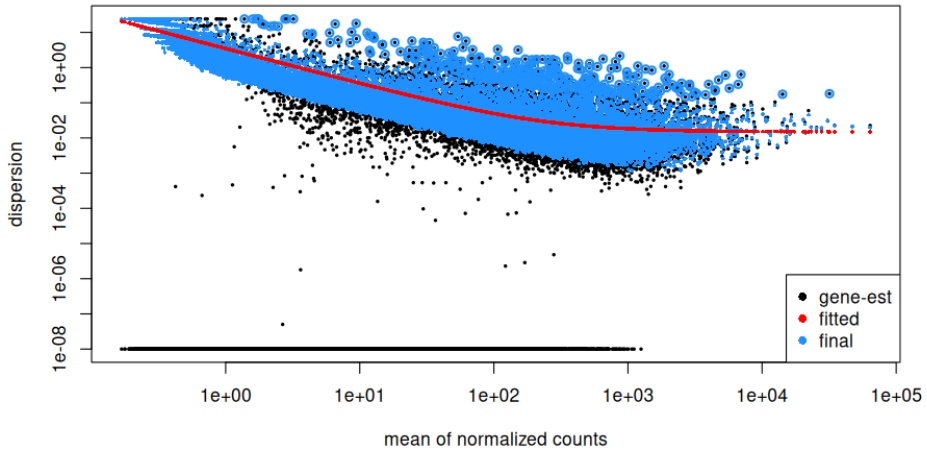
En el heatmap volvemos a ver que las muestras de un mismo paciente tienden a agruparse juntas.

Vamos a correr la función DESeq para realizar todos los pasos de DESeq2 y por consiguiente la expresión diferencial. Además, realizaremos el modelo lineal teniendo en cuenta el test de Wald.

```
> dds2 <- DESeq(dds, test = "Wald")
estimating size factors
estimating dispersions
gene-wise dispersion estimates
mean-dispersion relationship
final dispersion estimates
fitting model and testing
```

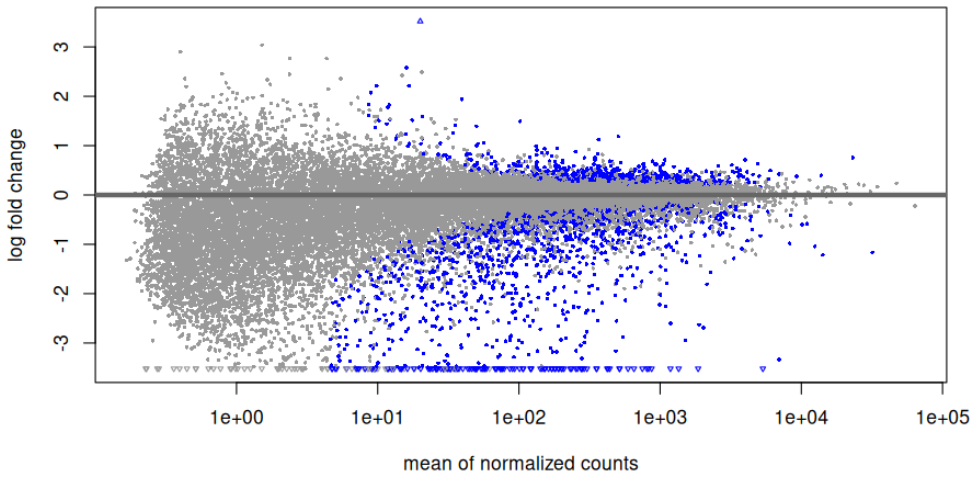
Realizamos la estimación de la dispersión mediante un gráfico.

```
> plotDispEsts(dds2)
```



Ahora vamos a realizar el plotMa. En DESeq2, la función plotMA muestra los cambios de log2 fold atribuibles a una variable dada sobre la media de recuentos normalizados para todas las muestras del DESeqDataSet. Los puntos se colorearán de azul si el valor p ajustado es inferior a 0,1. Los puntos que quedan fuera de la ventana se representan como triángulos abiertos que apuntan hacia arriba o hacia abajo.

```
> plotMA(dds2)
```



Observamos que cuanto mayor es la media, mayor es el número de cambios significativos (azul).

DEG de genes para el tratamiento con OHT vs Control 24h

Primero vamos a obtener la lista de genes DEG para OHT vs Control a las 24h:

```
> # Obtención de nuestra lista de genes DEG para OHT vs Control a las 24h:
> results_OHT_vs_Control_24h <- results(object = dds2,
+                       contrast = c("group","OHT_24h", "Control_24h"),
+                       alpha = 0.05,
+                       pAdjustMethod = "BH",
+                       tidy = TRUE
+ )
> head(results_OHT_vs_Control_24h)
              row   baseMean log2FoldChange      lfcSE       stat     pvalue      padj
1 ENSG00000000003 561.354460     0.12911065 0.08862672  1.4567915 0.14517394 0.5030496
2 ENSG00000000005   0.836186    -0.50376607 2.10671665 -0.2391238 0.81100960        NA
3 ENSG00000000419 244.052822     0.05239402 0.16921852  0.3096234 0.75684732 0.9087698
4 ENSG00000000457 190.404159    -0.23389397 0.11661347 -2.0057199 0.04488614 0.3646613
5 ENSG00000000460 344.140157    -0.17867779 0.22908194 -0.7799733 0.43540662 0.7406712
6 ENSG00000000938  11.129902    -0.68102376 0.50981847 -1.3358162 0.18160936 0.5420506
```

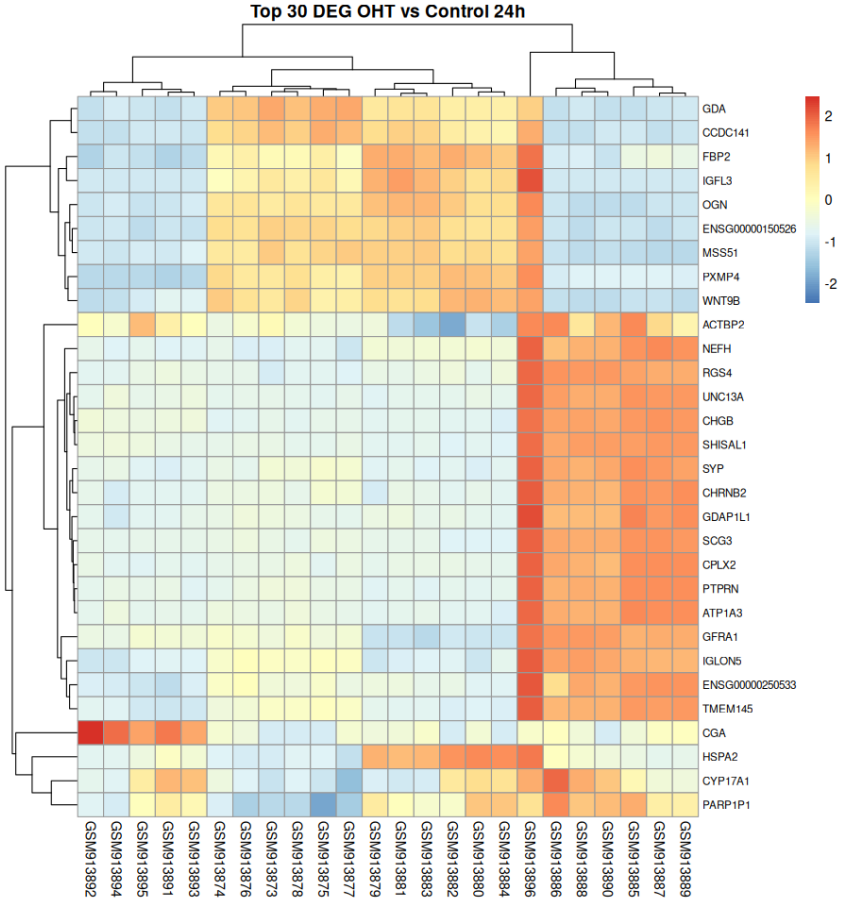
Ahora vamos a anotar los genes para mostrar los símbolos para facilitar el estudio biológico y refinamos quitando duplicados y NAs:

```
> genesID <-mygene::queryMany(results_OHT_vs_Control_24h$row, scopes="ensembl.gene",
fields="symbol", species="human")
Querying chunk 1
Querying chunk 2
Querying chunk 3
Querying chunk 4
Querying chunk 5
Querying chunk 6
Querying chunk 7
Querying chunk 8
Querying chunk 9
Querying chunk 10
Querying chunk 11
Querying chunk 12
Querying chunk 13
Querying chunk 14
Querying chunk 15
Querying chunk 16
Querying chunk 17
Querying chunk 18
Querying chunk 19
Querying chunk 20
Querying chunk 21
Querying chunk 22
Querying chunk 23
Querying chunk 24
Querying chunk 25
Finished
Pass returnall=TRUE to return lists of duplicate or missing query terms.
> genesID <- genesID[!duplicated(genesID$query),]
> results_OHT_vs_Control_24h$row <- ifelse(is.na(genesID$symbol),genesID$query,genesID$symbol)
> head(results_OHT_vs_Control_24h)
     row   baseMean log2FoldChange      lfcSE       stat     pvalue      padj
1 TSPAN6 561.354460     0.12911065 0.08862672  1.4567915 0.14517394 0.5030496
2   TNMD   0.836186    -0.50376607 2.10671665 -0.2391238 0.81100960        NA
3   DPM1 244.052822     0.05239402 0.16921852  0.3096234 0.75684732 0.9087698
4  SCYL3 190.404159    -0.23389397 0.11661347 -2.0057199 0.04488614 0.3646613
```

```
5  FIRRM 344.140157    -0.17867779 0.22908194 -0.7799733 0.43540662 0.7406712
6    FGR  11.129902    -0.68102376 0.50981847 -1.3358162 0.18160936 0.5420506
```

Realizamos el Heatmap de los 30 genes TOP DGE por p-valor ajustado para OHT vs Control 24h con los genes anotados por los símbolos de ensembl.

```
> gene_symbols <- ifelse(is.na(genesID$symbol), genesID$query, genesID$symbol)
> rownames(vsd) <- gene_symbols
> top30_OHT_vs_Control_24h <- head(order(results_OHT_vs_Control_24h$padj), 30)
> mat_OHT_Control <- assay(vsd)[top30_OHT_vs_Control_24h, ]
> pheatmap(mat_OHT_Control,
+          scale = "row",
+          main = "Top 30 DEG OHT vs Control 24h",
+          fontsize_row = 8)
```



Se observa una mayor expresión del gen CGA para la muestra GSM913892, así como una mayor expresión en la mayoría de los genes de la muestra GSM913896. También se ha encontrado una mayor expresión en un set de genes en las muestras GSM913886, GSM913888, GSM913890, GSM913885, GSM913887 y GSM913889.

Ahora vamos a filtrar la DEG por los genes que son estadísticamente significativos en el p ajustados y los vamos a ordenar según su log2FoldChange:

```
> deg_filtered_OHT_vs_Control_24h <- results_OHT_vs_Control_24h[results_OHT_vs_Control_24h$padj <
0.05, ]
> deg_filtered_OHT_vs_Control_24h_clean <- na.omit(deg_filtered_OHT_vs_Control_24h)
```

```
> print(deg_filtered_OHT_vs_Control_24h_clean$row)
 [1] "PTPRN"           "MAOB"            "CHGB"            "NEFH"            "PXMP4"           "SYP"
"SCG3"            "ATP1A3"
 [9] "OGN"             "RGS4"            "GDA"             "SLC12A5"         "GDAP1L1"
"HSPA2"           "CBFA2T3"         "UNC13A"
[17] "FBP2"            "CGA"             "PPP1R1A"         "SHISAL1"         "IGLON5"
"CPLX2"           "CYP17A1"         "INA"
[25] "ENSG00000150526" "DLG2"            "GFRA1"           "WNT9B"           "CHRNB2"
"CCDC141"         "FREM1"           "MSS51"
[33] "TMEM145"         "CFL1"            "CD24P4"          "IGFL3"           "PTPRT"
"DPP4"            "HLA-DRB5"        "ACTBP2"
[41] "PARP1P1"         "EEF1DP1"         "PRB4"            "AKAIN1"          "LINC00649"
"PCDHA10"         "ENSG00000250533" "THAP9-AS1"
> nrow(deg_filtered_OHT_vs_Control_24h_clean)
[1] 48
> deg_filtered_OHT_vs_Control_24h_clean <-
+  deg_filtered_OHT_vs_Control_24h_clean[order(deg_filtered_OHT_vs_Control_24h_clean$log2FoldChange,
+                                              decreasing = TRUE), ]
> head(deg_filtered_OHT_vs_Control_24h_clean)
            row    baseMean log2FoldChange      lfcSE      stat       pvalue       padj
6571        CGA    8.928703      3.3835481 0.72634467  4.658323 3.187957e-06 0.01493160
5775     CBFA2T3    9.853864      2.6898589 0.67467533  3.986894 6.694398e-05 0.03494443
15533  HLA-DRB5   35.342116      1.7942678 0.44519191  4.030324 5.569995e-05 0.03237335
23217 THAP9-AS1   82.713786      0.6492875 0.16704678  3.886860 1.015491e-04 0.04323915
12135      CFL1 1419.039144     -0.2008332 0.05154812 -3.896035 9.778046e-05 0.04260272
19139   EEF1DP1  270.154828     -0.3586266 0.09197170 -3.899314 9.646544e-05 0.04260272
```

Observamos que hay 48 genes que se encuentran diferencialmente expresados entre las muestras pertenecientes al grupo tratado con OHT con respecto al control tras 24h.

Los genes CGA, CBFA2T3 y HLA-DRB5 se encuentran más expresados en las muestras con OHT frente al control a las 24h.

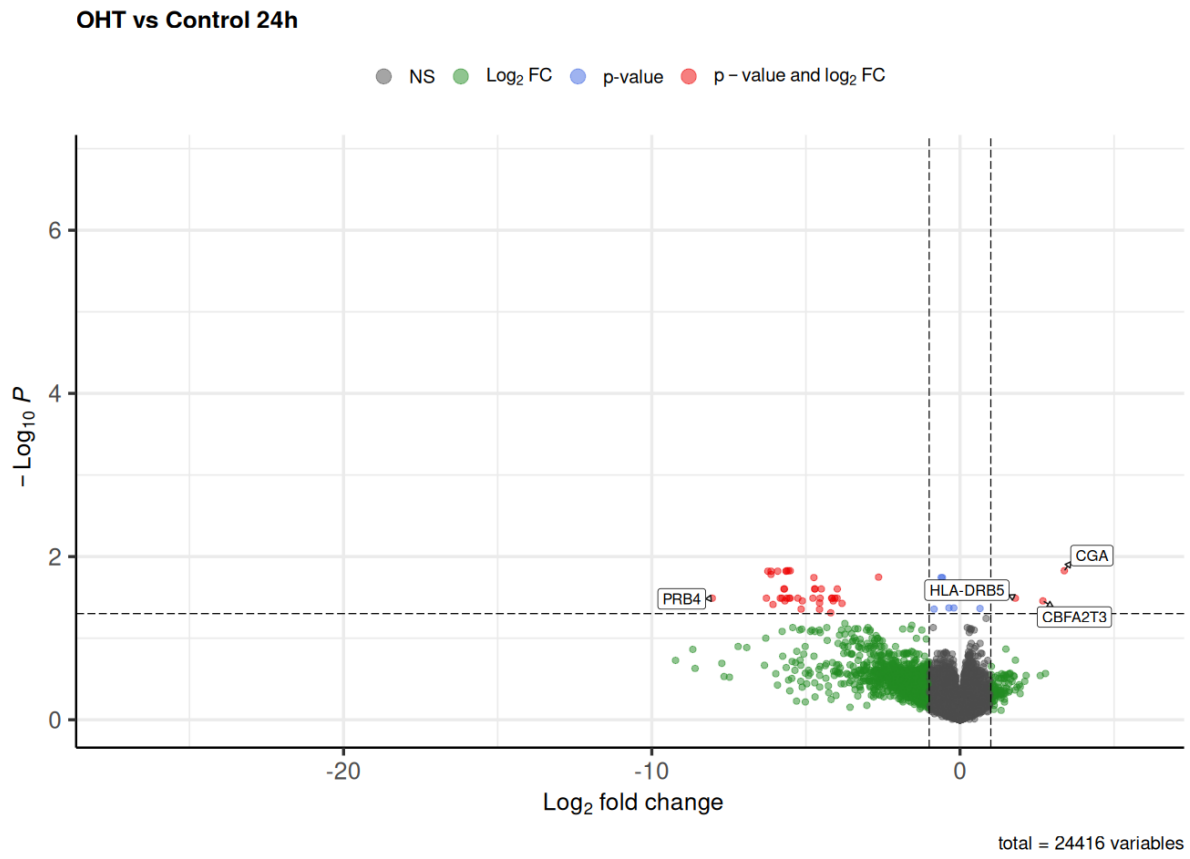
Los genes PRB4, GDA y NEFH se encuentran menos expresados en las muestras con OHT frente al control a las 24h.

Guardamos la tabla para añadirla al informe:

```
> write.csv(deg_filtered_OHT_vs_Control_24h_clean,
+           "DEG_filtered_OHT_vs_Control_24h.csv", row.names = FALSE)
```

Creamos el volcano plot para visualizarlos:

```
> EnhancedVolcano(results_OHT_vs_Control_24h,
+                 lab = results_OHT_vs_Control_24h$row,
+                 x = "log2FoldChange",
+                 y = "padj",
+                 title = "OHT vs Control 24h",
+                 FCcutoff = 1,
+                 pCutoff = 0.05,
+                 subtitle = NULL,
+                 boxedLabels = TRUE,
+                 drawConnectors = TRUE,
+                 labSize = 4.0)
```



DEG de genes para el tratamiento con DPN vs Control 24h

Primero vamos a obtener la lista de genes DEG para OHT vs Control a las 24h:

```
> results_DPN_vs_Control_24h <- results(object = dds2,
+                                       contrast = c("group","DPN_24h", "Control_24h"),
+                                       alpha = 0.05,
+                                       pAdjustMethod = "BH",
+                                       tidy = TRUE
+ )
> head(results_DPN_vs_Control_24h)
              row   baseMean log2FoldChange      lfcSE       stat    pvalue      padj
1 ENSG00000000003 561.354460     0.14407597 0.08903873  1.6181270 0.1056352 0.3994310
2 ENSG00000000005   0.836186    -0.95278333 2.12949014 -0.4474232 0.6545695        NA
3 ENSG00000000419 244.052822     0.06032924 0.16962228  0.3556681 0.7220891 0.8771143
4 ENSG00000000457 190.404159    -0.23675825 0.11779046 -2.0099951 0.0444317 0.2820864
5 ENSG00000000460 344.140157    -0.27831811 0.22956326 -1.2123809 0.2253666 0.5455183
6 ENSG00000000938  11.129902    -0.53598702 0.51064445 -1.0496286 0.2938889        NA
```

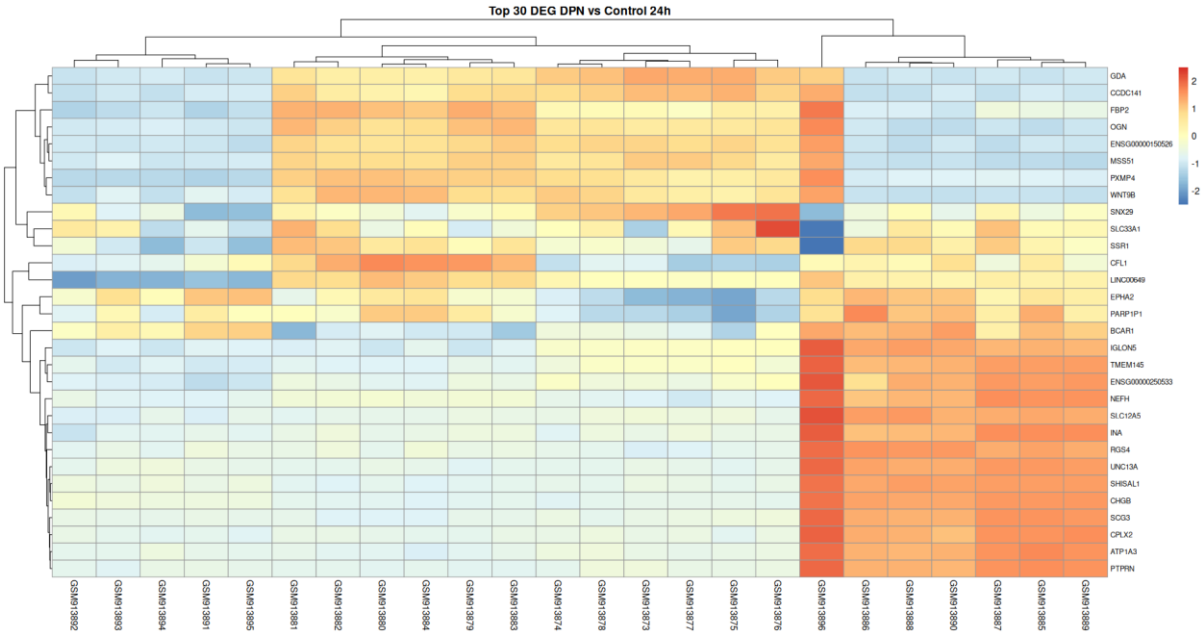
Ahora vamos a anotar los genes para mostrar los símbolos para facilitar el estudio biológico y refinamos quitando duplicados y NAs:

```
> genesID_DPN <-mygene::queryMany(results_DPN_vs_Control_24h$row, scopes="ensembl.gene",
fields="symbol", species="human")
Querying chunk 1
Querying chunk 2
Querying chunk 3
```

```
Querying chunk 4
Querying chunk 5
Querying chunk 6
Querying chunk 7
Querying chunk 8
Querying chunk 9
Querying chunk 10
Querying chunk 11
Querying chunk 12
Querying chunk 13
Querying chunk 14
Querying chunk 15
Querying chunk 16
Querying chunk 17
Querying chunk 18
Querying chunk 19
Querying chunk 20
Querying chunk 21
Querying chunk 22
Querying chunk 23
Querying chunk 24
Querying chunk 25
Finished
Pass returnall=TRUE to return lists of duplicate or missing query terms.
> genesID_DPN <- genesID_DPN[!duplicated(genesID_DPN$query),]
> results_DPN_vs_Control_24h$row <-
ifelse(is.na(genesID_DPN$symbol),genesID_DPN$query,genesID_DPN$symbol)
> head(results_DPN_vs_Control_24h)
     row   baseMean log2FoldChange      lfcSE       stat    pvalue      padj
1 TSPAN6 561.354460     0.14407597 0.08903873  1.6181270 0.1056352 0.3994310
2   TNMD   0.836186    -0.95278333 2.12949014 -0.4474232 0.6545695        NA
3   DPM1 244.052822     0.06032924 0.16962228  0.3556681 0.7220891 0.8771143
4  SCYL3 190.404159    -0.23675825 0.11779046 -2.0099951 0.0444317 0.2820864
5  FIRRM 344.140157    -0.27831811 0.22956326 -1.2123809 0.2253666 0.5455183
6    FGR  11.129902    -0.53598702 0.51064445 -1.0496286 0.2938889        NA
```

Realizamos el Heatmap de los 30 genes TOP DGE por p-valor ajustado para DPN vs Control 24h con los genes anotados por los símbolos de ensembl.

```
> gene_symbols_DPN <- ifelse(is.na(genesID_DPN$symbol), genesID_DPN$query, genesID_DPN$symbol)
> rownames(vsd) <- gene_symbols_DPN
> top30_DPN_vs_Control_24h <- head(order(results_DPN_vs_Control_24h$padj), 30)
> mat_DPN_Control <- assay(vsd)[top30_DPN_vs_Control_24h, ]
> pheatmap(mat_DPN_Control,
+          scale = "row",
+          main = "Top 30 DEG DPN vs Control 24h",
+          fontsize_row = 8)
```



Se observa una mayor expresión de un conjunto de genes para la muestra GSM913896, así como una mayor expresión en el mismo conjunto de genes de las muestras GSM913886, GSM913888, GSM913890, GSM913885, GSM913887 y GSM913889. De manera puntual, el gen SLC33A1 se expresa en gran medida en la muestra GSM913876.

Ahora vamos a filtrar la DEG por los genes que son estadísticamente significativos en el p ajustados y los vamos a ordenar según su log2FoldChange:

```
> deg_filtered_DPN_vs_Control_24h <- results_DPN_vs_Control_24h[results_DPN_vs_Control_24h$padj <
0.05, ]
> deg_filtered_DPN_vs_Control_24h_clean <- na.omit(deg_filtered_DPN_vs_Control_24h)
> print(deg_filtered_DPN_vs_Control_24h_clean$row)
 [1] "CYP26B1"         "SNX29"           "BCAR1"           "PTPRN"           "MAOB"
"CACNG4"          "CTTNBP2"         "AMPH"
 [9] "CHGB"            "G2E3"            "SCD"             "NEFH"            "PXMP4"           "SYP"
"SCG3"            "CLIP3"
[17] "ATP1A3"          "OGN"             "RUNDC3A"         "ENSG00000108753" "GCNT2"
"ACAP2"           "RGS4"            "GDA"
[25] "DUSP1"           "SLC12A5"         "GDAP1L1"         "SSR1"            "VASP"
"KIRREL2"         "HSPA2"           "UNC13A"
[33] "OLFM1"           "FBP2"            "SIX3"            "SHISAL1"         "IGLON5"
"EPHA2"           "CPLX2"           "CLTRN"
[41] "CYP17A1"         "INA"             "ENSG00000150526" "DLG2"            "GFRA1"
"NMNAT2"          "KCNJ15"          "ACAN"
[49] "WNT9B"           "ZNF761"          "CHRNB2"          "VCAM1"           "ACTG2"
"CCDC141"         "NFASC"           "FREM1"
[57] "MSS51"           "TMEM145"         "RAB24"           "CA5B"            "SLC33A1"
"CDK5R2"          "CFL1"            "GPX2"
[65] "RIMS2"           "ARSJ"            "CD24P4"          "IGFL1"           "CFAP126"
"PTPRT"           "S100A4"          "IPO4"
[73] "SLC2A10"         "DPP4"            "ZNF84"           "LINC02397"       "TTC3P1"
"EXOSC6"          "PARP1P1"         "PRB4"
[81] "AKAIN1"          "LINC03002"       "ZNF503-AS2"      "LINC00649"       "PCDHA10"
"ENSG00000250533" "THAP9-AS1"
> nrow(deg_filtered_DPN_vs_Control_24h_clean)
[1] 87
> deg_filtered_DPN_vs_Control_24h_clean <-
+  deg_filtered_DPN_vs_Control_24h_clean[order(deg_filtered_DPN_vs_Control_24h_clean$log2FoldChange,
+                                          decreasing = TRUE), ]
```

```
> head(deg_filtered_DPN_vs_Control_24h_clean)
           row   baseMean log2FoldChange     lfcSE     stat       pvalue       padj
23217 THAP9-AS1   82.71379      0.6151513 0.1686768 3.646924 2.653986e-04 0.04602965
1308    CTTNBP2  278.74201      0.6126441 0.1561525 3.923369 8.731908e-05 0.02684665
1985        SCD 4929.51885      0.5632706 0.1465170 3.844404 1.208456e-04 0.02886538
9684     ZNF761  189.32943      0.4562415 0.1219745 3.740466 1.836794e-04 0.03601181
14804   CFAP126  150.51652      0.4277350 0.1151917 3.713245 2.046185e-04 0.03850075
15420     ZNF84  325.70826      0.4186500 0.1117498 3.746314 1.794519e-04 0.03570036
```

Observamos que hay 87 genes que se encuentran diferencialmente expresados entre las muestras pertenecientes al grupo tratado con DPN con respecto al control tras 24h.

Los genes THAP9-AS1, CTTNBP2 y SCD se encuentran más expresados en las muestras con DPN frente al control a las 24h.

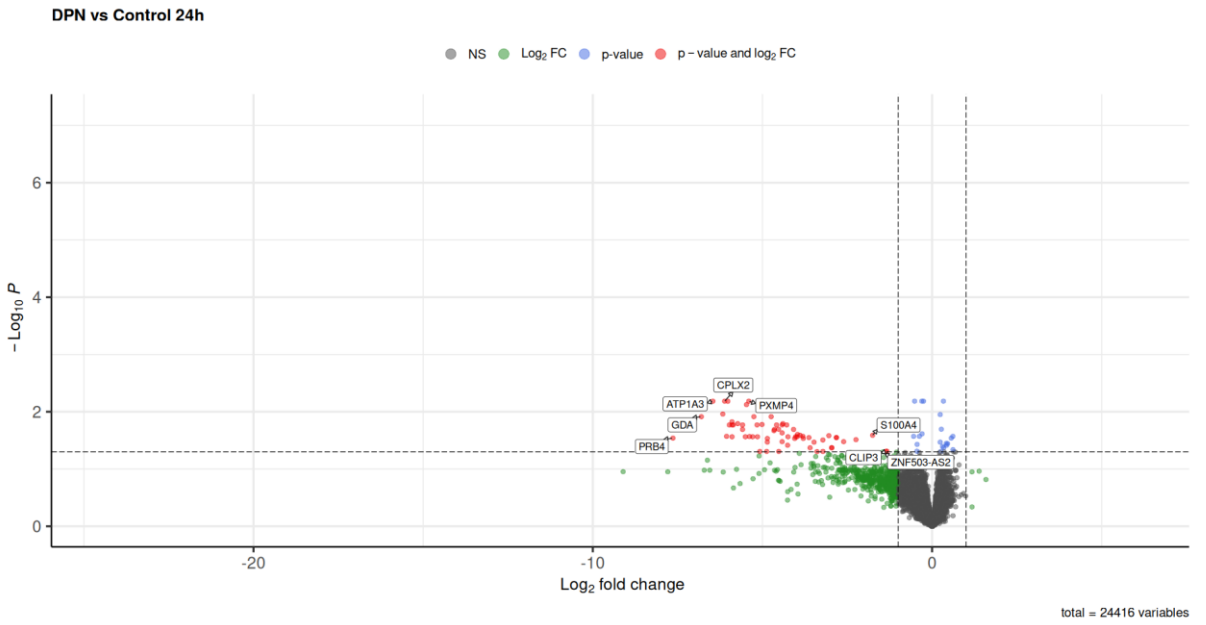
Los genes PRB4, GDA y ATP1A3 se encuentran menos expresados en las muestras con DPN frente al control a las 24h.

Guardamos la tabla para añadirla al informe:

```
> write.csv(deg_filtered_DPN_vs_Control_24h_clean,
+           "DEG_filtered_DPN_vs_Control_24h.csv", row.names = FALSE)
```

Creamos el volcano plot para visualizarlos:

```
> EnhancedVolcano(results_DPN_vs_Control_24h,
+                 lab = results_DPN_vs_Control_24h$row,
+                 x = "log2FoldChange",
+                 y = "padj",
+                 title = "DPN vs Control 24h",
+                 FCcutoff = 1,
+                 pCutoff = 0.05,
+                 subtitle = NULL,
+                 boxedLabels = TRUE,
+                 drawConnectors = TRUE,
+                 labSize = 4.0)
```



Por último, guardo el objeto dds2 para poder utilizarlo en la siguiente pregunta:

```
> saveRDS(dds2, file = "dds2.rds")
```

Pregunta 5 (2.5 puntos)

Nuestro colaborador ha comparado las muestras tratadas con DPN tras 48h con las muestras control. Nos ha llamado para contarnos que los cambios de expresión tras 48h son mucho más evidentes y nos preguntamos si el DPN produce algún efecto en las primeras 24h. Para contestar esta pregunta, le pedimos que genere un GMT (disponible en la carpera de input) con los genes más expresados en las muestras tratadas tras 48h (DPN_perturbed) y los genes más expresados en la muestra control (DPN_unperturbed). Realiza un análisis con GSEA para determinar el efecto del tratamiento a las 24h. ¿A qué conclusión llegas? Incluye una tabla con los resultados del análisis, destacando las columnas con los valores utilizados para extraer vuestras conclusiones. También incluye los gráficos característicos de este tipo de análisis.

Para poder responder a esta pregunta, necesitamos realizar un análisis de enriquecimiento de genes o GSEA (Gene Set Enrichment Analysis), el cual nos permite focalizar la atención a las funciones biológicas más relevantes.

Para ello, vamos a ejecutar GSEA en el modo preranked, es decir, proporcionando el ranking de genes de nuestro experimento. Por lo que necesitamos generar un archivo .rnk previamente.

En el script GSEA.R se detallan los pasos a seguir para generar este archivo y se puede encontrar en el siguiente enlace: https://github.com/dvalcarcel/transcriptomic-final-exercise/tree/main/Apartado2/Pregunta%205

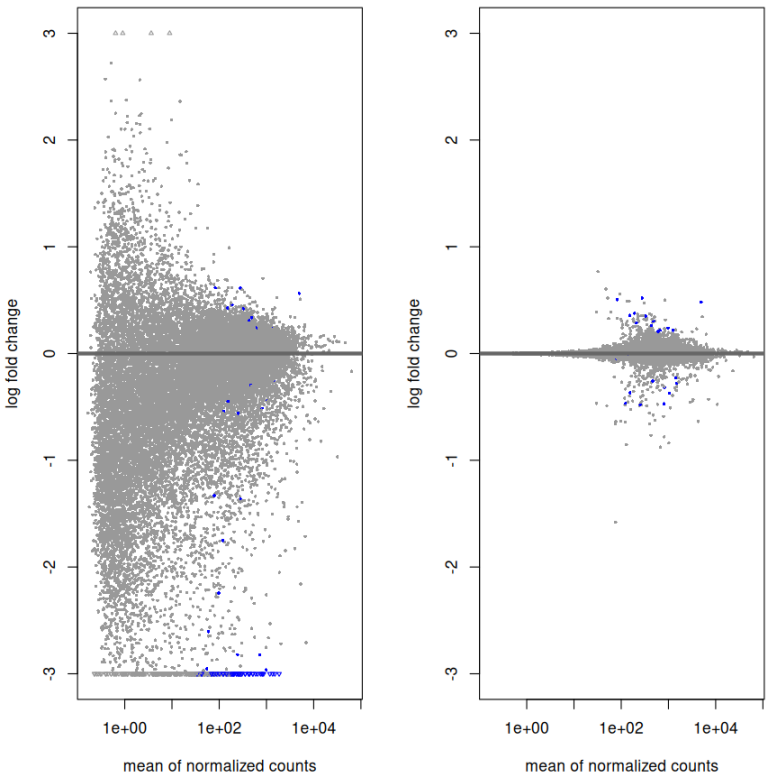
Para crear este archivo, primero realizamos la DEG de DPN vs Control a las 24h:

```
> res <- results(dds2, alpha = 0.05, contrast = c("group", "DPN_24h", "Control_24h"))
> summary(res)


out of 24416 with nonzero total read count
adjusted p-value < 0.05
LFC > 0 (up)       : 13, 0.053%
LFC < 0 (down)     : 74, 0.3%
outliers [1]       : 0, 0%
low counts [2]     : 10888, 45%
(mean count < 28)
[1] see 'cooksCutoff' argument of ?results
[2] see 'independentFiltering' argument of ?results
```

Después, tenemos que encoger los genes con cuentas bajas del objeto dds2, para evitar la imprecisión que pueden generar sus LFCs, utilizando la función lfcShrink de DESeq2:

```
> res.ape <- lfcShrink(dds2, coef = "group_DPN_24h_vs_Control_24h", type = "apeglm",
+                      res = res)
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
> summary(res.ape) # Mismo número de genes up/down padj < 0.05


out of 24416 with nonzero total read count
adjusted p-value < 0.05
LFC > 0 (up)       : 13, 0.053%
LFC < 0 (down)     : 74, 0.3%
outliers [1]       : 0, 0%
low counts [2]     : 10888, 45%
(mean count < 28)
[1] see 'cooksCutoff' argument of ?results
[2] see 'independentFiltering' argument of ?results
```

De manera visual, podemos observar lo que realiza la función:



Ahora creamos y guardamos el archivo .rnk:

```
> # Creamos el archivo .rnk
> rnk <- data.frame(Feature = rownames(res.ape), LFC = res.ape$log2FoldChange)
> # Guardamos el archivo .rnk (sin la cabecera y separado por tabuladores)
> write.table(rnk, file = "DPN_Control_24h.rnk", sep = "\t", quote = FALSE,
+             col.names = FALSE, row.names = FALSE)
```

Visualizamos el archivo generado mediante bash:

```
(base) vant@MOOVE15:~/Escritorio/transcriptomic-final-exercise/Apartado2$ head
DPN_Control_24h.rnk
```

```
ENSG00000000003       0.0622758726017299
ENSG00000000005       -0.00104405273522744
ENSG00000000419       0.00714608924081319
ENSG00000000457       -0.0796617309824683
ENSG00000000460       -0.0198578028133431
ENSG00000000938       -0.00824748739138022
ENSG00000000971       -0.00790883146953365
ENSG00000001036       -0.0308522421292216
ENSG00000001084       -0.00896863981524837
ENSG00000001167       0.00372976633493136
```

Descargamos el programa GSEA del siguiente enlace: https://www.gsea-msigdb.org/gsea/downloads.jsp

Descomprimimos el archivo:

```
(base) vant@MOOVE15:~/Escritorio/transcriptomic-final-exercise/Apartado2$ unzip
GSEA_LinuxIntel_4.4.0-WithJava.zip
```

Nos situamos en el directorio de GSEA y lo ejecutamos con java:

```
(base) vant@MOOVE15:~/Escritorio/transcriptomic-final-exercise/Apartado2$ cd
GSEA_Linux_4.4.0/
(base) vant@MOOVE15:~/Escritorio/transcriptomic-final-exercise/Apartado2/GSEA_Linux_4.4.0$
java --module-path=modules -Xmx4g @gsea.args --patch-module=jide.common=lib/jide-
components-3.7.4.jar:lib/jide-dock-3.7.4.jar:lib/jide-grids-3.7.4.jar --
module=org.gsea_msigdb.gsea/xapps.gsea.GSEA
```

Una vez ejecutada la aplicación, vamos a la sección “Tools / Run GSEAPreranked”.

En Gene sets database cargamos el archivo gmt realizado por nuestro colaborador (DPN_response.gmt), number of permutations 1000, en ranked list nuestro archivo .rnk de los genes prerankeados, seleccionamos no collapse para conservar el formato del archivo original y no seleccionamos ninguna plataforma de chip. Introducimos un nombre para nuestro análisis, enrichment statistic weighted, la ruta del output y una semilla para la permutacion (123).

Como alternativa, proporciono el código para ejecutarlo desde la terminal:

```
gsea-cli.sh GSEAPreranked -gmx /home/vant/Escritorio/transcriptomic-final-
exercise/Apartado2/input/DPN_response.gmt -collapse No_Collapse -mode Abs_max_of_probes -
norm meandiv -nperm 1000 -rnd_seed 123 -rnk /home/vant/Escritorio/transcriptomic-final-
exercise/Apartado2/DPN_Control_24h.rnk -scoring_scheme weighted -rpt_label DPN_Control_24h
-create_svgs false -help false -include_only_symbols true -make_sets true -plot_top_x 20 -
set_max 500 -set_min 15 -zip_report false -out /home/vant/Escritorio/transcriptomic-final-
exercise/Apartado2/GSEA_output
```

Los archivos generados por GSEA se pueden encontrar en el siguiente enlace: https://github.com/dvalcarcel/transcriptomic-final-exercise/tree/main/Apartado2/Pregunta%205/GSEA_output

En el archivo index.html podemos observar que GSEA ha fenotipado en na_pos y na_neg.

En el fenotipo na_pos encontramos 1 gene set upregulados significativo con un FDR (false discovery rate) < 25%, enriquecido significativamente con un p valor <1%. Este gen set es el DPN_perturbed, es decir, las muestras tratadas con DPN a las 48h.

	GS follow link to MSigDB	GS DETAILS	SIZE	ES	NES	NOM p-val	FDR q-val	FWER p-val	RANK AT MAX	LEADING EDGE
1	DPN_PERTURBED	Details ...	100	0.67	1.84	0.000	0.000	0.000	2726	tags=30%, list=11%, signal=34%

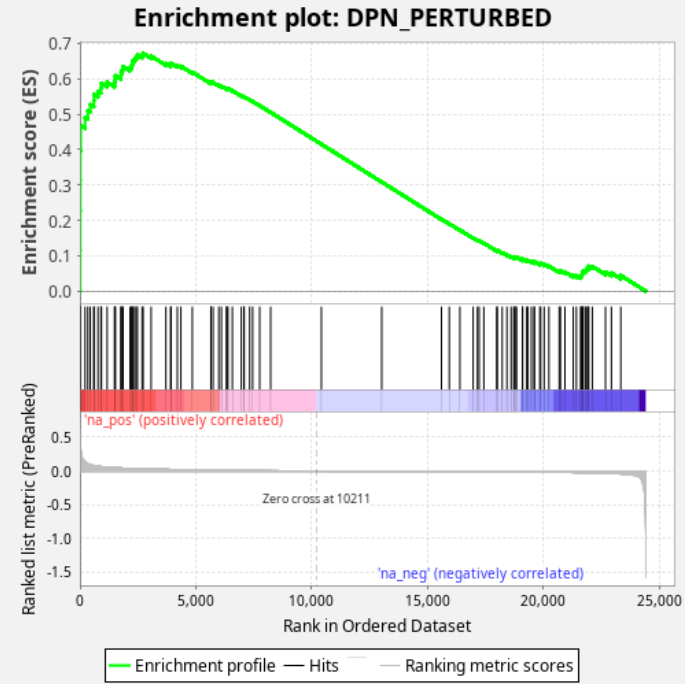
Este gen set tiene un Enrichment Score (ES) de 0.67, un Normalized Enrichment Score (NES) de 1.84 y un False Discovery Rate (FDR) de 0.

El ES refleja el grado en el que el conjunto de genes está sobrerrepresentado. Es la desviación máxima desde 0. Al tener un valor positivo, indica un enriquecimiento en la parte superior de la lista de genes.

El NES es el estadístico principal del análisis del enriquecimiento. Tiene en cuenta las diferencias en el tamaño del conjunto de genes, así como la correlación entre los conjuntos de genes y el conjunto de datos de expresión.

El FDR indica el porcentaje de falsos positivos. En este caso, al ser 0 es un valor excelente.

A continuación, muestro el gráfico del enriquecimiento de genes:



En el script GSEA.R, he filtrado y guardado un nuevo tsv con los genes enriquecidos en el leading edge que se puede encontrar en el siguiente enlace: https://github.com/dvalcarcel/transcriptomic-final-exercise/tree/main/Apartado2/Pregunta%205

A continuación, muestro la tabla con esos genes enriquecidos en el leading edge en la muestra tratada con DPN a las 24h y sus estadísticos:

```
> DPN_perturbed_refined
            SYMBOL RANK.IN.GENE.LIST RANK.METRIC.SCORE RUNNING.ES
1  ENSG00000138449                6         0.50126529  0.1156797
2  ENSG00000099194                9         0.48103213  0.2268446
3  ENSG00000171227               14         0.37330657  0.3130138
4  ENSG00000226549               16         0.35982981  0.3961896
5  ENSG00000197594               33         0.31382841  0.4681099
6  ENSG00000128590              223         0.13138177  0.4907216
7  ENSG00000157326              342         0.10492951  0.5101357
8  ENSG00000151726              443         0.09158358  0.5272034
9  ENSG00000166598              601         0.07699920  0.5385542
10 ENSG00000219481              608         0.07663590  0.5560308
11 ENSG00000211584              780         0.06807426  0.5647418
12 ENSG00000198830              908         0.06311164  0.5741146
13 ENSG00000099875              928         0.06232183  0.5877462
14 ENSG00000173210             1174         0.05486993  0.5903602
15 ENSG00000044574             1492         0.04844313  0.5885268
16 ENSG00000197976             1513         0.04790117  0.5987823
17 ENSG00000117643             1530         0.04745630  0.6090994
18 ENSG00000121064             1767         0.04312729  0.6093678
19 ENSG00000167608             1780         0.04294507  0.6188061
20 ENSG00000144712             1831         0.04197763  0.6264579
21 ENSG00000112294             1861         0.04133559  0.6348249
22 ENSG00000168209             2159         0.03751587  0.6312869
23 ENSG00000131620             2229         0.03671851  0.6369411
24 ENSG00000066248             2257         0.03638988  0.6442465
25 ENSG00000175356             2285         0.03598892  0.6514592
26 ENSG00000176155             2362         0.03523971  0.6564835
27 ENSG00000119242             2425         0.03458733  0.6619327
28 ENSG00000151552             2481         0.03387239  0.6675043
29 ENSG00000215644             2683         0.03165625  0.6665592
30 ENSG00000179218             2726         0.03102909  0.6720080
```

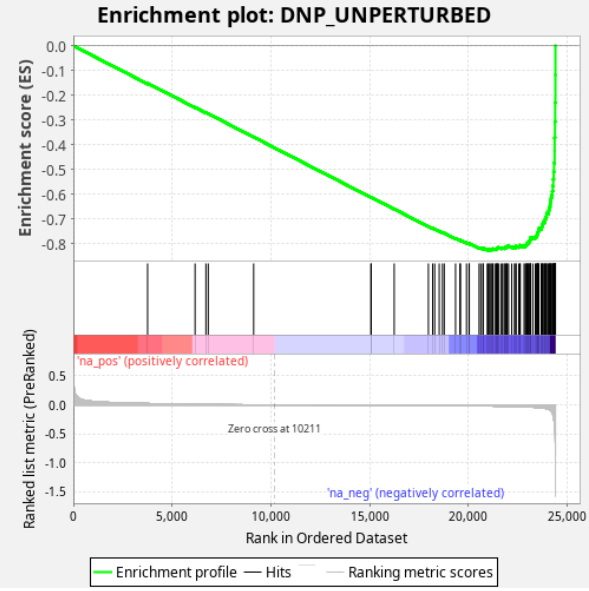
Por otro lado, en el fenotipo na_neg encontramos 1 gene set upregulados significativo con un FDR (false discovery rate) < 25%, enriquecido significativamente con un p valor <1%. Este gen set es el DPN_unperturbed, es decir, las muestras control a las 48h.

	GS follow link to MSigDB	GS DETAILS	SIZE	ES	NES	NOM p-val	FDR q-val	FWER p-val	RANK AT MAX	LEADING EDGE
1	DNP_UNPERTURBED	Details ...	100	-0.83	-2.34	0.000	0.000	0.000	3455	tags=76%, list=14%, signal=88%

Este gen set tiene un ES de -0.83, un NES de –2.34 y un FDR de 0.

El ES, al ser negativo, indica un enriquecimiento en la parte inferior de la lista de genes. El FDR al ser 0 es un valor excelente.

A continuación, muestro el gráfico del enriquecimiento de genes:



En el script GSEA.R, he vuelto a filtrar estos genes enriquecidos en el leading edge y los he guardado en un archivo tsv: https://github.com/dvalcarcel/transcriptomic-final-exercise/tree/main/Apartado2/Pregunta%205

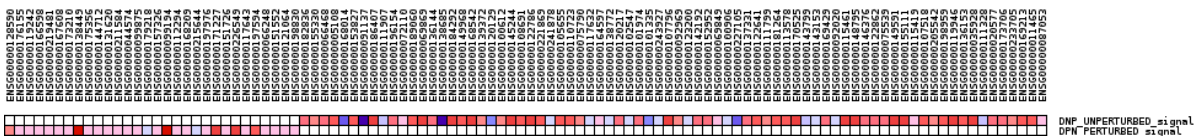
A continuación, muestro la tabla con esos genes enriquecidos en el leading edge en la muestra control a las 24h y sus estadísticos:

```
> DPN_unperturbed_refined
             SYMBOL RANK.IN.GENE.LIST RANK.METRIC.SCORE    RUNNING.ES
25  ENSG00000035928             20962       -0.01809384 -8.250804e-01
26  ENSG00000145244             21010       -0.01838709 -8.245361e-01
27  ENSG00000119946             21081       -0.01875812 -8.248876e-01
28  ENSG00000170525             21102       -0.01887481 -8.231671e-01
29  ENSG00000184292             21171       -0.01933420 -8.233588e-01
30  ENSG00000155111             21228       -0.01966646 -8.230122e-01
31  ENSG00000115461             21251       -0.01979254 -8.212504e-01
32  ENSG00000149591             21351       -0.02040349 -8.225729e-01
33  ENSG00000069869             21420       -0.02084754 -8.225607e-01
34  ENSG00000122786             21447       -0.02100868 -8.207995e-01
35  ENSG00000168542             21468       -0.02116193 -8.187709e-01
36  ENSG00000107796             21492       -0.02128843 -8.168486e-01
37  ENSG00000148795             21552       -0.02174633 -8.163452e-01
38  ENSG00000186407             21672       -0.02251405 -8.182058e-01
39  ENSG00000198959             21730       -0.02282947 -8.174742e-01
40  ENSG00000181264             21814       -0.02350490 -8.177209e-01
41  ENSG00000107518             21860       -0.02376519 -8.163696e-01
42  ENSG00000105655             21907       -0.02410115 -8.150144e-01
43  ENSG00000005108             21954       -0.02448070 -8.136079e-01
44  ENSG00000011465             21978       -0.02467542 -8.112293e-01
45  ENSG00000153827             22042       -0.02521600 -8.104229e-01
46  ENSG00000110723             22199       -0.02666136 -8.132464e-01
47  ENSG00000149968             22337       -0.02775736 -8.151409e-01
48  ENSG00000108691             22400       -0.02847482 -8.138543e-01
49  ENSG00000189060             22423       -0.02865108 -8.108990e-01
50  ENSG00000113578             22558       -0.02986128 -8.123866e-01
51  ENSG00000136153             22622       -0.03050531 -8.108677e-01
52  ENSG00000221869             22630       -0.03055910 -8.070384e-01
53  ENSG00000122641             22827       -0.03276800 -8.106841e-01
54  ENSG00000169213             22885       -0.03345689 -8.085207e-01
```

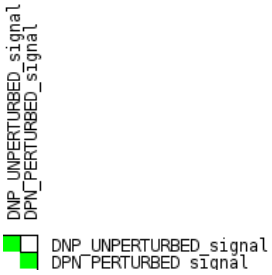
55	ENSG00000020577	22927	-0.03403240	-8.056218e-01
56	ENSG00000152952	22971	-0.03458605	-8.027305e-01
57	ENSG00000169429	22981	-0.03471722	-7.984233e-01
58	ENSG00000101974	23045	-0.03561277	-7.962161e-01
59	ENSG00000182836	23054	-0.03580333	-7.917215e-01
60	ENSG00000122862	23110	-0.03671569	-7.890367e-01
61	ENSG00000196154	23135	-0.03707016	-7.850293e-01
62	ENSG00000120217	23140	-0.03713271	-7.801911e-01
63	ENSG00000066468	23156	-0.03740054	-7.757691e-01
64	ENSG00000137331	23268	-0.03958272	-7.750011e-01
65	ENSG00000173706	23394	-0.04181596	-7.745080e-01
66	ENSG00000136144	23446	-0.04305656	-7.708045e-01
67	ENSG00000075539	23483	-0.04382975	-7.663800e-01
68	ENSG00000146376	23490	-0.04395529	-7.607047e-01
69	ENSG00000142192	23513	-0.04479564	-7.555743e-01
70	ENSG00000100612	23544	-0.04555713	-7.506702e-01
71	ENSG00000139372	23562	-0.04592514	-7.451820e-01
72	ENSG00000155330	23569	-0.04605953	-7.392233e-01
73	ENSG00000111799	23694	-0.05023896	-7.375542e-01
74	ENSG00000092969	23721	-0.05137977	-7.317012e-01
75	ENSG00000205542	23748	-0.05229566	-7.257248e-01
76	ENSG00000075790	23771	-0.05337819	-7.194381e-01
77	ENSG00000233705	23800	-0.05494311	-7.131873e-01
78	ENSG00000164597	23878	-0.05861865	-7.084564e-01
79	ENSG00000087053	23893	-0.05977710	-7.009785e-01
80	ENSG00000115419	23924	-0.06175864	-6.938917e-01
81	ENSG00000072110	23962	-0.06441514	-6.867349e-01
82	ENSG00000069849	23968	-0.06470569	-6.782228e-01
83	ENSG00000092020	24059	-0.07459474	-6.718742e-01
84	ENSG00000123200	24080	-0.07635807	-6.624092e-01
85	ENSG00000241878	24108	-0.08070898	-6.526458e-01
86	ENSG00000243927	24149	-0.08821402	-6.424060e-01
87	ENSG00000102547	24164	-0.09291720	-6.304633e-01
88	ENSG00000138772	24180	-0.09560058	-6.182002e-01
89	ENSG00000117632	24209	-0.10488807	-6.052204e-01
90	ENSG00000100906	24257	-0.14025797	-5.882567e-01
91	ENSG00000143153	24281	-0.16265447	-5.672886e-01
92	ENSG00000143799	24304	-0.20749252	-5.402384e-01
93	ENSG00000111907	24325	-0.23179023	-5.098324e-01
94	ENSG00000111328	24345	-0.27341270	-4.737777e-01
95	ENSG00000120129	24371	-0.37084371	-4.248430e-01
96	ENSG00000101335	24372	-0.39227867	-3.719924e-01
97	ENSG00000227105	24392	-0.48075008	-3.080036e-01
98	ENSG00000168014	24401	-0.57825202	-2.304263e-01
99	ENSG00000091137	24412	-0.83794510	-1.179435e-01
100	ENSG00000138685	24414	-0.87603664	4.143043e-05

Además, para comparar el enriquecimiento de genes en la muestra tratada con DPN y la muestra control a las 24h, he ejecutado el Leading Edge Analysis de GSEA.

En el siguiente heatmap se muestran los valores de expresión de enriquecimiento de los genes en ambos subset, donde el rojo indica alta expresión y el azul baja expresión. Se puede observar una clara diferencia entre genes tratados con DPN y la muestra control:

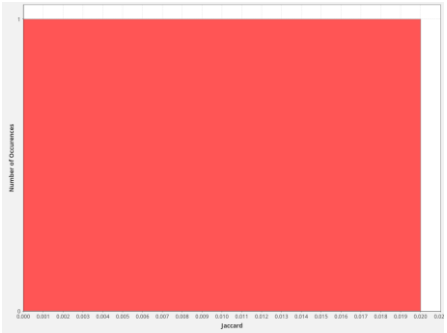


El siguiente gráfico muestra que no existe solapamiento entre ambos subset de genes:



Además, el siguiente gráfico muestra la cantidad de veces que aparece un gen en ambos subset de genes. En este caso, solo se ve un gen (amarillo) presente en ambos subsets.

El histograma muestra los coeficientes de similitud de Jaccard entre los distintos conjuntos de genes utilizados en el análisis GSEA. Solo hay una barra con un valor de 1 y con un Jaccard=0.02, lo que indica que los conjuntos de genes DPN vs Control tienen una similitud extremadamente baja. Esto también indica que no hay solapamiento de genes entre la muestra tratada y la muestra control.



Como conclusión, se puede observar que **el tratamiento con DPN también produce efectos a las 24h**. Concretamente, **produce un enriquecimiento de los siguientes 30 genes:**

```
> DPN_perturbed_refined$SYMBOL
 [1] "ENSG00000138449" "ENSG00000099194" "ENSG00000171227" "ENSG00000226549" "ENSG00000197594"
"ENSG00000128590" "ENSG00000157326" "ENSG00000151726"
 [9] "ENSG00000166598" "ENSG00000219481" "ENSG00000211584" "ENSG00000198830" "ENSG00000099875"
"ENSG00000173210" "ENSG00000044574" "ENSG00000197976"
[17] "ENSG00000117643" "ENSG00000121064" "ENSG00000167608" "ENSG00000144712" "ENSG00000112294"
"ENSG00000168209" "ENSG00000131620" "ENSG00000066248"
[25] "ENSG00000175356" "ENSG00000176155" "ENSG00000119242" "ENSG00000151552" "ENSG00000215644"
"ENSG00000179218"
```