Análisis de RNAseq en ratones con células B deficientes de T-bet durante la infección con *Plasmodium berghei*

Durante la infección con *Plasmodium falciparum* (agente causal de la malaria) se observa una respuesta retardada de anticuerpos específicos de alta afinidad contra el patógeno. Para que esta respuesta se lleve a cabo de forma óptima es esencial la colaboración de linfocitos T foliculares los cuales no expresan el factor de transcripción T-bet. Llamativamente, se ha observado la expresión de este factor en células B productoras de anticuerpos durante la infección que contribuyen a la inducción de la anemia autoinmune característica de la enfermedad. Por este motivo, se realizó un estudio de transcriptómica comparativa utilizando ratones $Tbx21^{fl/fl}Cd23^{Cre}$, los cuales poseen células B deficientes de T-bet. Se utilizaron como fuente de RNA linfocitos B con fenotipo de centro marginal obtenidos mediante *sorting* (usando un citómetro de flujo) a partir de esplenocitos totales.

Antes que nada cargo las librerías que voy a usar:

```
library(tidyverse)
library(DESeq2)
library(skimr)
library(ggplot2)
library(pheatmap)
```

Inicialmente, descargué el archivo de texto "GSE120728_Supp-Count-Table.txt" con los datos del análisis de RNAseq de cuentas por gen, para cada una de las condiciones y sus respectivas réplicas. Dado que los datos están separados entre sí por "TAB" utilicé la función *read.tsv*. Está claro que para poder realizar esta función establecí el *working directory* como la carpeta donde se encontraba el archivo. Cargo los datos:
Countdata<-read_tsv("GSE120728_Supp-Count-Table.txt")

Para tener una idea de cómo son los datos utilizo la función skim, la cual nos da una serie de características y estadísiticos de cada una de las variables, además de mostrarnos un histograma de cada una de las muestras lo cual nos puede dar una idea de cómo están distribuidos los datos.

```
skim(Countdata)
── Data Summary ────────────────────────
                           Values
Name                       Countdata
Number of rows             27179
Number of columns          8
_____________________
Column type frequency:
character                  1
numeric                    7
_____________________
Group variables            None

── Variable type: character ────────────────────────────────────────────────────
skim_variablen_missingcomplete_rate   min   max empty n_unique whitespace
1 Symbol              874         0.968     1    16     0    26305          0

── Variable type: numeric ──────────────────────────────────────────────────────
skim_variablen_missingcomplete_rate      mean        sd    p0   p25    p50     p75
1 EntrezID                          0            1 13052150. 33540824. 11287 57769 104718 353234.
2 D10-WT-GC-Bcell_Rep1              0            1     1087.     5221.     0     0      8    709
3 D10-WT-GC-Bcell_Rep2              0            1     1111.     6203.     0     0      7    694.
4 D10-WT-GC-Bcell_Rep3              0            1     1040.     5880.     0     0      5    624
5 D10-TbetKO-GC-Bcell_Rep1          0            1      741.     6227.     0     0      0    416.
6 D10-TbetKO-GC-Bcell_Rep2          0            1      981.     6252.     0     0      8    670
7 D10-TbetKO-GC-Bcell_Rep3          0            1      392.     1997.     0     0      3    275
       p100 hist
1 100862406 ▇▁▁▁
2    512470 ▇▁▁▁
3    671938 ▇▁▁▁
4    486132 ▇▁▁▁
5    747398 ▇▁▁▁
6    768584 ▇▁▁▁
7    219107 ▇▁▁▁
```

Los datos importados se tratan de una tabla con el el *EntrezID* (identificador numerico del gen), el *Symbol* (nombre abreviado del gen) y las cuentas para 3 réplicas de RNA proveniente de ratones WT o "KO" al día 10 de infección con *Plasmodium berghei*. A primer vista podemos ver que existen 874 entradas sin nombre de gen (NA), lo cual es algo que puede llegar a traer complicaciones más tarde durante el análisis. Luego podemos apreciar que los ratones WT parecen tener media mas alta de cuentas que su contraparte KO. Por otro lado, mirando los estadísticos en conjunto da la impresión que las réplicas WT serían más homogéneas que las KO. En cuanto a la distribución que todas presentan valores próximos al eje Y, dando una distribución asimétrica y por ende, no de carácter normal.

```
>sample(Countdata)
# Atibble: 27,179 x 8
   `D10-WT-GC-Bcel… `D10-WT-GC-Bcel… Symbol `D10-TbetKO-GC-… `D10-TbetKO-GC-… `D10-WT-GC-Bcel…
<db1><db1><chr><db1><db1><db1>
 1                0                0 Xkr4                   0                0                0
 2                0                0 Gm199…                 0                0                0
 3                0                0 Gm105…                 0                0                0
 4                0                0 Rp1                    0                0                0
 5                0                0 Sox17                  0                0                0
 6             1947             2701 Mrpl15              2725              775             1174
 7             5287             5733 Lypla1              5199             1801             3245
 8               15                5 Gm198…                 9                2               11
 9             5803             6284 Tcea1               6602             2181             1995
10                0                0 Rgs20                  6                1                0
# … with 27,169 more rows, and 2 more variables: EntrezID<db1>, `D10-TbetKO-GC-Bcell_Rep1` <db1>
```

Utilizando la función simple (default n=10) ya podemos ver que la mitad de los datos que figuran poseen lecturas nulas en por lo menos una de sus variables.

Antes de continuar, formateo los datos:

Elimino la columna "EntrezID" que me da información innecesaria y/o redundante con "Symbol"

```
Countdata<-select(Countdata,-EntrezID)
```

Elimino las entradas sin identificadores de genes:

```
Countdata<-filter(Countdata,!is.na(Symbol))
```

Pongo de nombres de las entradas a la columna Symbol y la elimino como columna, son requisitos para poder correr la función DESeq2 posteriormente.

```
row.names(Countdata)<-Countdata$Symbol

Countdata<-select(Countdata, -Symbol)
```

Renombro las columnas para que sean más legibles y entendibles:

```
colnames(Countdata)<-c("WT1", "WT2", "WT3", "KO1", "KO2", "KO3")
```

Volviendo al análisis, verifico cuántas entradas poseen al menos un valor nulo de cuentas:

```
filter(Countdata, WT1==0, WT2==0, WT3==0, KO1==0, KO2==0, KO3==0)
```

```
# A tibble: 8,652 x 6
     WT1   WT2   WT3   KO1   KO2   KO3
   <db1> <db1> <db1> <db1> <db1> <db1>
 1     0     0     0     0     0     0
 2     0     0     0     0     0     0
 3     0     0     0     0     0     0
 4     0     0     0     0     0     0
 5     0     0     0     0     0     0
 6     0     0     0     0     0     0
```

```
 7     0     0     0     0     0     0
 8     0     0     0     0     0     0
 9     0     0     0     0     0     0
10     0     0     0     0     0     0
# ... with 8,642 more rows
```

Lo que da un resultado de 8652 entradas con al menos un valor nulo en sus variables. Esto representa casi 1/3 de las entradas totales, lo cual puede explicar las distribuciones observadas en el “skim”. Esto tiene sentido dado que se parte células B de un fenotipo particular y se compara con el genoma entero de ratón, de forma que debido a la alta diferenciación de estas células gran parte del transcriptoma no se encuentra representado.

Quizás sea necesario posteriormente realizar algún tipo de transformación de los datos para asemejar la población a la normalidad. En estos casos, la transformación logarítmica puede resultar útil. De todas formas, debido a que la función DESeq2 del paquete Bioconductor utiliza directamente las cuentas no será necesario realizar la transformación.

Creo los vectores necesarios para correr la funcion DESeq2:

Este primer vector indica cuáles son las condiciones (utiliza la clase de objeto “factor”) y la cantidad de réplicas que tienen:

```
condition<-factor(c(rep("WT", 3), rep("KO", 3)))
```

En el siguiente, coloco los nombres de cada una de las muestras como nombres de columnas al vector creado anteriormente y lo convierto en un “data frame”:

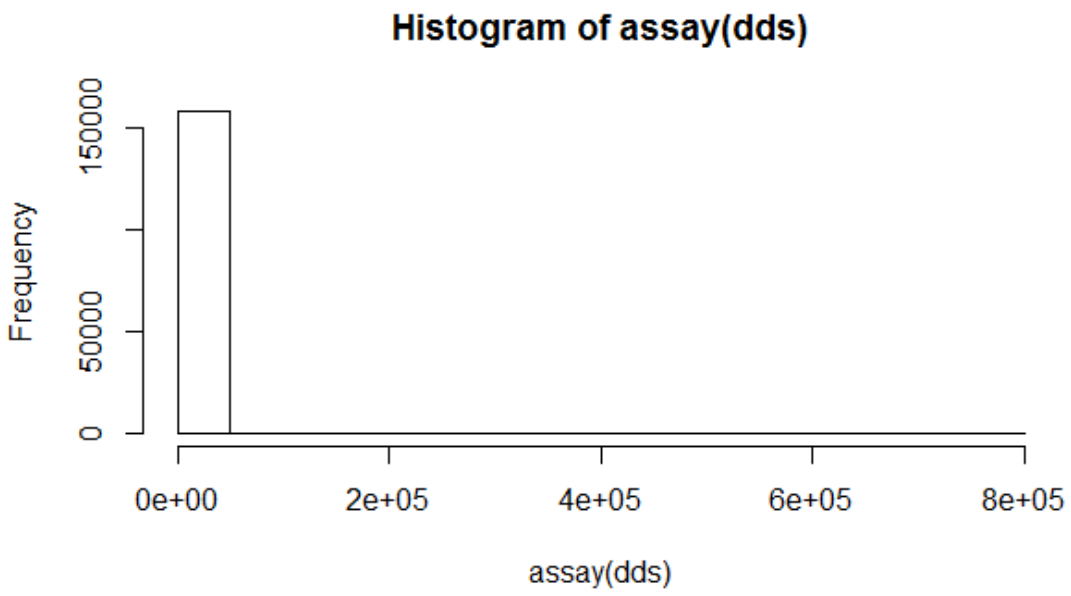
```
coldata<-data.frame(row.names=c("WT1", "WT2", "WT3", "KO1", "KO2", "KO3"), condition)
```

Creo el dataset que usa DESeq2 a partir de la tabla formateada y los objetos creados:

```
dds<-DESeqDataSetFromMatrix(countData=Countdata, colData=coldata, design=~condition)
```

Veo cómo se distribuyen los datos y muestra lo que vimos inicialmente en el “skim”:

```
hist(assay(dds))
```

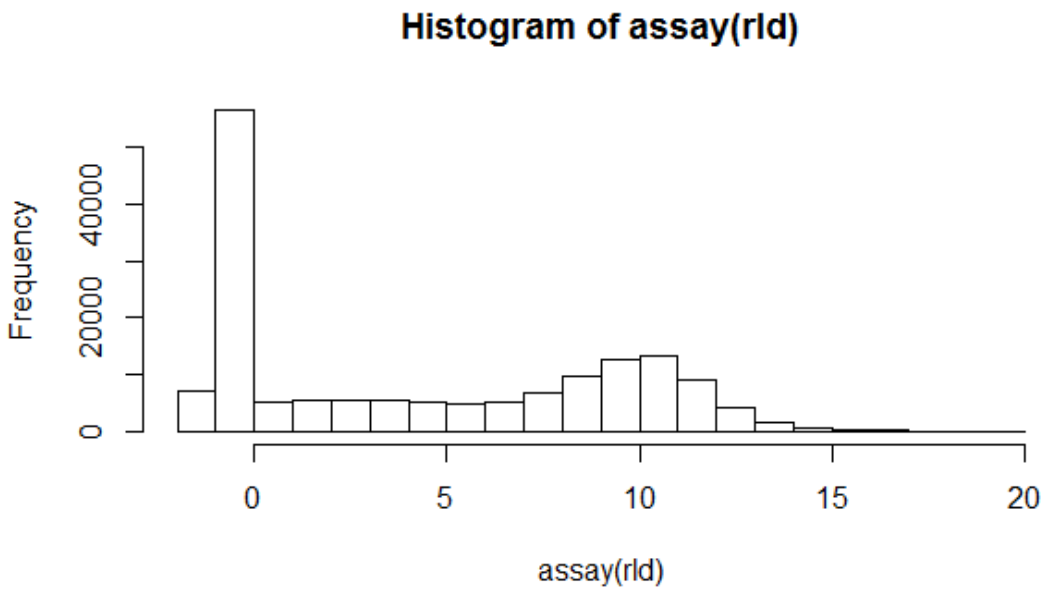


Ahora si, transformo los datos utilizando un logaritmo en base 2:

```
rld<-rlogTransformation(dds)
```

Verifico cómo se distribuyen ahora:

hist(assay(rld))



Vemos que ahora tienen una forma semejante a la normalidad, por lo menos aquellos datos que poseían un valor considerable de cuentas.

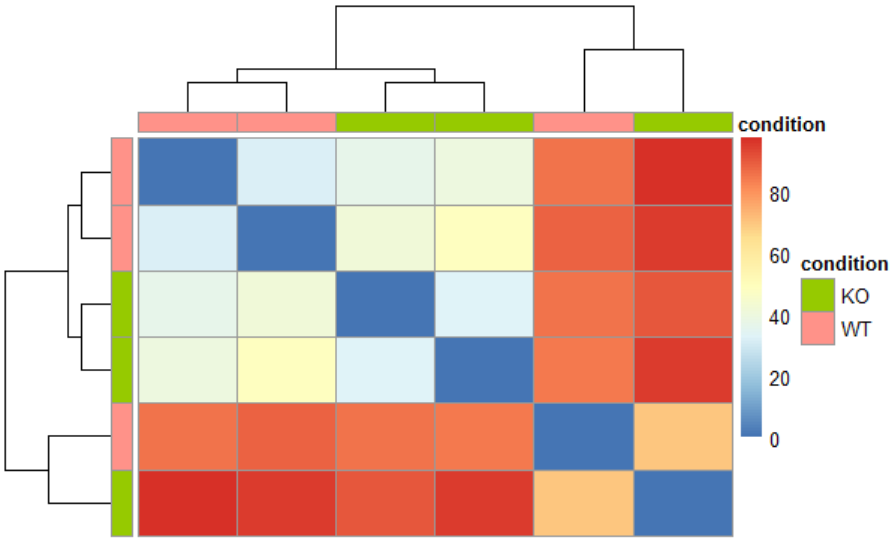
Para graficar los datos y ver cuánto se parecen las muestras entre sí podemos usar un heatmap. Para eso creo la matriz distancia necesaria para realizarlo mediante la función “pheatmap”:

sampleDists<-as.matrix(dist(t(assay(rld))))

nombres<-as.data.frame(condition,colnames(Countdata))

Realizo el heatmap:

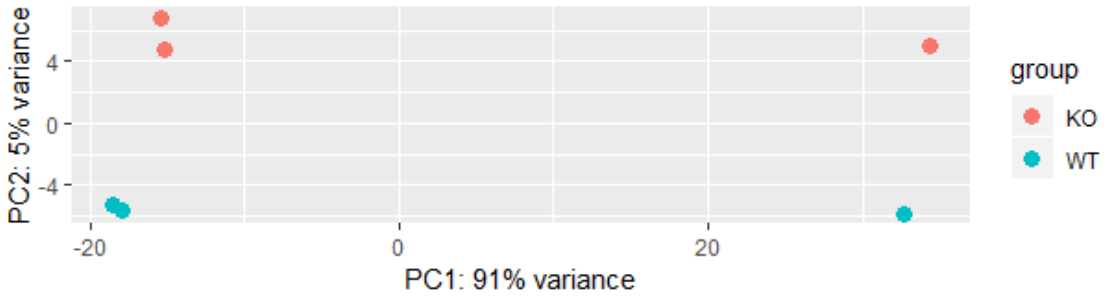
pheatmap(sampleDists, annotation_row = nombres, annotation_col = nombres, show_rownames = F , show_colnames = F, annotation_names_row = F)



Esta función utiliza PCA para graficar las muestras, es decir que compone variables nuevas a partir de las existentes que puedan explicar las diferencias entre muestras. Nótese que una par de réplicas de backgrounds genéticos diferentes se asemejan más entre sí que las demás, cuando uno esperaría que todas las muestras WT se agrupen por lado y las KO por otro. Esto podría tratarse de un error técnico.

Ahora analicemos cómo son los valores de los componentes principales de variabilidad 1 y 2 :

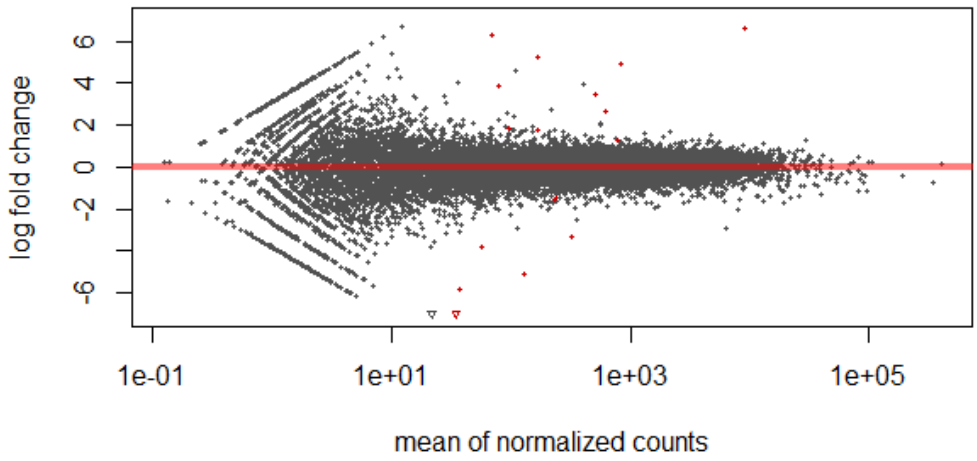
DESeq2::plotPCA(rld)



Acá vemos claramente como esas dos muestras difieren del resto, de forma concordante con el *heatmap*. Nótese que la diferencia relativa entre los PC1 y PC2 de las muestras divergentes es bastante similar al de las restantes. Esto apoyaría la idea que se trata de un problema técnico.

Luego podemos graficar un *MAplot* para ver cómo cambia la diferencia de expresión en función del número de cuentas:

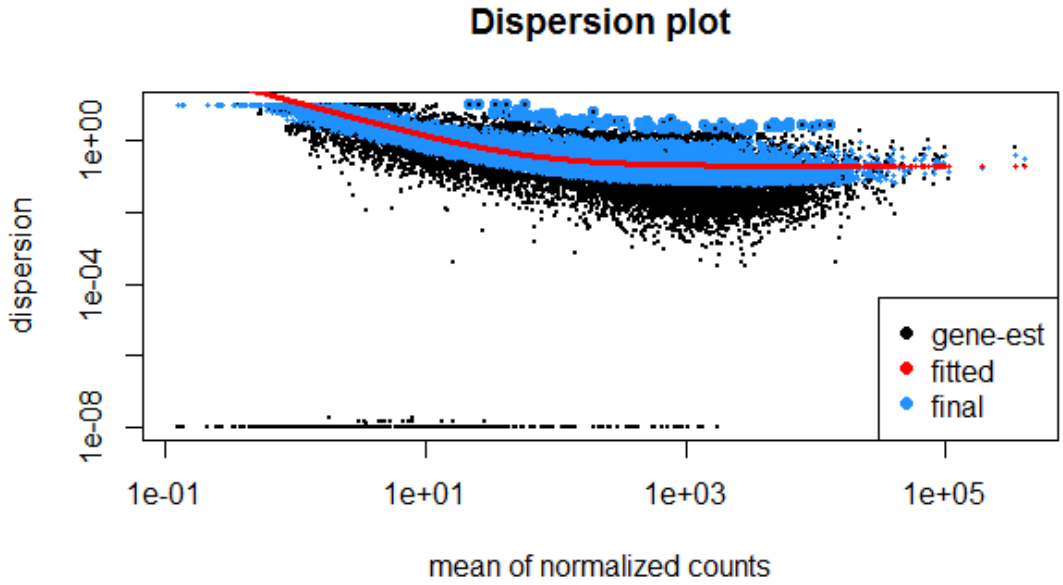
DESeq2::plotMA(dds, ylim=c(-7,7))



Se puede observar cómo a menor cantidad de cuentas la magnitud del cambio entre condiciones para un gen se hace más grande. Esto tiene sentido debido a que a menor número de cuentas la dispersión resulta mucho mayor. Aquellos puntos que posean una gran cantidad de cuentas y alta diferencia entre condiciones (puntos rojos), serán aquellos que seguramente tendrán diferencias estadísticamente significativas entre condiciones.

Para demostrar la afirmación anterior acerca de la dispersión puede realizarse el siguiente gráfico:

plot<-plotDispEsts(dds, main="Dispersion plot")



Y efectivamente se ve cómo la dispersión decrece a medida que aumentan las *reads*.

Realizamos test de Wald entre condiciones, para observar si existen diferencias estadísticamente significativas:

resT1<-results(dds, cooksCutoff=FALSE, contrast=c("condition","WT","KO"))

Esto genera un objeto de tipo “lista”, que no se puede visualizar. Procedemos entonces a confeccionar un *data frame*, volviendo a poner los “Symbol” como nombres de las entradas:

```
resT1<-cbind(Symbol=rownames(as.data.frame(resT1)),as.data.frame(resT1))
```

Hacemos un *skim* de este *data frame:*

```
> skim(resT1)
-- Data Summary ------------------------
                           Values
Name                       resT1
Number of rows             26305
Number of columns          8
_________________________
Column type frequency:
  factor                   1
  numeric                  7
_________________________
Group variables            None

-- Variable type: factor -------------------------------------------------------
  skim_variable n_missing complete_rate ordered n_unique
1 Symbol                0             1 FALSE      26305
  top_counts
1 061: 1, 061: 1, 061: 1, 061: 1

-- Variable type: numeric ------------------------------------------------------
  skim_variable  n_missing complete_rate          mean            sd        p0
1 baseMean               0             1          859.          4656.      0.
2 log2FoldChange      8652             0.671       -0.0411        1.17   -2.26e+ 1
3 lfcSE               8652             0.671        1.27          1.23    2.89e- 1
4 stat                8652             0.671       -0.0297        0.907  -9.29e+ 0
5 pvalue              8652             0.671        0.545         0.278   4.46e-53
6 padj               12987             0.506        0.996         0.0504  5.95e-49
7 EntrezID               0             1     10361249.     30273868.      1.13e+ 4
        p25         p50        p75          p100 hist
1     0         8.31       619.         403209.  ▇▁▁▁
2    -0.464     0.00452      0.406         6.71  ▁▁▁▇▁
3     0.433     0.636        1.59          4.08  ▇▁▁▂
4    -0.608     0.00694      0.547        15.3   ▁▇▁▁
5     0.316     0.565        0.782         1.000 ▅▃▃▃▇
6     0.999     0.999        0.999         1.000 ▁▁▁▁▇
7 56535     97541       320411     100862406     ▇▁▁▁
```

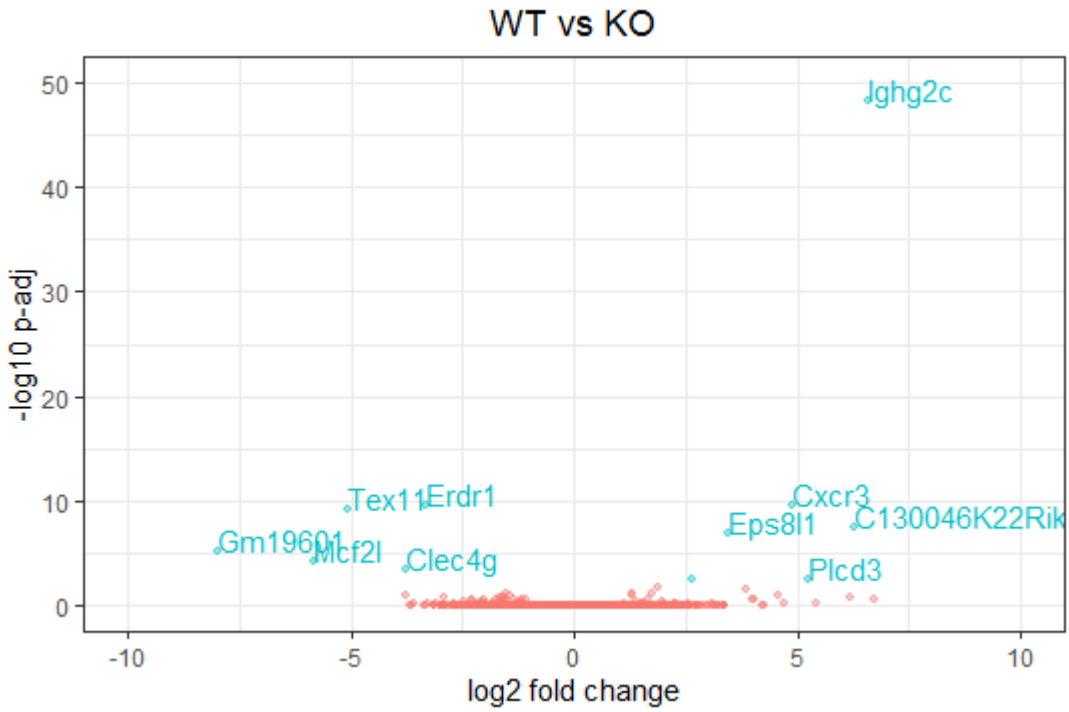
Vemos que presenta dos valores de p-valor, un p-valor a secas y otro “ajustado” que resulta bastante más exigente. Eso se debe a que en el primer caso se calcula utilizando las cuentas y en el segundo se usa la transofrmación con logaritmo en base 2 (“FDR”).

Para crear un Vulcano plot con esta información creo una tabla de referencias y las uno con la función *merge* utilizando “Symbol”:

```
Referencias<-select(Countdata, EntrezID, Symbol)%>%
 filter(.,!is.na(Symbol))
resT1<-merge(resT1,Referencias,by.x="Symbol",by.y="Symbol")
```

Grafico el Vulcano plot utilizando como punto de corte de significancia 0,01. Este grafico vincula los p-valores ajustados con el logaritmo en base 2 del cociente entre condiciones:

```
data <- resT1

data$threshold = as.factor(data$padj < 0.01)

g <- ggplot(data=data,

        aes(x=log2FoldChange, y =-log10(padj),   colour=threshold)) +

 geom_point(alpha=0.4, size=1) +   xlim(c(-10, 10)) + ylim(c(0,50))+

 xlab("log2 fold change") + ylab("-log10 p-adj") + ggtitle("WT vs KO")+

 theme_bw() + theme(legend.position="none",plot.title = element_text(hjust = 0.5))+

 geom_text(aes(label=ifelse(-log10(pvalue)>5 & (log2FoldChange>3 | log2FoldChange< -3)  ,as.character(Symbol),'')),hjust=0,vjust=0)
```



Aquellos genes con fold change > 0 se encuentran *upregulados* en los ratones WT con respecto a los KO, mientras que si el fold change < 0 se hallan *downregulados.*

Luego genero una tabla con los 10 genes de mayor significancia estadística:

```
Symbols<-c("Ighg2c", "Tex11", "Erdr1","Gm19601", "Mcf2l", "Clec4g", "Pcld3", "Cxcr3",

        "Eps8l1", "C130046K22Rik")

Descripcion<-c("Immunoglobulin heavy constant gamma 2C", "Testis expressed sequence 11",

            "Erythroid differentiation regulator 1", "Predicted gene",

            "Guanine Nucleotide Exchange Factor DBS", "C-Type Lectin Domain Family 4 Member G",

            "Probable dolichyl pyrophosphate Glc1Man9GlcNAc2 alpha-1,3-glucosyltransferase",

            "Chemokine receptor CXCR3","Epidermal growth factor receptor kinase substrate 8-like protein 1",

            "RIKEN cDNA C130046K22 gene")
```

Resultados<-cbind(Symbols, Descripcion)

	Symbols	Descripcion
1	Ighg2c	Immunoglobulin heavy constant gamma 2C
2	Tex11	Testis expressed sequence 11
3	Erdr1	Erythroid differentiation regulator 1
4	Gm19601	Predicted gene
5	Mcf2l	Guanine Nucleotide Exchange Factor DBS
6	Clec4g	C-Type Lectin Domain Family 4 Member G
7	Pcld3	Probable dolichyl pyrophosphate Glc1Man9GlcNAc2 alp...
8	Cxcr3	Chemokine receptor CXCR3
9	Eps8l1	Epidermal growth factor receptor kinase substrate 8-like...
10	C130046K22Rik	RIKEN cDNA C130046K22 gene

En particular es importante notar la *downregulación* de transcriptos en ratones KO con respecto a WT de cadena pesada de la inmunoglobulina 2C junto con CXCR3. Ambas moléculas son indicadores de la generación de anticuerpos de alta afinidad, de manera que podemos decir que T-bet estaría teniendo un rol en la generación de los mismos en el contexto de la infección con *Plasmodium berghei.*