Análisis Datos Series GSE35240

Alba Moya Garcés

4 de mayo, 2020

Contents

Abstract

Basándonos en los datos de microarrays proporcionados por Baumbach(2012) realizaremos un análisis de los mismos utilizando diferentes paquetes R y Bioconductor de los datos sin procesar (Archivos binarios obtenidos del proceso de hibridación). La línea de análisis contiene diferentes pasos hasta lelgar a los resultados finales y su discusión. Se describirá en la medida de lo posible cada proceso y se facilitará el código necesario para su reproducibilidad a lo largo del texto de manera parcial y en su totalidad en un repositorio de github.

Objetivos

Se pretende replicar los análisis desarrollados por los autores, tomando los datos en crudos facilitados en la base de datos *Gene Expression Omnibus* (GEO). El análisis resultante debería servir para evaluar cómo la pérdida o amplificación de los centrosomas puede afectar la fisiología celular al perfilarse el transcriptoma global cerebral y de los discos imaginales de las larvas de *Drosophila* y llegar a las mismas conclusiones que Baumbach(2012).

Material y métodos

Estudio elegido:

Se ha seleccionado el estudio **"Gene expression in mitotic tissues of Drosophila larvae without centrosomes or too many centrosomes"**, analizan el efecto de la sobreexpresión o pérdida de centrosomas en la formación de células tumorales en *Drosophila melanogaster*. Para ello, se tomaron dos líneas mutantes con pérdida de centrosomas (DSas-4 y DSas-6) y una línea productora de sobreexpresión de centrosomas (SakOE) y se compararó la expresión genética de las mismas con dos líneas de tipo salvaje de control (w67 y OregonR).

Baumbach(2012) explica que se diseccionaron tejidos mitóticos (cerebrales y de los discos imaginales) de 10 larvas de *Drosophila* en estadío 3 de las diferentes líneas a estudio y se extrajo el ARN de tres réplicas en cada una de ellas (15 muestras). Se hibridaron con arrays *Affymetrix Drosophila Genome 2.0* y se obtuvo la serie GSE35240 con todos los datos almacenados en archivos .CEL

El artículo resultante se puede encontrar íntegro en la página web de Biology Open en el siguiente enlace: https://bio.biologists.org/content/1/10/983.short

Preparación del área de trabajo:

Para llevar a cabo un análisis de microarrays, el analista debe gestionar una gran cantidad de archivos entre aquesllos que ocupan los datos originales y los generados durante su análisis. Es por ello que siempre se debería comenzar creando als carpetas necesarias para simplificar la ruta de trabajo. Se recomienda generar una **carpeta principal** con el nombre de nuestro proyecto en cuyo interior alojaremos una carpeta con los archivos de **datos** y otra con los **resultados** generados del análisis

Estas carpetas las genereamos rápidamente desde el explorador de archivos o la consola de cualquiera de los sistemas operativos usuales. Desde R también podemos generar estas subcarpetas mediante el siguiente código:

```
setwd(".")
dir.create("data")
dir.create("results")
```

El código completo para desarrollar este análisis, o cualquier otro a partir de su adaptación, puede descargarse del siguiente repositorio de *GitHub*:

https://github.com/albamgarces/reanalisis_microarrays.git.

Instalación de paquetes en R

Se necesitarán paquetes adicionales a los incluidos en la intalación básica de R para poder llevar a cabo el análisis. Estos paquetes pueden descargarse tanto del repositorio CRAN para los paquetes típicos de R o directamente de Bioconductor para las funciones del mismo.

A continuación se muestran los paquetes necesarios fpara este estudio que requieren instalación:

```
# UNCOMMENT IF INSTALL REQUIRES
# install.packages("knitr")
#install.packages("cluster")
# install.packages("gplots")
# install.packages("ggplot2")
# install.packages("ggrepel")
# install.packages("BiocManager")
# BiocManager::install("oligo")
# BiocManager::install("arrayQualityMetrics")
# BiocManager::install("pvca")
# BiocManager::install("pacman")
# BiocManager::install("geneplotter")
# BiocManager::install("org.Dm.eg.db")
# BiocManager::install("limma")
# BiocManager::install("genefilter")
# BiocManager::install("drosophila2.db")
# BiocManager::install("ReactomePA")
```

Descarga de los datos

En lugar de descargar los archivos .CEL y construir manualmente el archivo "targets", se utilizará el paquete `geoQuery` para generar de forma automática el objeto ExpressionSet necesario para el análisis. De esta forma se evitan posibles errores de transcripción o codificador por parte del analista. Además, como los datos ya están normalizados, se procederá a desarrollar el análisis de calidad directamente sobre los datos normalizados.

```
if (!require(GEOquery)) {
  BiocManager::install("GEOquery")
}
require(GEOquery)
gse <- getGEO("GSE35240")
rawData <- gse[[1]]
rawData
```

```
FALSE ExpressionSet (storageMode: lockedEnvironment)
```

```
FALSE assayData: 18952 features, 15 samples
FALSE   element names: exprs
FALSE protocolData: none
FALSE phenoData
FALSE   sampleNames: GSM864362 GSM864363 ... GSM864376 (15 total)
FALSE   varLabels: title geo_accession ... tissue:ch1 (41 total)
FALSE   varMetadata: labelDescription
FALSE featureData
FALSE   featureNames: 1616608_a_at 1622892_s_at ... AFFX-TrpnX-M_at (18952
FALSE     total)
FALSE   fvarLabels: ID CLONE_ID_LIST ... Gene Ontology Molecular Function (16
FALSE     total)
FALSE   fvarMetadata: Column Description labelDescription
FALSE experimentData: use 'experimentData(object)'
FALSE   pubMedIds: 23213376
FALSE Annotation: GPL1322
```

El objeto `expressionSet`combina las difetrentes fuentes de información del estudio en una única estructura. Además de incluir toda la información generada durante el desarrollo del experimento, podemos cambiar la visualización de los datos para que sea más sencillo su uso.

Control de calidad de los datos

Se debe primero analizar si los datos tiene suficiente calidad para poder trabajar con ellos. Unos datos de mala calidad prodían producir demasiado ruido en el análisis que no será resuelto al realizar el proceso de normalización.

El primer paso para llevar esto a cabo será descargar el paquete `ArrayQualityMetrics` que nos permite desarrollar un estudio de calidad de los datos. Si resulta algún array fuera de los límites de calidad propuestos, aparecerá marcado con un asterisco y podrá ser detectado inmediatamente. Si el mismo array destaca tres veces, debería ser analizado y considerar su eliminación del estudio para mejorar cualitativamente el experimento.

```
arrayQualityMetrics(rawData, outdir="./results/rawdata_quality", force=TRUE)
```

Se realizará un análisis conjunto de la calidad de los datos y se creará una nueva carpeta con un informe llamado *index.html* en el que podremos acceder al resumen de los análisis desarrollados y que nos indicará las muestras de calidad dudosa mediante una marca en cada análisis y muestra en las que la calidad sea deficiente. En el objeto de nuestro estudio y como podemos ver no tenemos ninguna muestra cuya calidad deba preocu-

- Array metadata and outlier detection overview

	array	sampleNames	*1	*2	*3	title	characteristics_ch1	description	genotype:ch1
☐	1	D-Sas4_1				brains and imaginal discs from D-Sas4 mutant 3rd instar Drosophila larvae, biological replicate 1	genotype: DSas-4 mutant	Gene expression data from mitotic Drosophila cells that do not have centrioles	DSas-4 mutant
☐	2	D-Sas4_2				brains and imaginal discs from D-Sas4 mutant 3rd instar Drosophila larvae, biological replicate 2	genotype: DSas-4 mutant	Gene expression data from mitotic Drosophila cells that do not have centrioles	DSas-4 mutant
☐	3	D-Sas4_3				brains and imaginal discs from D-Sas4 mutant 3rd instar Drosophila larvae, biological replicate 3	genotype: DSas-4 mutant	Gene expression data from mitotic Drosophila cells that do not have centrioles	DSas-4 mutant
☐	4	D-Ssas6_1				brains and imaginal discs from D-Ssas6 mutant 3rd instar Drosophila larvae, biological replicate 1	genotype: DSas-6 mutant	Gene expression data from mitotic Drosophila cells that do not have centrioles	DSas-6 mutant
☐	5	D-Ssas6_2				brains and imaginal discs from D-Ssas6 mutant 3rd instar Drosophila larvae, biological replicate 2	genotype: DSas-6 mutant	Gene expression data from mitotic Drosophila cells that do not have centrioles	DSas-6 mutant
☐	6	D-Ssas6_3				brains and imaginal discs from D-Ssas6 mutant 3rd instar Drosophila larvae, biological replicate 3	genotype: DSas-6 mutant	Gene expression data from mitotic Drosophila cells that do not have centrioles	DSas-6 mutant
☐	7	SakOE_1				brains and imaginal discs from Sak overexpressing 3rd instar Drosophila larvae, biological replicate 1	genotype: Sak overexpression	Gene expression data from mitotic Drosophila cells that have too many centrosomes	Sak overexpression
☐	8	SakOE_2				brains and imaginal discs from Sak overexpressing 3rd instar Drosophila larvae, biological replicate 2	genotype: Sak overexpression	Gene expression data from mitotic Drosophila cells that have too many centrosomes	Sak overexpression
☐	9	SakOE_3				brains and imaginal discs from Sak overexpressing 3rd instar Drosophila larvae, biological replicate 3	genotype: Sak overexpression	Gene expression data from mitotic Drosophila cells that have too many centrosomes	Sak overexpression
☐	10	w67WT_1				brains and imaginal discs from w67 wild type 3rd instar Drosophila larvae, biological replicate 1	genotype: white wild type	Gene expression data from mitotic Drosophila wild type cells	white wild type
☐	11	w67WT_2				brains and imaginal discs from w67 wild type 3rd instar Drosophila larvae, biological replicate 2	genotype: white wild type	Gene expression data from mitotic Drosophila wild type cells	white wild type
☐	12	w67WT_3				brains and imaginal discs from w67 wild type 3rd instar Drosophila larvae, biological replicate 3	genotype: white wild type	Gene expression data from mitotic Drosophila wild type cells	white wild type
☐	13	OregonRWT_1				brains and imaginal discs from OregonR wild type 3rd instar Drosophila larvae, biological replicate 1	genotype: OregonR wild type	Gene expression data from mitotic Drosophila wild type cells	OregonR wild type
☐	14	OregonRWT_2				brains and imaginal discs from OregonR wild type 3rd instar Drosophila larvae, biological replicate 2	genotype: OregonR wild type	Gene expression data from mitotic Drosophila wild type cells	OregonR wild type
☐	15	OregonRWT_3				brains and imaginal discs from OregonR wild type 3rd instar Drosophila larvae, biological replicate 3	genotype: OregonR wild type	Gene expression data from mitotic Drosophila wild type cells	OregonR wild type

The columns named *1, *2, ... indicate the calls from the different outlier detection methods.

parnos.

Desde este documento, se puede acceder a los gráficos de medición de la calidad de cada una de las muestras. A continuación desarrollaremos algunas de ellas de manera global.

Gráficos de densidad

Mediante un histograma de densidad de Kernel, podemos hacernos una idea de las distribuciones de los distintos arrays del conjunto de datos. En la figura Podemos apreciar que, probablemente debido al gran número de arrays, todos siguen el mismo patrón de distribución de la señal.

Diagramas de cajas

El diagrama de cajas también no mostrará la distribución de las intensidades, en la figura se pueden apreciar pequeñas variaciones esperables en los datos normalizados.

Análisis Componentes Principales

Mediante el análisis de componentes principales podemos detectar si las muestras se agrupan entre otras muestras del mismo grupo o si no hay una clara correspondencia entre ellas. Que las muestras no se agrupen por "familias" podría ser debido al efecto *batch* por defectos técnicos.

Podemos realizar el análisis de componentes principales (ACP) de los datos, observando en el gráfico de la figura, la distribución de las dos primeras componentes de la expresión de cada gen (observaciones) sobre cada muestra (variables). Podemos ver como se distribuyen uniformemente a lo largo de la primera componente principal, salvo una perturbación que ocurre en valores altos de ambas componentes que habría que analizar.

```
FALSE Importance of components:
FALSE                             PC1     PC2     PC3    PC4     PC5     PC6     PC7
FALSE Standard deviation     13.5716 0.39882 0.34649 0.3322 0.22932 0.19722 0.18510
FALSE Proportion of Variance  0.9967 0.00086 0.00065 0.0006 0.00028 0.00021 0.00019
FALSE Cumulative Proportion   0.9967 0.99758 0.99823 0.9988 0.99911 0.99932 0.99950
FALSE                             PC8     PC9    PC10    PC11    PC12    PC13    PC14
FALSE Standard deviation     0.14542 0.12419 0.11269 0.10449 0.09683 0.09171 0.08711
FALSE Proportion of Variance 0.00011 0.00008 0.00007 0.00006 0.00005 0.00005 0.00004
FALSE Cumulative Proportion  0.99962 0.99970 0.99977 0.99983 0.99988 0.99993 0.99997
FALSE                            PC15
```

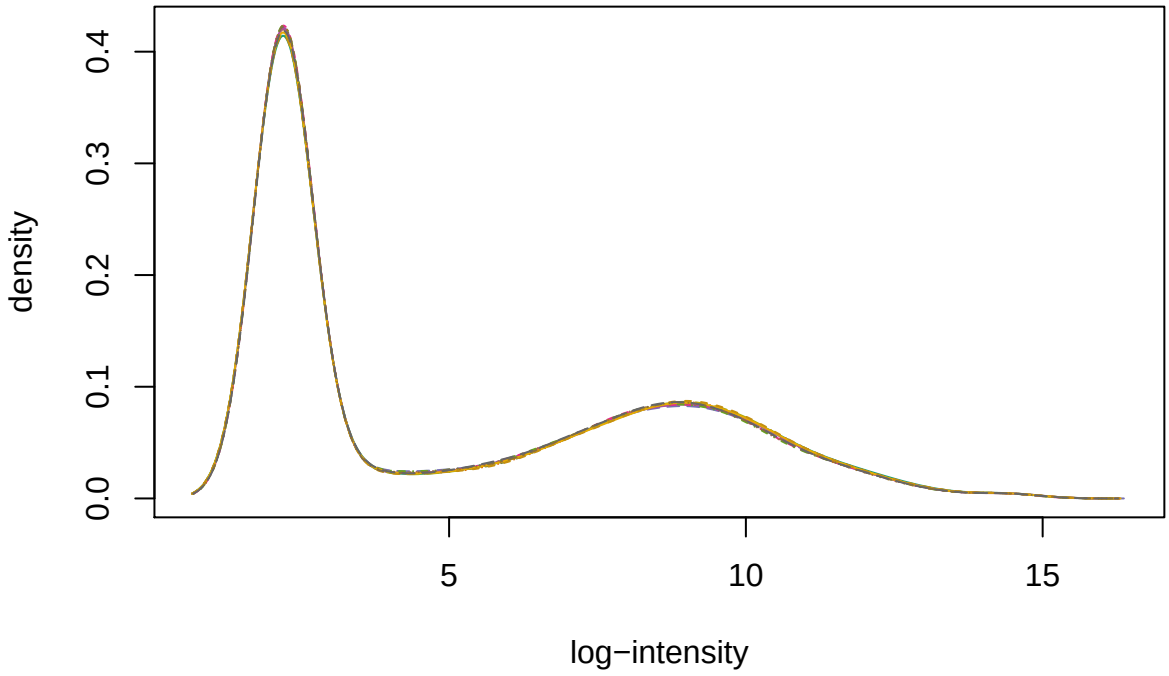


Figure 1: Histograma de los arrays del conjunto de datos.

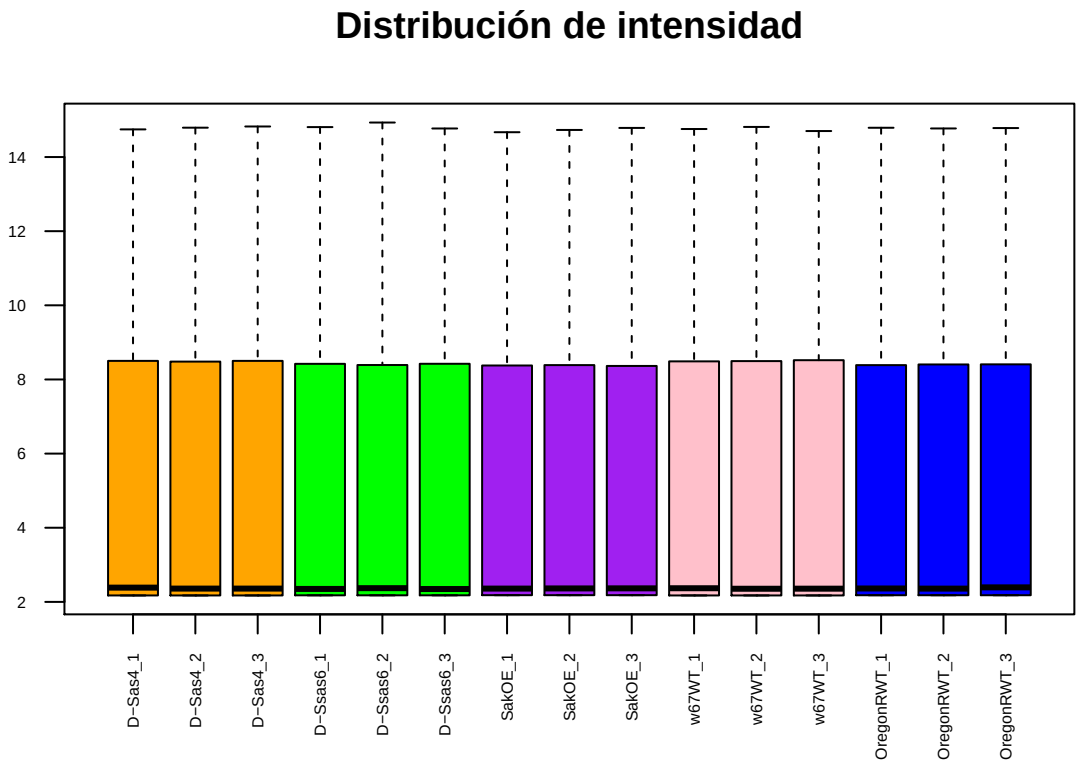


Figure 2: Diagramas de caja de la intensidad de los arrays para los datos normalizados.

```
FALSE Standard deviation     0.07759
FALSE Proportion of Variance 0.00003
FALSE Cumulative Proportion  1.00000

FALSE [1] 15 15
```

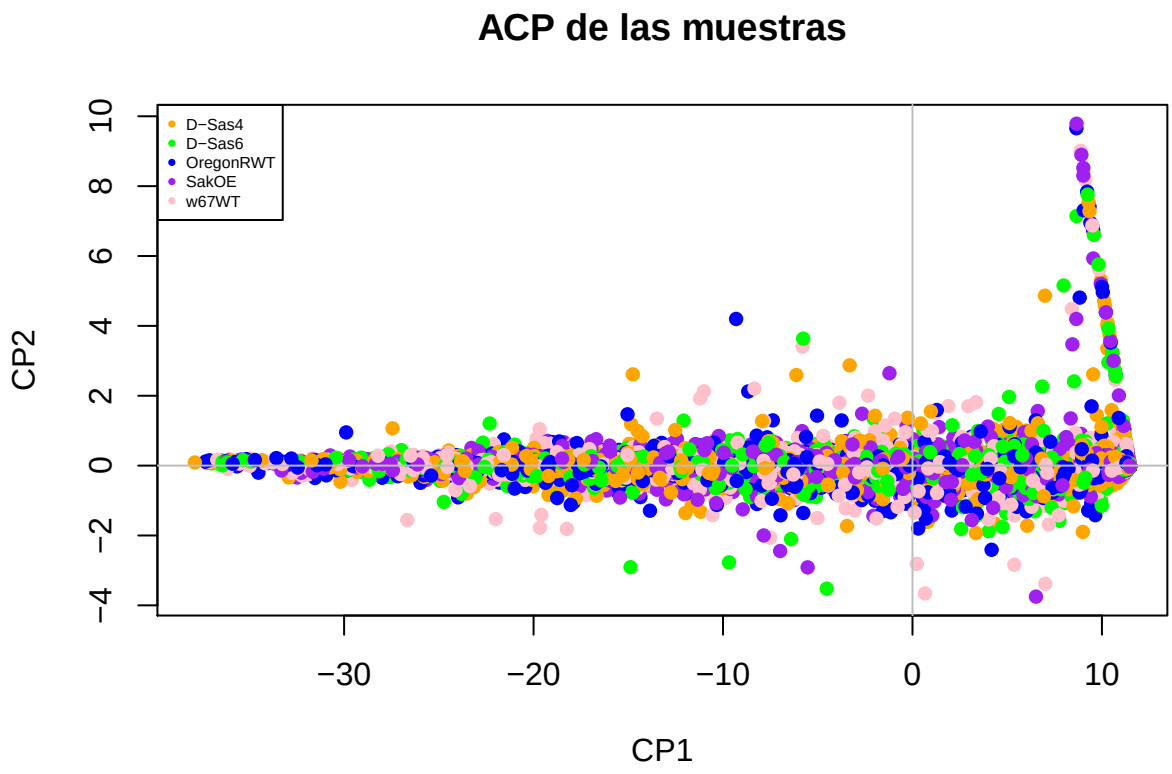


Figure 3: Dos primeras componentes principales de los datos sin procesar utilizando como variables la expresión en las muestras

Pero lo que nos interesa es considerar las diferentes muestras como observaciones, de forma que para cada muestra tenemos el perfil de expresión sobre todos los genes. De esta forma, podremos localizar rápidamente cuál es la principal fuente de variabilidad.

```
#convertimos en matriz los datos para poder analizar los CP
#transponemos la matriz para indicar que las muestras son las observaciones
#y los genes las variables
trawData_ACP <-prcomp(t(as.matrix(rawData)), center = TRUE, scale=FALSE)
```

El siguiente gráfico nos muestra las dos primeras componentes principales de la expresión sobre los genes de cada muestra. Se puede ver como una de las muestras para la línea salvaje OregonR no se agrupa con el resto de muestras de la línea. Esto puede deberse probablemente a errores técnicos a la hora de procesar las muestras.

Si eliminaramos del estudio esta muestra, el gráfico nos quedaría más acorde a lo que esperaríamos.

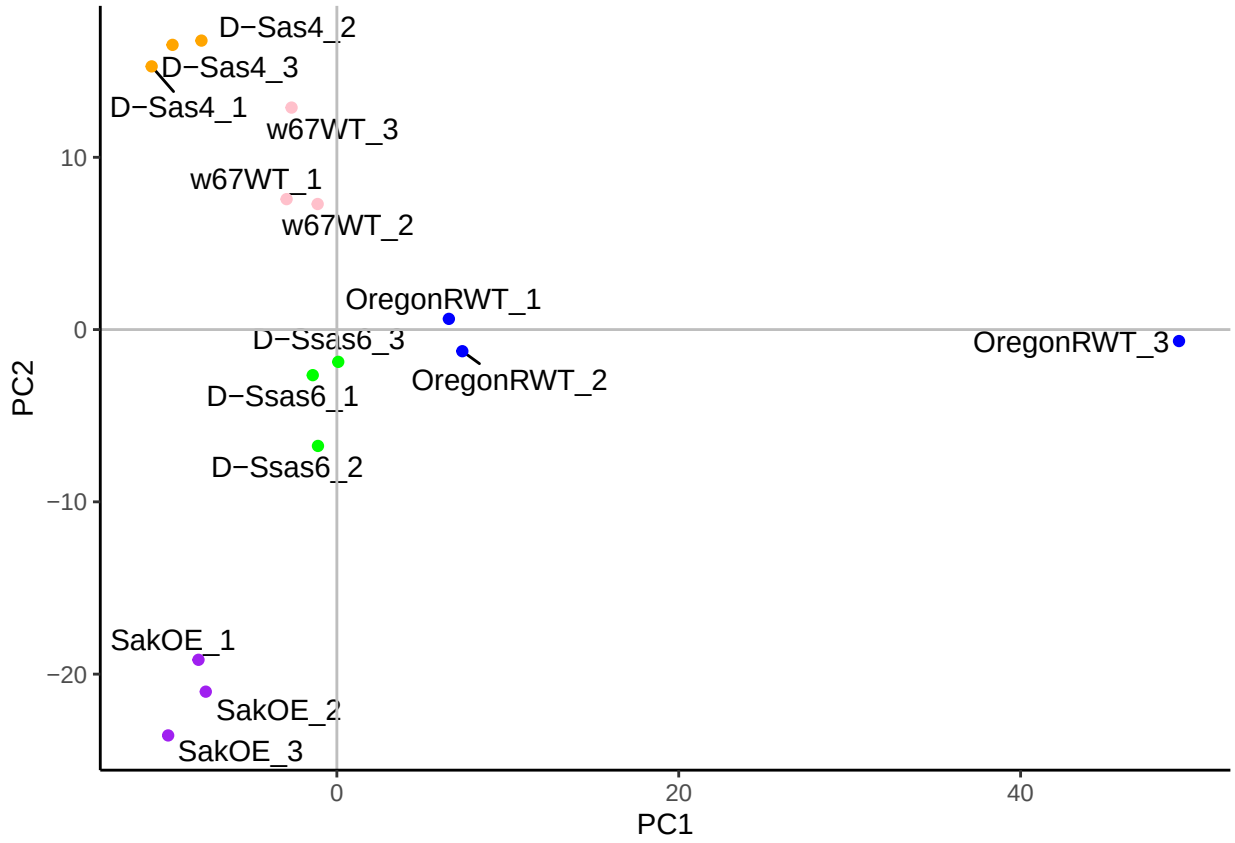
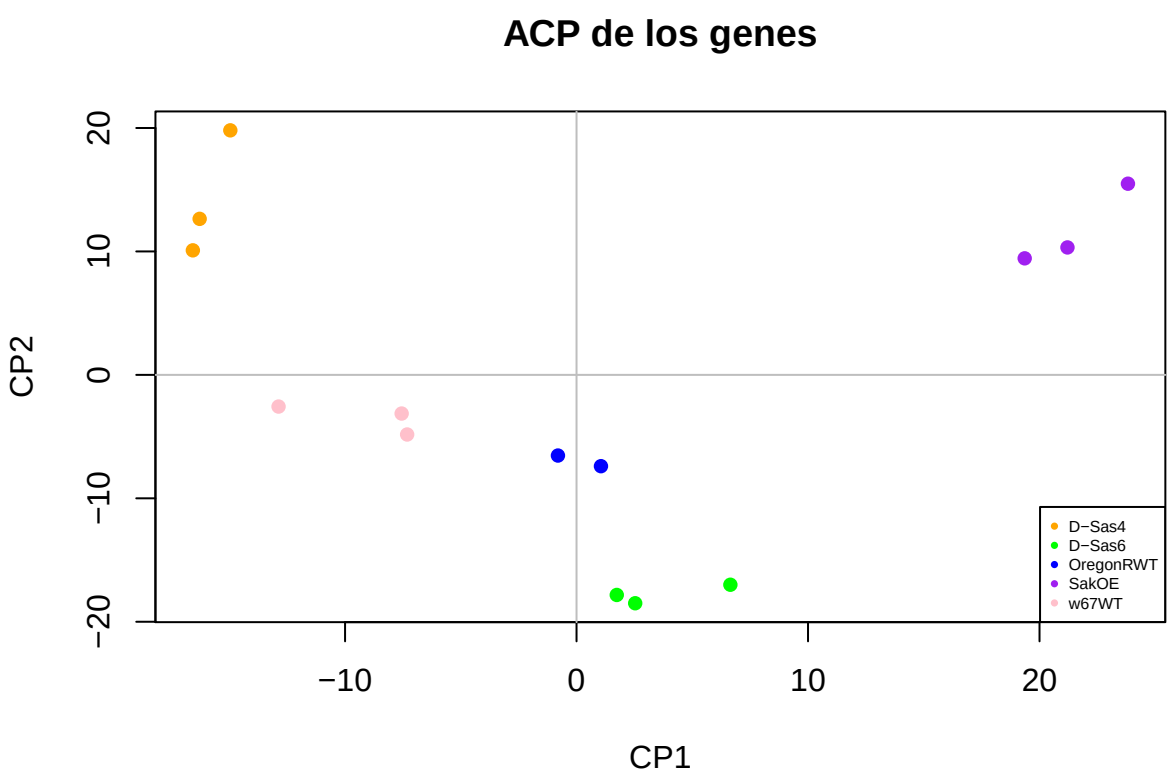


Figure 4: Dos primeras componentes principales de los datos sin procesar utilizando como variables la expresión de los genes

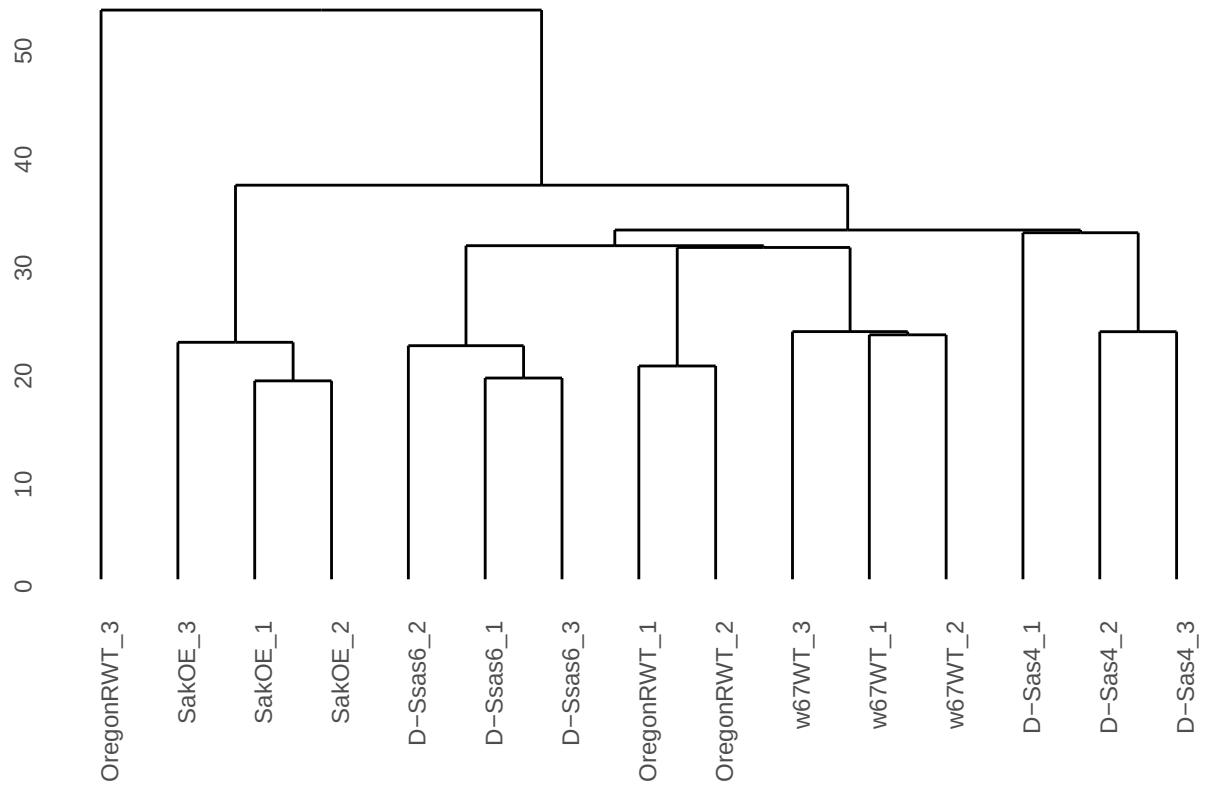


Se podría decir que eliminando la muestra discordante, los datos se agrupan mejor. Pero es muy raro que en un estudio se elimine un parámetro únicamente con una única prueba que indique algún tipo de problema. Antes de llegar a esos extremos, se siguen una serie de paso de análisis de los datos que sanearán el conjunto.

Clúster Jerárquico

Otra forma de asegurarnos que las muestra se agrupan según los grupos experimentales, es mediante un clúster jerárquico que nos agrupa las muestras por grado de similaridad. Mediante un dendograma, se va mostrando a qué valor se produce l aunión de los grupos y qué grupos se unen.

Como podemos observar, la muestra OregonRWT3 vuelve a quedarse aislada de las otras dos réplicas.



Normalización

Antes de comenzar cualquier análisis de datos en crudo, debemos transformarlos para corregir diferencias que pueda haber entre muestras. De esta forma hacemos que los arrays sean comparables unos con otros para poder determinar si una diferenciación es real o simplemente es debida a la escala de los datos originales.

Para llevar a cabo esta normalización primero tendremos que corregir el ruido de fondo. Tras la normalización, se procederá al resumen de los valores de cada grupo de sondas en un único valor de expresión para cada gen.

Al descargar los datos con getGEO, los datos ya aparecen normalzados. Lo podemos deducir al explorar el *boxplot* de expresión generado anteriormente. Es por esto que se pasará directamente al proceso de filtraje de datos.

Filtraje

El filtraje no específico permite hacer una criba de genes con poca variabilidad entre condiciones o con alguna otra característica que lleve a que nos interese eliminarlos.

Variabilidad genética

Para cada uno de los genes, tendremos un valor numérico (expresión)que nos indicará su abundancia. De cada una de las muestras, a su vez tenemos diversas covariables que las describen y que pueden ser categóricas, numéricas, temporales... y si la variable tiene dos categorías, entonces nos define dos grupos (control y tratamiento, por ejemplo).

Los grupos cuantiosos de genes suelen presentar una gran variabilidad de la expresión genética que debe ser ajustada mediante los valores p para poder determinar si hay diferencias entre la expresión de genes de dos (o más) grupos considerados.

Si un gen concreto se expresa de forma diferencia, se espera una cierta diferenciación entre los grupos y, por tanto, la varianza de dicho gen será mayor que la de aquellos que no presenten esta expresión diferencial. Al trazar la variabilidad genética de todos los genes es útil para decidir qué porcentaje de genes presenta una variabilidad que podría atribuirse a causas distintas de la variación aleatoria.

El siguiente gráfico representa las desviaciones estándar de todos los genes ordenados de menor a mayor valor, siendo aquellos genes con una desviación estándar superior al 90-95% los que se podrían considerar que tienen una variabilidad significativamente mayor que el resto.

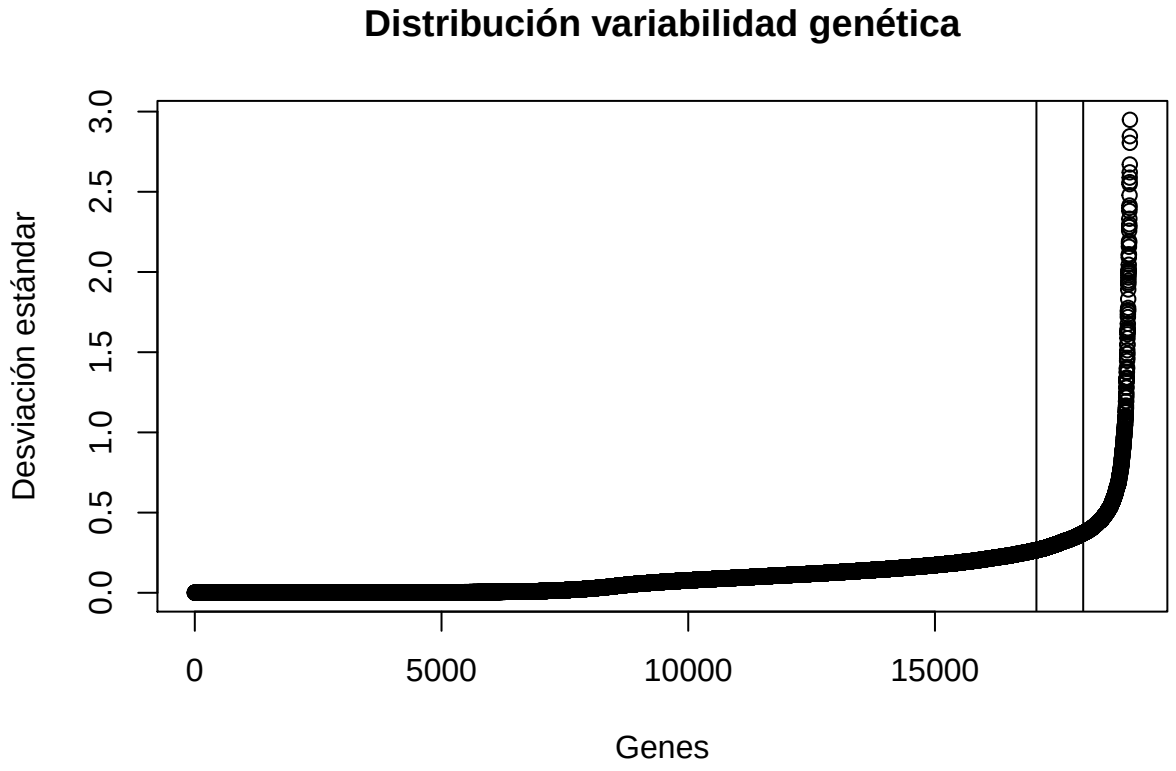


Figure 5: Distribución de la variabilidad genética en orden ascendente. Las líneas verticales representan los percentiles 90 y 95%

Para realizar el filtraje de genes expresados diferencialmente, utilizamos el paquete `genefilter`. Para ello necesitamos conocer primero el paquete de anotación que utilizan nuestros datos.

```
fun <- function(GDSDataobj){
  fulltitle<- Meta(GDSDataobj)$title
  title <- strsplit(fulltitle, "\\[|\\]")[[1]][2]
  title <- paste0(gsub("_|-| ", "", tolower(title)), ".db")
  title
}
eset <- getGEO("GSE35240")[[1]]
fun(getGEO(annotation(eset)))
```

```
FALSE [1] "drosophila2.db"
```

```
annotation(rawData) <- "drosophila2.db"
filtered <- nsFilter(rawData,
                            require.entrez = TRUE, remove.dupEntrez = TRUE,
                            var.filter = TRUE, var.func = IQR, var.cutoff = 0.75, filterByQuantile = TRU
```

La función `nsfilter` permite eliminar los genes que varían poco o de los cuales no se tiene anotación. Devuelve los valores filtrados en un nuevo `expressionSet` y un informe de los resultados del filtraje.

```
FALSE $numDupsRemoved
FALSE [1] 1799
FALSE
FALSE $numLowVar
FALSE [1] 9470
FALSE
FALSE $numRemoved.ENTREZID
FALSE [1] 4526
```

Después del filtraje han quedado 3157 genes disponibles para analizar.

```
FALSE ExpressionSet (storageMode: lockedEnvironment)
FALSE assayData: 3157 features, 15 samples
FALSE   element names: exprs
FALSE protocolData: none
FALSE phenoData
FALSE   sampleNames: D-Sas4_1 D-Sas4_2 ... OregonRWT_3 (15 total)
FALSE   varLabels: title geo_accession ... tissue:ch1 (41 total)
FALSE   varMetadata: labelDescription
FALSE featureData
FALSE   featureNames: 1637146_at 1632978_at ... 1627248_at (3157 total)
FALSE   fvarLabels: ID CLONE_ID_LIST ... Gene Ontology Molecular Function (16
FALSE     total)
FALSE   fvarMetadata: Column Description labelDescription
FALSE experimentData: use 'experimentData(object)'
FALSE   pubMedIds: 23213376
FALSE Annotation: drosophila2.db
```

Selección de genes

Para seleccionar los genes diferencialente expresados, podemos basanrons en diferentes aproximaciones. Para realizar este análisis, nos basaremos en un modelo lineal construido con el programa `limma` e implementado por Smyth(2015) para comparar la expresión genética entre los diferentes grupos.

Matriz de diseño

El primer paso para realizar este análisis es crear la matriz de diseño a partir de los datos filtrados. Ésta consiste en una matriz que describe la correspondencia de cada muestra al grupo experimental que le corresponde. Tendremos tantas líneas como muestras y en las columnas presentaremos los diferentes grupos. Cada fila contendrá un 1 en la columna correspondiente con su grupo.

Diseñamos la matriz

```
designMat <- model.matrix(~0+rawData_filtro$`genotype:ch1`, pData(rawData_filtro))
colnames(designMat)<- c("DSas4", "DSas6", "OregonRWT", "SakOE", "w67WT")
designMat
```

```
FALSE             DSas4 DSas6 OregonRWT SakOE w67WT
FALSE D-Sas4_1        1     0         0     0     0
FALSE D-Sas4_2        1     0         0     0     0
FALSE D-Sas4_3        1     0         0     0     0
FALSE D-Ssas6_1       0     1         0     0     0
FALSE D-Ssas6_2       0     1         0     0     0
FALSE D-Ssas6_3       0     1         0     0     0
FALSE SakOE_1         0     0         0     1     0
FALSE SakOE_2         0     0         0     1     0
FALSE SakOE_3         0     0         0     1     0
FALSE w67WT_1         0     0         0     0     1
FALSE w67WT_2         0     0         0     0     1
FALSE w67WT_3         0     0         0     0     1
FALSE OregonRWT_1     0     0         1     0     0
FALSE OregonRWT_2     0     0         1     0     0
FALSE OregonRWT_3     0     0         1     0     0
FALSE attr(,"assign")
FALSE [1] 1 1 1 1 1
FALSE attr(,"contrasts")
FALSE attr(,"contrasts")$`rawData_filtro$`genotype:ch1``
FALSE [1] "contr.treatment"
```

Matriz de contraste

Tras definir el modelo lineal a través de la matriz de diseño, podemos formular las preguntas de interés (comparaciones o contrastes entre los parámetros) La matriz de contraste nos servirá para describir als comparaciones entre grupos. Contendrá tantas columnas como comparaciones y en las líneas tendremos cada grupo. La comparación entre grupos se representa por "1" o "-1" en las filas de grupos a comparar y "0" en el resto.

En este caso Baumbach(2012) hacen las siguientes comparaciones:

- transcriptoma global perdida de centrosoma vs. normal:
- w67 WT vs DSas6
- Oregon-R WT vs DSas6
- w67 WT vs DSas4
- Oregon-R WT vs DSas4
- transcriptoma global sobreexpresión centrosoma vs normal:
- W67 vs SakOE
- Oregon-R WT vs SakOE
- w67WT vs Oregon-R WT

```
contrastMAt <- makeContrasts(
#comparamos un mutante con ambos controles
  w67vsSas4 = w67WT-DSas4,
  OrvsSas4 = OregonRWT-DSas4,
#lo mismo con el otro mutante
  w67vsSas6 = w67WT-DSas6,
```

```
  OrvsSas6 = OregonRWT-DSas6,
#y con el sobreexpresado
  w67vsOE = w67WT-SakOE,
  OrvsOE = OregonRWT-SakOE,
#Finalmente comparamos ambos controles
  w67vsOrv = w67WT - OregonRWT,
  levels=designMat)
contrastMAt
```

```
FALSE            Contrasts
FALSE Levels      w67vsSas4 OrvsSas4 w67vsSas6 OrvsSas6 w67vsOE OrvsOE w67vsOrv
FALSE   DSas4            -1       -1         0        0       0      0        0
FALSE   DSas6             0        0        -1       -1       0      0        0
FALSE   OregonRWT         0        1         0        1       0      1       -1
FALSE   SakOE             0        0         0        0      -1     -1        0
FALSE   w67WT             1        0         1        0       1      0        1
```

Estimación del modelo y selección de genes

Tras definir las matrices de diseño y contraste, estimamos el modelo, los contrastes y realizamos las pruebas de significación.

De nuevo, utilizamos el paquete `limma` para lelvar a cabo estas estimaciones. Este análisis nos proporcionará los estadísticos necesarios para rdenar los genes diferencialmente expresados.

```
fit <- lmFit(rawData_filtro, designMat)
fit.main <- contrasts.fit(fit, contrastMAt)
#regularización de la varianza para estimaciones de error mejoradas
fit.main <- eBayes(fit.main)
save(fit.main, file="./results/fit.main.Rda")
```

Ajustamos los p-valor para controlar el porcentaje de falsos positivos que puedan darse debido a la gran cantidad de contrastes que se realizarán simultáneamente segun Benjamini y Hochberg (1995).

La función `topTable`genera una lista de genes ordenados según su expresión diferencial para cada contraste. Nos proporciona los estadísticos siguientes:

Mostramos las primeras líneas de la tabla generada para la primera comparación: mutante D-Sas4 frente al tipo salvaje w^67:

```
FALSE                        ID     logFC   AveExpr         t      P.Value
FALSE 1637465_at     1637465_at -4.555681  3.277981 -74.70584 1.304845e-17
FALSE 1640426_at     1640426_at -4.081056  3.346456 -38.21471 4.491235e-14
FALSE 1635802_s_at 1635802_s_at -3.139877  4.063517 -34.89862 1.346752e-13
FALSE 1636055_at     1636055_at -7.140540  3.744406 -32.94172 2.704423e-13
FALSE 1626667_at     1626667_at -2.189375 10.198374 -20.11474 1.000496e-10
FALSE 1627558_at     1627558_at  2.771586  7.048168  20.07724 1.022903e-10
FALSE                  adj.P.Val        B
FALSE 1637465_at   4.119396e-14 27.16509
FALSE 1640426_at   7.089414e-11 22.08264
FALSE 1635802_s_at 1.417232e-10 21.19662
FALSE 1636055_at   2.134466e-10 20.61389
FALSE 1626667_at   5.382173e-08 15.17850
FALSE 1627558_at   5.382173e-08 15.15686
```

Anotación Genética

Creamos una función que nos generará la anotación genética de los genes almacenados en cada topTable.

```
annotatedTopTable <- function(topTab, anotPackage){
  topTab <- cbind(PROBEID=rownames(topTab), topTab)
  myProbes <- rownames(topTab)
  thePackage <- eval(parse(text = anotPackage))
  geneAnots <- select(thePackage, myProbes, c("SYMBOL", "ENTREZID", "GENENAME"))
  annotatedTopTab<- merge(x=geneAnots, y=topTab, by.x="PROBEID", by.y="PROBEID")
return(annotatedTopTab)
}
```

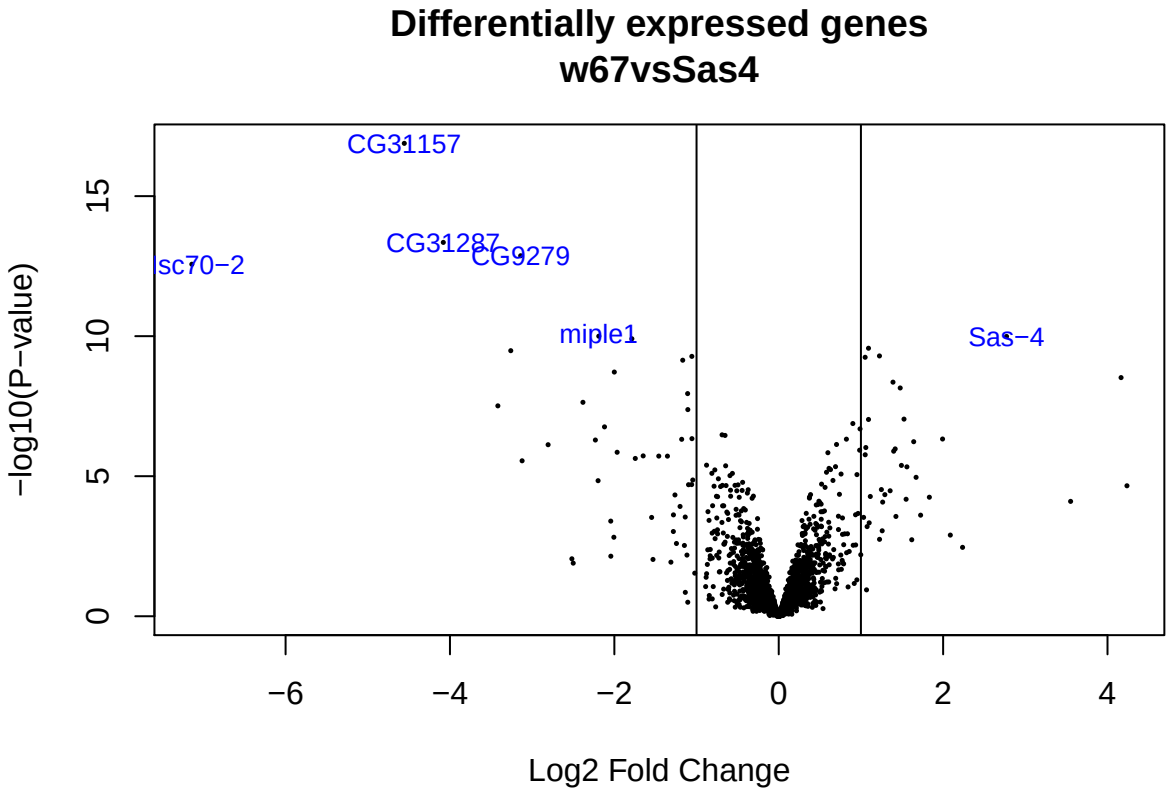
De esta forma, tenemos en una única tabla toda la información necesaria de cada una de nuestras comparaciones. La siguiente tabla nos muestra los datos para la comparación entre la línea salvaje OregonR y la línea sobreexpresada.

```
FALSE      PROBEID  SYMBOL ENTREZID
FALSE 1 1622896_at   pinta    42635
FALSE 2 1622901_at  CG9993    37358
FALSE 3 1622906_at    Sod3    36232
FALSE 4 1622909_at Pi3K21B    33203
FALSE 5 1622922_at    Nlg4    42402
FALSE 6 1622926_at  CG5776    34680
FALSE                                                          GENENAME
FALSE 1 prolonged depolarization afterpotential (PDA) is not apparent
FALSE 2                                          uncharacterized protein
FALSE 3                                           Superoxide dismutase 3
FALSE 4                                                          Pi3K21B
FALSE 5                                                    Neuroligin 4
FALSE 6                                          uncharacterized protein
```

Gráficos

Volcano Plot

Mediante un *volcano plot* podemos ver representados en abcisas los cambios de expresión en escala logarítmica y en ordenadas el estadístico p-valor en escala -log10. Como ejemplo, ilustramos la comparación entre la línea salvaje w67 y el mutante D-Sas4, marcando los seis primeros genes con valores más altos (mayor diferenciabilidad).



Comparaciones múltiples: Diagrama de Venn

Al realizar comparaciones, a veces nos resulta importante para el estudio ver qué genes cambian simultáneamente en más de una comparación. Mediante la función `decidetests` podemos realizar estas comparaciones a la vez que ajustamos los p-valor entre las mismas, seleccionando únicamente los genes que cambian en una o más condiciones.

Como resultado, obtendremos una tabla en la que cada comparación obtendrá un "1" (*up*) si el gen está sobreexpresado; un "0" (*NotSig*) si no se encuentran cambios significativos; o un "-1" (*Down*) si disminuye su expresión.

```
res <- decideTests(fit.main, method = "separate", adjust.method = "fdr", p.value = 0.05, lfc = 1)
summary(res)
```

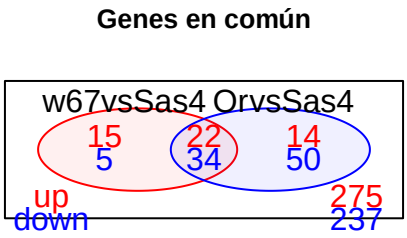
```
FALSE        w67vsSas4 OrvsSas4 w67vsSas6 OrvsSas6 w67vsOE OrvsOE w67vsOrv
FALSE Down          39       84        27       49      38     54        13
FALSE NotSig      3081     3037      3081     3083    3046   3047      3117
FALSE Up            37       36        49       25      73     56        27
```

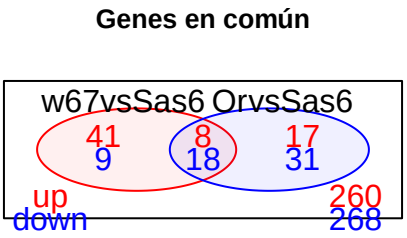
Para resumir el análisis contamos las fila que tienen como mínimo una celda distinta de "0".

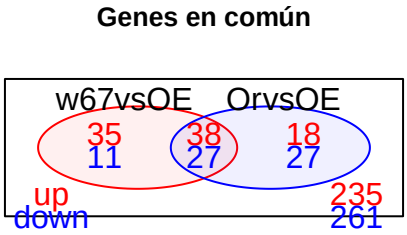
```
FALSE TestResults matrix
FALSE             Contrasts
FALSE              w67vsSas4 OrvsSas4 w67vsSas6 OrvsSas6 w67vsOE OrvsOE w67vsOrv
FALSE   1637146_at         0        0         0        0       0      0        1
```

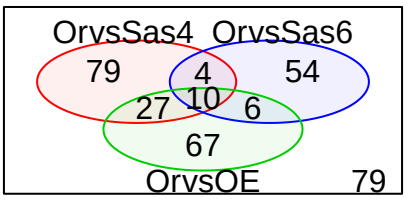
```
FALSE   1633109_at          0         0          1         0         0         0         0
FALSE   1639896_at          0        -1          0         0         1         1         0
FALSE   1633520_at          1         1          0         0         1         1         0
FALSE   1628668_at          0         0          0         0         0        -1         0
FALSE   1624862_at          0         1          0         0         0         0         0
```

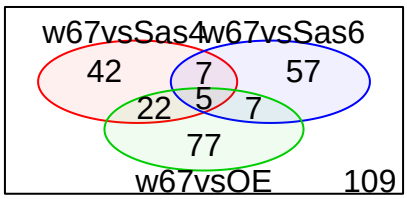
Y con un **diagrama de Venn** podemos visualizar mejor estos resultados.











Perfiles de Expresión: Mapas de color

Las expresiones de cada gen pueden visualizarse agrupándolas para destacar aquellos genes que se encuentran sobre o infra regulados simultáneamente y generando mapas de color o `Heatmaps`.

Seleccionamos todos aquellos genes que hayan resultado diferencialmente expresados en alguna de las comparaciones.

```
probeNames <- rownames(res.selected)
HMdata <- exprs(rawData_filtro)[probeNames,]
geneSymbols2 <- select(drosophila2.db, rownames(HMdata), c("SYMBOL"))
SYMBOLS2 <- geneSymbols2$SYMBOL
rownames(HMdata) <- SYMBOLS2
write.csv(HMdata, file=file.path("./results/data4Heatmap.csv"))

#creamos el heatmaps
color.map <- colorRampPalette(c("yellow", "brown"))(n=299)
heatmap.2(HMdata, scale="row",
```

```
          Rowv = TRUE,
          Colv = TRUE,
          col=color.map,
          sepcolor = "white",
           cexCol = 0.5,
          dendrogram = "both",
          density.info = "histogram")
```

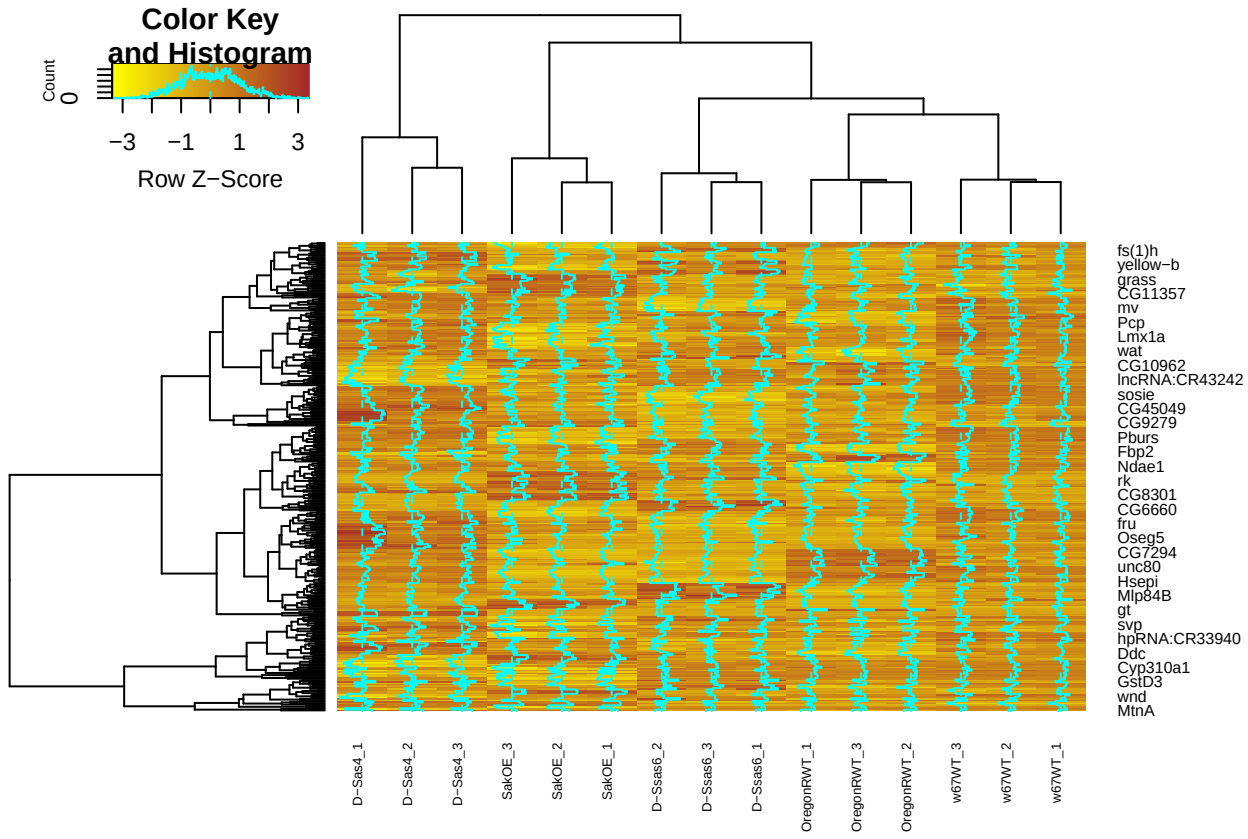


Figure 6: Mapa de color de la expresión de los datos agrupados por similitud de genes (filas) y muestras (columnas).

Significancia Biológica: Análisis de enriquecimiento

Este análisis toma como entrada los identificadores *Entrez* de la lista de genes seleccionada y el nombre del paquete de la anotación correspondiente, y la salida que se obtiene es la lista de categorías que representa cada conjunto seleccionado.

Utilizaremos el paquete `GOstats`para llevar a cabo este análisis. El primer paso será preparar la lista de genes que será analizada. Haremos una selección más permisiva que la realizada anteriormente para tener un análisis más fiable.

```
listOfTables <- list(w67vsSas4=topTab_w67vsSas4,
    OrvsSas4=topTab_OrvsSas4,
    w67vsSas6=topTab_w67vsSas6,
```

```
    OrvsSas6=topTab_OrvsSas6,
    w67vsOE=topTab_w67vsOE,
    OrvsOE=topTab_OrvsOE,
    w67vsOrv=topTab_w67vsOrv)
listOfSelected<- list()
for(i in 1:length(listOfTables)){

#seleccionamos la topTable
  topTab <- listOfTables[[i]]

#seleccionamos los genes que incluiremos en el análisis
  whichGenes<-topTab["adj.P.Val"]<0.1
  selectedIDs <- rownames(topTab)[whichGenes]

# convertimos ID en Entrez
  EntrezIDs<- select(drosophila2.db, selectedIDs, c("ENTREZID"))
  EntrezIDs <- EntrezIDs$ENTREZID
  listOfSelected[[i]] <- EntrezIDs
  names(listOfSelected)[i] <- names(listOfTables)[i]
  }
sapply(listOfSelected, length)
```

```
FALSE w67vsSas4  OrvsSas4 w67vsSas6  OrvsSas6   w67vsOE    OrvsOE  w67vsOrv
FALSE       380      1678      1497      1267      1790      1358      1232
```

Ahora utilizaremos todos los genes de los que disponemos en el estudio para definir el "universo" con aquellos genes que al menos presenten una anotación en Gene Ontology.

Por cuestiones de tiempo y espacio, se realizará la significación biólogia de una de las comparaciones.

```
FALSE ####################################
FALSE Comparison:  w67vsSas4
FALSE ####################################
FALSE Comparison:  OrvsSas4
FALSE ####################################
FALSE Comparison:  w67vsSas6
FALSE ####################################
FALSE Comparison:  OrvsSas6
FALSE ####################################
FALSE Comparison:  w67vsOE
FALSE ####################################
FALSE Comparison:  OrvsOE
FALSE ####################################
FALSE Comparison:  w67vsOrv
```

Por algún motivo que escapa a mis conocimientos bioinformáticos, no soy capaz de generar el análisis.

Resultados

La siguiente tabla muestra la lista de todos los archivos generados con los resultados del análisis.

Table 1: Archivos de resultados

Lista de resultados
data4Heatmap.csv
fit.main.Rda
normalized.Data.csv
normalized.Filtered.Data.csv
rawData_filtro.Rda
rawData_quality
topAnnotated_OrvsOE.csv
topAnnotated_OrvsSas4.csv
topAnnotated_OrvsSas6.csv
topAnnotated_w67vsOE.csv
topAnnotated_w67vsOrv.csv
topAnnotated_w67vsSas4.csv
topAnnotated_w67vsSas6.csv

Bibliografía

Guillermo Ayala. Bioinformática EStadística. Análisis estadístico de datos ómicos. Material docente Universidad de Valencia. Consultado: Mayo 2020 Uhttps://www.uv.es/ayala/docencia/tami/tami13.pdf

Baumbach, Janina, Mitchell P Levesque, and Jordan W Raff. 2012. "Centrosome Loss or Amplification Does Not Dramatically Perturb Global Gene Expression in Drosophila." *Biology Open* 1 (10). The Company of Biologists Ltd: 983–93.

Benjamini, Yoav, and Yosef Hochberg. 1995. "Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing." *Journal of the Royal Statistical Society: Series B (Methodological)* 57 (1). Wiley Online Library: 289–300.

Ritchie, Matthew E., Belinda Phipson, Di Wu, Yifang Hu, Charity W. Law, Wei Shi, and Gordon K. Smyth. 2015. "limma powers differential expression analyses for RNA-sequencing and microarray studies." *Nucleic Acids Research* 43 (7): e47–e47. doi:10.1093/nar/gkv007.