PEC 1

Análisis de Datos Ómicos

Máster de Bioinformática y Bioestadística UOC y UB

Ana M Gómez Martínez

29 de abril, 2021

Contents

Abstract

En este informe se plantean las cuestiones que deseamos responder (Objetivos) con nuestro estudio, se realizan los análisis necesarios (Pipeline del análisis, qué se ha hecho en cada paso y resultados) y una discusión (Discusión) de los mismos.

Así pues, vamos a investigar el Síndrome de Turner (ST), que es el trastorno de aneuploidía del cromosoma X más común en las mujeres. Su cariotipo predominante es `45X`, una pérdida completa del segundo cromosoma sexual. Según el origen parental del cromosoma X único, los pacientes con 45X se pueden dividir en dos grupos: `45Xm` (cromosoma X heredado materno) y `45Xp` (heredado paterno).

Objetivos

Queremos ver el impacto del **cromosoma X de los padres en el fenotipo ST**, ya que se ha encontrado que los pacientes con ST de `45Xm` y `45Xp` se asocian con diferentes grados de gravedad en el fenotipo. Para ello, vamos a analizar la expresión génica diferencial de `45Xm` y `45Xp` mediante

microarrays, analizando también la expresión génica para `46XX` hembra normal, **con el fin de investigar los cambios en la expresión génica de todo el genoma entre pacientes con monosomía `X` ST y mujeres normales**.

Materiales y métodos

Nota: El código completo se muestra en el Apéndice, ya que ahora, para facilitar la lectura, mostraremos solo la parte del código más relevante.

Naturaleza de los datos, tipo de experimento, diseño experimental, tipo de microarrays utilizados

Los datos se encuentran en GEO con identificador `GSE46687`, donde obtuvimos la información del estudio, en el que se realizaron perfiles de expresion de arrays. Este es el link. De ahí pudimos obtener los archivos *.CEL* para realizar el estudio. Como hemos visto en teoría, estos archivos son datos de bajo nivel de Arrays de un color. Más información acerca del tipo del array: [HG-U133_Plus_2] Affymetrix Human Genome U133 Plus 2.0 Array

Con respecto a nuestro diseño experimental, tenemos 18 unidades experimentales (individuos), y trabajaremos con el factor `cariotipo`, que tiene 3 niveles (3 grupos), que son `45Xp`, `45Xm` y `46XX`. Cada nivel tiene un total de 6 réplicas, las cuales hemos escogido al azar (`selectSamples(77146010)`), como vemos en el siguiente apartado de *Selección de muestras*.

Selección de muestras

Seleccionamos las muestras con las que trabajaremos:

```
mySelected <- selectSamples(77146010)
targetsAll <-read.csv(file="targetsAll.csv", row.names = 1, head=TRUE)
myTargets <- targetsAll[mySelected,]
```

```
library(GEOquery)
filePaths = getGEOSuppFiles("GSE46687")

library(oligo)
celFiles <- list.celfiles("./data2", full.names = TRUE)

library(Biobase)
my.targets <-read.AnnotatedDataFrame(file.path("./data2","myTargets.csv"),
                                     header = TRUE, row.names = 1, sep=",")

rawData <- read.celfiles(celFiles, phenoData = my.targets)
```

Pipeline del análisis, qué se ha hecho en cada paso y Resultados

NO ES PRECISO entrar en el detalle de los métodos, más bien hacer una descripción cualitativa indicando por qué se ha llevado a cabo cada paso, y cual ha sido el "input" suministrado al procedimiento y el "output" obtenido.

→ Lo he organizado así porque considero que facilita la lectura y además conseguimos que el documento no sea tan extenso.

1. Identificar qué grupos hay y a qué grupo pertenece cada muestra.

Echando un vistazo a `my.targets` vemos que hay 3 grupos según el cariotipo, como ya hemos comentado previamente, y el nombre de las muestras aparece en la columna de `X.title.`. Por tanto, nombraremos las columnas del `rawData` con estos nombres de las muestras y mostraremos el `head`. Es importante que nos fijemos en el parámetro `Annotation`, nos servirá para más adelante.

```
colnames(rawData) <- my.targets@data$X.title.
head(rawData)
```

```
## ExpressionFeatureSet (storageMode: lockedEnvironment)
## assayData: 6 features, 18 samples
##   element names: exprs
## protocolData
##   rowNames: "XX_rep1" "XX_rep2" ... "Xp_rep10" (18 total)
##   varLabels: exprs dates
##   varMetadata: labelDescription channel
## phenoData
##   rowNames: "XX_rep1" "XX_rep2" ... "Xp_rep10" (18 total)
##   varLabels: X.title. X.karyotype.
##   varMetadata: labelDescription channel
## featureData: none
## experimentData: use 'experimentData(object)'
## Annotation: pd.hg.u133.plus.2
```

2. Exploración y control de calidad de los datos crudos

El siguiente paso es hacer un control de calidad del rawData. Para ello, utilizamos la función `arrayQualityMetrics`.

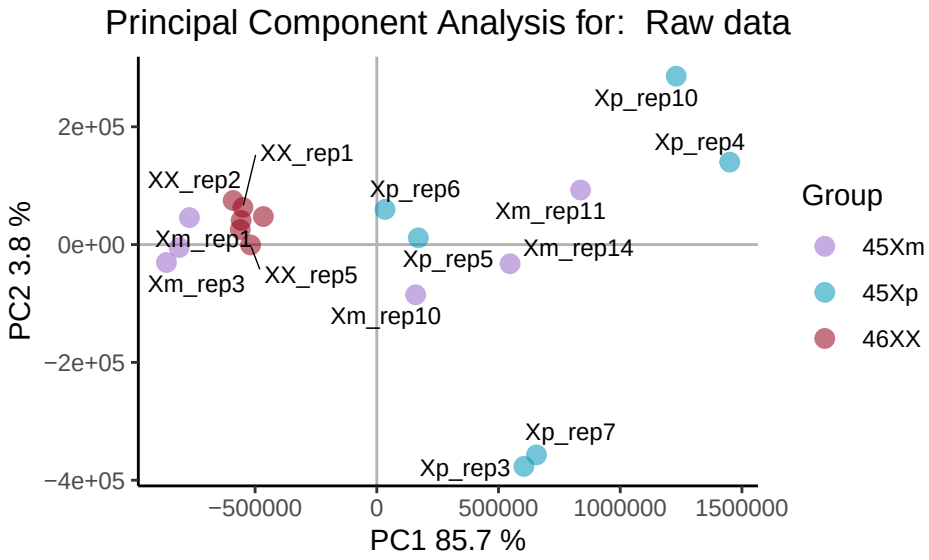
```
library(arrayQualityMetrics)
arrayQualityMetrics(rawData)
```

	array	sampleNames	*1	*2	*3	X.title.	X.karyotype.
☐	1	"XX_rep1"			x	"XX_rep1"	"46XX"
☐	2	"XX_rep2"				"XX_rep2"	"46XX"
☐	3	"XX_rep3"			x	"XX_rep3"	"46XX"
☐	4	"XX_rep5"			x	"XX_rep5"	"46XX"
☐	5	"XX_rep8"			x	"XX_rep8"	"46XX"
☐	6	"XX_rep9"				"XX_rep9"	"46XX"
☐	7	"Xm_rep1"			x	"Xm_rep1"	"45Xm"
☐	8	"Xm_rep3"			x	"Xm_rep3"	"45Xm"
☐	9	"Xm_rep5"			x	"Xm_rep5"	"45Xm"
☐	10	"Xm_rep10"				"Xm_rep10"	"45Xm"
☐	11	"Xm_rep11"			x	"Xm_rep11"	"45Xm"
☐	12	"Xm_rep14"			x	"Xm_rep14"	"45Xm"
☐	13	"Xp_rep3"			x	"Xp_rep3"	"45Xp"
☐	14	"Xp_rep4"	x		x	"Xp_rep4"	"45Xp"
☐	15	"Xp_rep5"			x	"Xp_rep5"	"45Xp"
☐	16	"Xp_rep6"			x	"Xp_rep6"	"45Xp"
☐	17	"Xp_rep7"			x	"Xp_rep7"	"45Xp"
☐	18	"Xp_rep10"	x		x	"Xp_rep10"	"45Xp"

Fig.1: Estudio de calidad de nuestros datos crudos

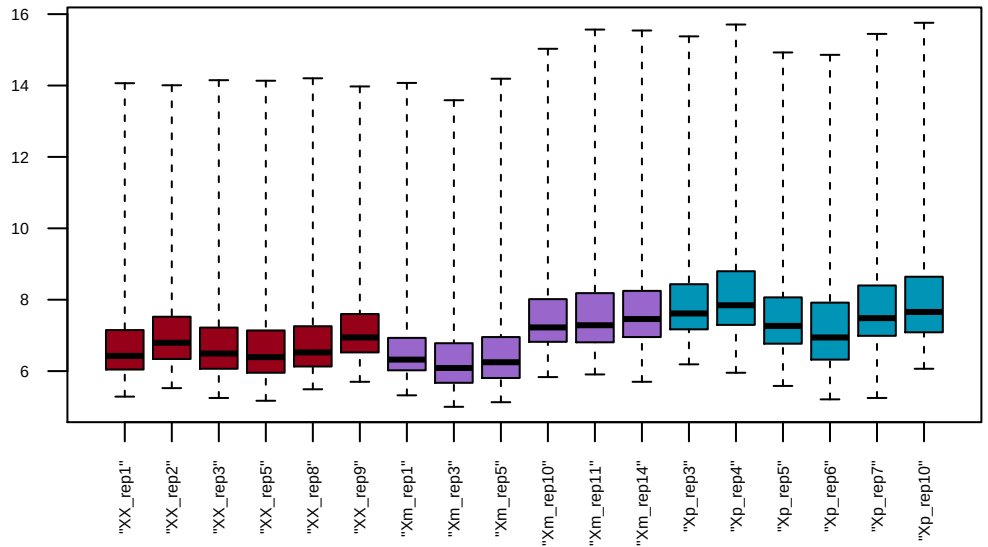
Hay un par de muestras, `Xp_rep4` y `Xp_rep10` que tienen dos cruces (outliers detectados por dos métodos distintos), lo cual no es muy bueno.

Por otro lado, mediante un Análisis de Componentes Principales, obtenemos el siguiente gráfico:



Vemos que para el grupo normal `46XX` y el `45Xp`, sí que se observan diferencias claras en cuanto a la distribución en la gráfica.

También podemos hacer un boxplot:



Vemos que la distribución no es muy homogénea, por lo que normalizar es buena idea.

3. Normalización

Esta normalización puede realizarse fácilmente con la función `rma`, a la cual le damos los datos crudos, y obtenemos los normalizados.

```
eset_rma <- rma(rawData)
```

4. [Control de calidad de los datos normalizados] (opcional)

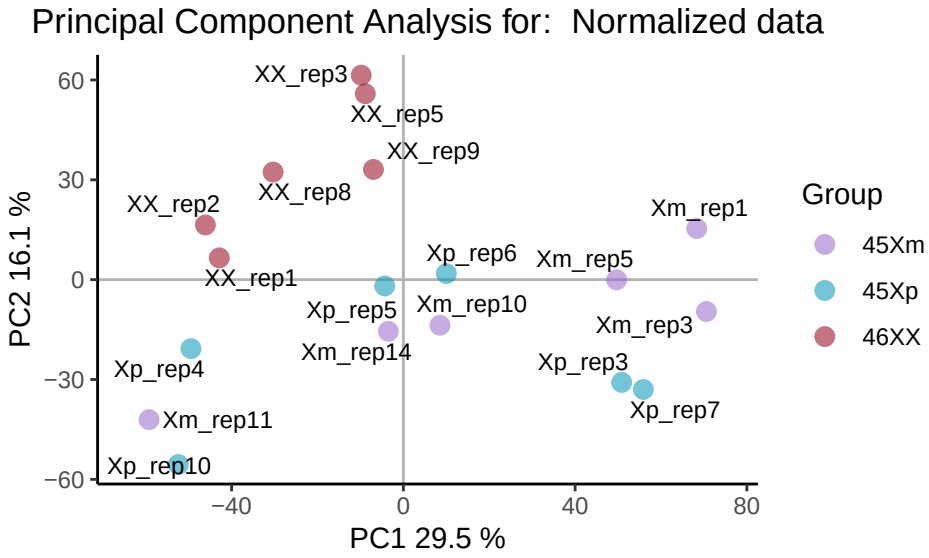
Como ya hemos hecho anteriormente, para medir la calidad podemos volver a emplear la función `arrayQualityMetrics`. Esta vez la utilizaremos sobre los datos normalizados `eset_rma`, y esto es lo que obtenemos:

	array	sampleNames	*1	*2	*3	X.title.	X.karyotype.
☐	1	"XX_rep1"				"XX_rep1"	"46XX"
☐	2	"XX_rep2"				"XX_rep2"	"46XX"
☐	3	"XX_rep3"				"XX_rep3"	"46XX"
☐	4	"XX_rep5"				"XX_rep5"	"46XX"
☐	5	"XX_rep8"				"XX_rep8"	"46XX"
☐	6	"XX_rep9"				"XX_rep9"	"46XX"
☐	7	"Xm_rep1"	x			"Xm_rep1"	"45Xm"
☐	8	"Xm_rep3"				"Xm_rep3"	"45Xm"
☐	9	"Xm_rep5"				"Xm_rep5"	"45Xm"
☐	10	"Xm_rep10"				"Xm_rep10"	"45Xm"
☐	11	"Xm_rep11"				"Xm_rep11"	"45Xm"
☐	12	"Xm_rep14"				"Xm_rep14"	"45Xm"
☐	13	"Xp_rep3"				"Xp_rep3"	"45Xp"
☐	14	"Xp_rep4"				"Xp_rep4"	"45Xp"
☐	15	"Xp_rep5"				"Xp_rep5"	"45Xp"
☐	16	"Xp_rep6"				"Xp_rep6"	"45Xp"
☐	17	"Xp_rep7"				"Xp_rep7"	"45Xp"
☐	18	"Xp_rep10"				"Xp_rep10"	"45Xp"

Fig.2: Estudio de calidad de nuestros datos normalizados

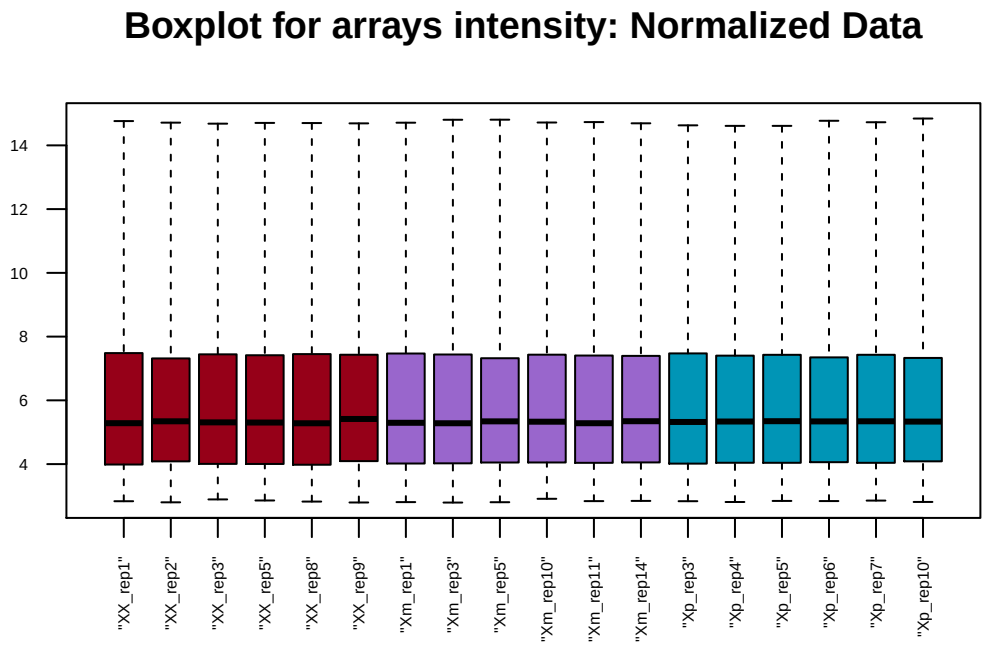
Ahora solo hay una muestra con una cruz.

También podemos hacer otra vez en PCA:



Observamos que hay diferencia entre el grupo normal `46XX` con los otros dos grupos, que es lo que a priori esperaríamos.

Con respecto al Boxplot:



Ahora sí presentan una distribución homogénea.

5. Filtraje no específico [opcional]

El filtraje no específico nos permite eliminar genes que varían poco entre condiciones o que queremos quitar por otras razones, como por ejemplo, que no tengamos anotaciones sobre ellos. Así también podemos eliminar ruido de fondo. Aunque hay controversia con este paso, por que se puede eliminar información de forma no intencionada.

Es importante filtrar según los datos de anotación del paquete correspondiente, en este caso: `hgu133plus2.db`. Aquí, estamos anotando los datos normalizados `eset_rma` y los filtramos con la función `nsFilter`, obteniendo ahora una selección de genes que nosotros hemos indicado en el código.

```
library(genefilter)
# BiocManager::install("hgu133plus2.db")
library(hgu133plus2.db)
annotation(eset_rma) <- "hgu133plus2.db"
filtered <- nsFilter(eset_rma,
                     require.entrez = TRUE, remove.dupEntrez = TRUE,
                     var.filter=TRUE, var.func=IQR, var.cutoff=0.75,
                     filterByQuantile=TRUE, feature.exclude = "^AFFX")
```

```
eset_filtered <-filtered$eset
eset_filtered
```

```
## ExpressionSet (storageMode: lockedEnvironment)
## assayData: 5040 features, 18 samples
##   element names: exprs
## protocolData
##   rowNames: "XX_rep1" "XX_rep2" ... "Xp_rep10" (18 total)
##   varLabels: exprs dates
##   varMetadata: labelDescription channel
## phenoData
##   rowNames: "XX_rep1" "XX_rep2" ... "Xp_rep10" (18 total)
##   varLabels: X.title. X.karyotype.
##   varMetadata: labelDescription channel
## featureData: none
## experimentData: use 'experimentData(object)'
## Annotation: hgu133plus2.db
```

Ahora tenemos 5040 genes.

6. Identificación de genes diferencialmente expresados

Lo primero que haremos es la matriz de diseño:

```
##            Xm45 Xp45 XX46
## "XX_rep1"    0    0    1
## "XX_rep2"    0    0    1
## "XX_rep3"    0    0    1
## "XX_rep5"    0    0    1
## "XX_rep8"    0    0    1
```

```
## "XX_rep9"     0    0    1
## "Xm_rep1"     1    0    0
## "Xm_rep3"     1    0    0
## "Xm_rep5"     1    0    0
## "Xm_rep10"    1    0    0
## "Xm_rep11"    1    0    0
## "Xm_rep14"    1    0    0
## "Xp_rep3"     0    1    0
## "Xp_rep4"     0    1    0
## "Xp_rep5"     0    1    0
## "Xp_rep6"     0    1    0
## "Xp_rep7"     0    1    0
## "Xp_rep10"    0    1    0
## attr(,"assign")
## [1] 1 1 1
## attr(,"contrasts")
## attr(,"contrasts")$X.karyotype.
## [1] "contr.treatment"
```

A continuación, haremos la matriz de contraste. Queremos ver si hay diferencias entre los grupos 46XX, 65Xm y 45 Xp. Por tanto, nuestra matriz de contraste es la siguiente:

```
##       Contrasts
## Levels Xm45vsXp45 Xm45vsXX46 Xp45vsXX46
##   Xm45          1          1          0
##   Xp45         -1          0          1
##   XX46          0         -1         -1
```

El siguiente paso es hacer la estimación del modelo, para el cual utilizaremos las funciones `lmFit`, `contrasts.fit` y `eBayes`. Así, con nuestros datos normalizados y filtrados `eset_filtered` y con la matriz de diseño y contraste, será posible hacer el modelo.

```
library(limma)
fit<-lmFit(eset_filtered, designMat)
fit.main<-contrasts.fit(fit, cont.matrix)
fit.main<-eBayes(fit.main)
```

Por tanto, ahora con la función topTable, podemos obtener los DEGs (genes diferencialmente expresados), entre los grupos que nos interesan. Mostraremos el `head` de `topTab.Xm45vsXp45`

```
head(topTab.Xm45vsXp45)
```

```
##                  logFC  AveExpr         t      P.Value adj.P.Val         B
## 235982_at   -1.6910638 6.891623 -5.144126 3.904396e-05 0.1967816 -3.550486
## 201688_s_at -0.9871805 7.636514 -4.150662 4.298306e-04 0.9422899 -3.781208
## 228377_at   -1.0642186 6.406992 -3.645711 1.454836e-03 0.9422899 -3.914503
## 220377_at   -0.7639219 7.625188 -3.563446 1.771707e-03 0.9422899 -3.937076
## 238532_at   -0.6264659 4.614611 -3.549825 1.830352e-03 0.9422899 -3.940834
## 224715_at   -0.7593637 6.184749 -3.523140 1.950817e-03 0.9422899 -3.948212
```

7. Anotación de los resultados

En este paso, creamos la función `annotatedTopTable`, que anotará los genes almacenados en las variables topTable (`topTab.Xm45vsXp45`, `topTab.Xm45vsXX46` y `topTab.Xp45vsXX46`) que acabamos de obtener justo en el paso anterior, con el paquete correspondiente `hgu133plus2.db`.

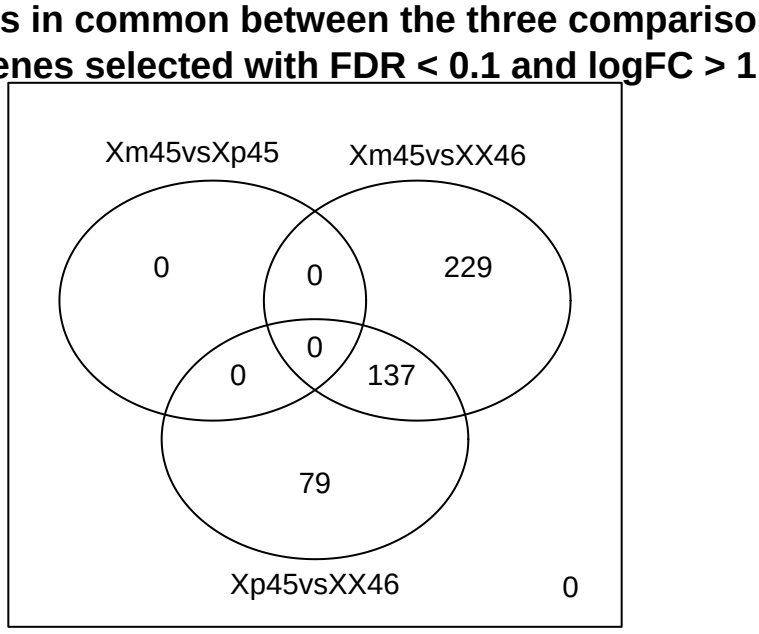
8. Comparación entre distintas comparaciones (si hay más de una comparación, ver qué genes han sido seleccionados en más de una comparación)

En este paso utilizaremos la función `decideTests` del paquete `limma`

```
##        Xm45vsXp45 Xm45vsXX46 Xp45vsXX46
## Down            0        314        169
## NotSig       5040       4674       4824
## Up              0         52         47
```

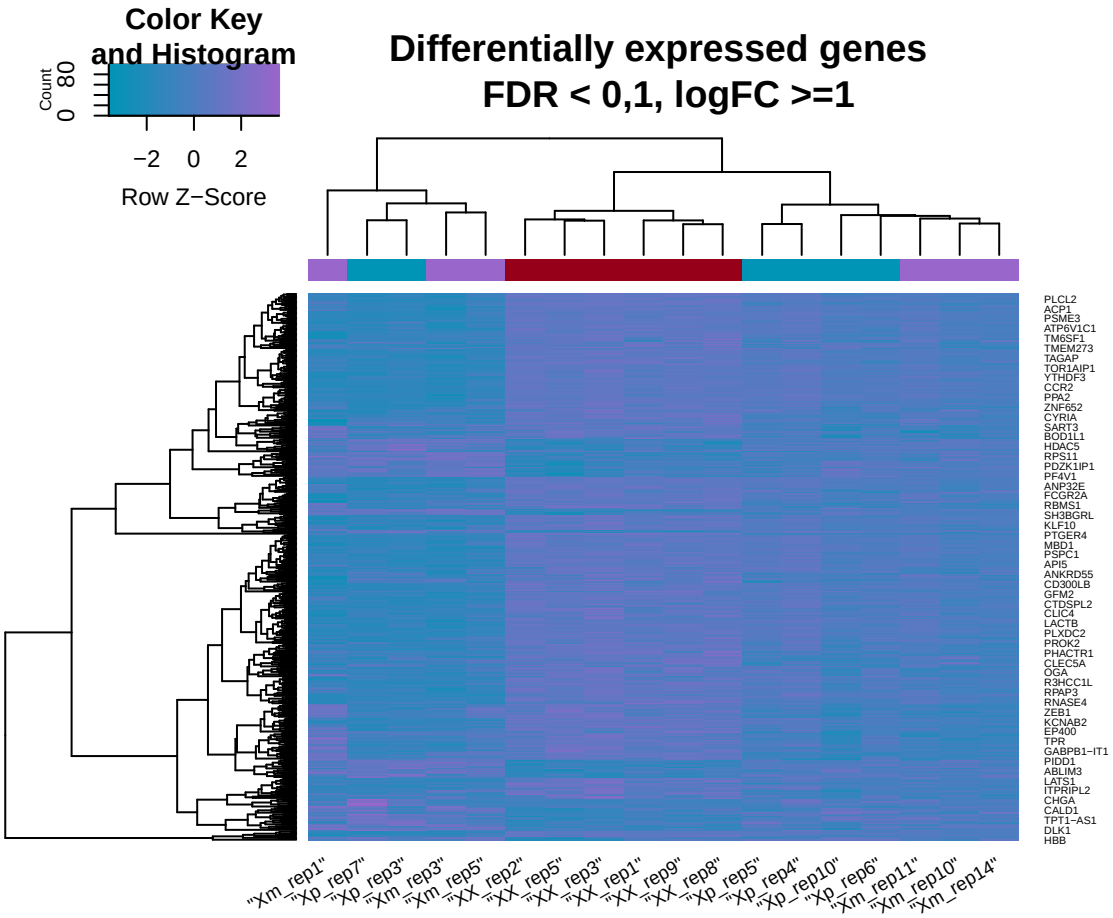
Vamos a interpretar esta tabla. Por ejemplo, para el contraste (grupo) `Xm45vsXp45`, hay 0 genes regulados aguas abajo y aguas arriba que presenten expresión diferencial. Para `Xm45vsXX46`, hay `314` regulados aguas abajo y `52` regulados aguas arriba que presentan expresión diferencial. Y para `Xp45vsXX46` hay `169` regulados aguas abajo y `47` aguas arriba con expresión diferencial.

Utilizaremos un diagrama de Venn para ver el número de genes que se expresan diferencialmente entre los distintos grupos. Son comparaciones múltiples porque tenemos tres grupos que estamos comparando. Si quisiéramos por ejemplo mirar estos genes en solo uno de los grupos, en ese caso podríamos utilizar un `Volcano plot`.



Como vemos, para el grupo `Xm45vsXp45` el número de genes en común es 0, mientras que `Xm45vsXX46` y `Xp45vsXX46` sí tienen genes en común.

También podemos hacer un *Heatmap* para ver los perfiles de expresión:



9. Análisis de significación biológica ("Gene Enrichment Analysis")

El análisis de significación biológica estudia las funciones de los genes buscando sus anotaciones en bases de datos de anotación funcional.

El primer paso es preparar la lista de genes que analizaremos:

```
## Xm45vsXp45 Xm45vsXX46 Xp45vsXX46
##          0       1984       1571
```

Como vemos, hay 0 genes para el grupo `Xm45vsXp45`, por tanto, tomaremos las otras dos columnas.

El siguiente paso es obtener los identificadores *Entrez*. Vamos a definir que nuestro universo contenga todos los genes que tengan al menos una anotación en Gene Ontology. Estamos utilizando paquetes de anotaciones de organismos, de Gene Ontology (`org.Hs.egGO`) y de rutas metabólicas (`org.Hs.egPATH`).

Así pues, ya podemos utilizar el paquete `ReactomePA` para hacer el análisis de significación biológica, el cual lo haremos sobre la segunda y tercera lista.

```
## ####################################
## Comparison:  Xm45vsXX46
##                          ID                                Description
## R-HSA-6798695 R-HSA-6798695                    Neutrophil degranulation
```

```
## R-HSA-166520   R-HSA-166520                            Signaling by NTRKs
## R-HSA-187037   R-HSA-187037                      Signaling by NTRK1 (TRKA)
## R-HSA-76002     R-HSA-76002 Platelet activation, signaling and aggregation
## R-HSA-168898   R-HSA-168898                    Toll-like Receptor Cascades
## R-HSA-4420097 R-HSA-4420097                          VEGFA-VEGFR2 Pathway
##                GeneRatio   BgRatio       pvalue     p.adjust       qvalue
## R-HSA-6798695  112/1219 477/10668 2.084890e-14 2.845875e-11 2.330688e-11
## R-HSA-166520    37/1219 134/10668 1.926326e-07 1.314718e-04 1.076715e-04
## R-HSA-187037    33/1219 115/10668 3.315664e-07 1.508627e-04 1.235521e-04
## R-HSA-76002     58/1219 263/10668 4.791707e-07 1.635170e-04 1.339156e-04
## R-HSA-168898    37/1219 153/10668 6.283031e-06 1.715268e-03 1.404754e-03
## R-HSA-4420097   27/1219  99/10668 1.089887e-05 2.479492e-03 2.030631e-03
##
## R-HSA-6798695 HP/KCNAB2/SCAMP1/CPNE1/ATP8B4/RAC1/ACTR2/IQGAP1/HSP90AB1/CD58/RAB37/CAND1/FCGR
## R-HSA-166520
## R-HSA-187037
## R-HSA-76002
## R-HSA-168898
## R-HSA-4420097
##                Count
## R-HSA-6798695   112
## R-HSA-166520     37
## R-HSA-187037     33
## R-HSA-76002      58
## R-HSA-168898     37
## R-HSA-4420097    27
## ###################################
## Comparison:  Xp45vsXX46
##                            ID
## R-HSA-6798695 R-HSA-6798695
## R-HSA-512988   R-HSA-512988
## R-HSA-373753   R-HSA-373753
## R-HSA-5663202 R-HSA-5663202
## R-HSA-2029480 R-HSA-2029480
##                                                                          Descripti
## R-HSA-6798695                                                  Neutrophil degranulati
## R-HSA-512988                                 Interleukin-3, Interleukin-5 and GM-CSF signali
## R-HSA-373753                                                    Nephrin family interactio
## R-HSA-5663202 Diseases of signal transduction by growth factor receptors and second messenge
## R-HSA-2029480                                  Fcgamma receptor (FCGR) dependent phagocytos
##                GeneRatio   BgRatio       pvalue     p.adjust       qvalue
## R-HSA-6798695    79/937 477/10668 1.757874e-08 0.0000231512 2.076142e-05
## R-HSA-512988     14/937  48/10668 4.153255e-05 0.0273491850 2.452606e-02
## R-HSA-373753      9/937  23/10668 7.881347e-05 0.0345991143 3.102762e-02
## R-HSA-5663202    56/937 387/10668 1.218924e-04 0.0349844135 3.137315e-02
## R-HSA-2029480    19/937  86/10668 1.328186e-04 0.0349844135 3.137315e-02
##
```

```
## R-HSA-6798695 CAND1/CPNE1/IQGAP1/ATP8B4/HP/TBC1D10C/RAC1/DEGS1/RAB37/SPTAN1/C3AR1/GLIPR1/AC
## R-HSA-512988
## R-HSA-373753
## R-HSA-5663202
## R-HSA-2029480
##                Count
## R-HSA-6798695    79
## R-HSA-512988     14
## R-HSA-373753      9
## R-HSA-5663202    56
## R-HSA-2029480    19
```

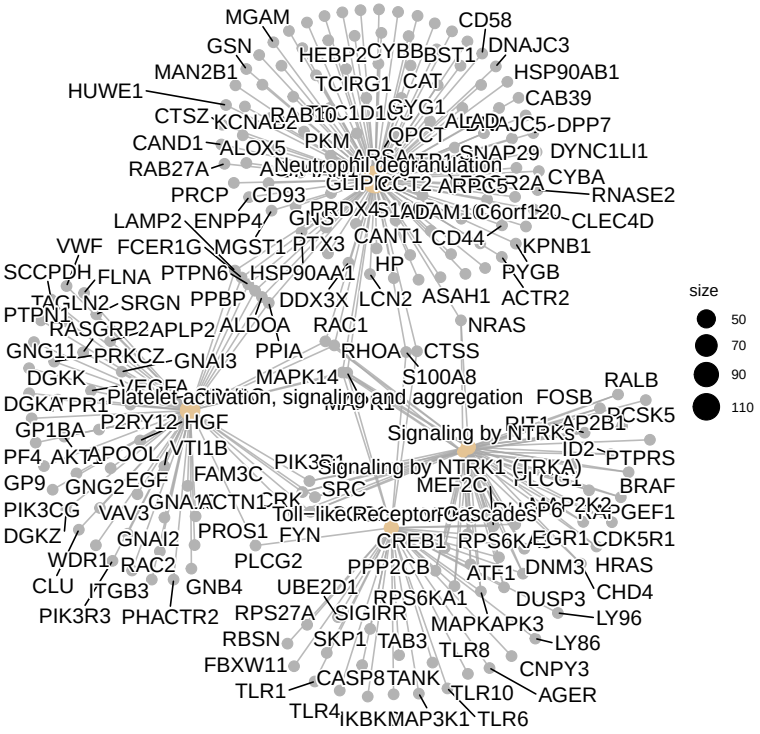


Fig.3: Red obtenida del análisis de enriquecimiento de Reactome en la lista obtenida de la comparación entre 45Xm y 46XX

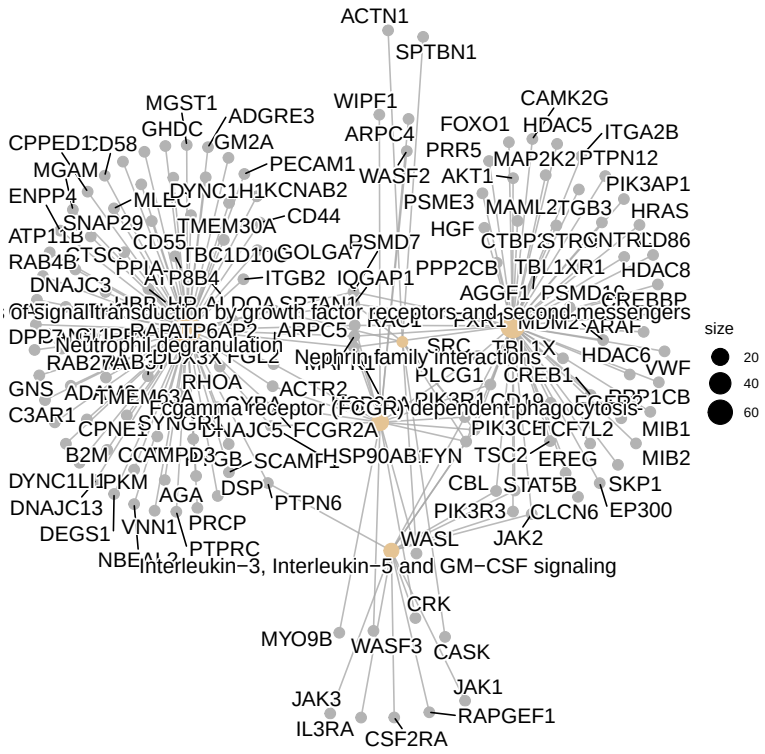


Fig.4: Red obtenida del análisis de enriquecimiento de Reactome en la lista obtenida de la comparación entre 45Xp y 46XX

Discusión

Finalmente, haremos una breve discusión sobre los resultados que hemos ido mostrando anteriormente.

Tras haber normalizado y filtrado nuestros datos, nos quedamos con un total de `5040` genes. Después, hicimos un modelo lineal con `limma` para identificar aquellos genes que estaban diferencialmente expresados (DEGs), quedándonos con los que mayores valores tenían. El siguiente paso fue anotar los resultados y comparar entre grupos. Esto, como ya comentamos, nos daba una tabla donde había 0 genes regulados aguas abajo y aguas arriba que presentaran expresión diferencial para `Xm45vsXp45`. A continuación, el diagrama de Venn mostraba unos grupos acordes con los resultados obtenidos en la tabla para la expresión diferencial. También realizamos un Heatmap para ver esos DEGs pero de forma más general.

Finalmente, para la parte del análisis de significación biológica, que podría decirse, es el que más nos interesa, no pudimos comparar los genes de `Xm` y `Xp` ya que la lista para este grupo es `0`, pero sí pudimos compararlos con el cariotipo normal `XX`. Además, la red obtenida del análisis de enriquecimiento es altamente compleja, a pesar de que solo mostramos 2 categorías en esta (es decir, el número de términos enriquecidos que queremos mostrar).

Por tanto, esto podría sugerir, que si bien existe diferencia entre los cariotipos `Xp` y `Xm` con el normal `XX` (tiene sentido, porque está faltando un cromosoma X entero), ya que como hemos visto en nuestros análisis, tenemos DEGs, entre los dos primeros cariotipos `Xp` y `Xm` parece que la diferencia no es significativa, al menos con los datos que hemos utilizado.

Apéndice

A continuación, se muestra el **código** empleado.

Selección de muestras

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

selectSamples<- function (myID){
set.seed(myID)
selected <- c(sample(1:10, 6),11, sample(12:26, 5), sample(27:36,6))
selected <- sort(selected)
}

mySelected <- selectSamples(77146010)
targetsAll <-read.csv(file="targetsAll.csv", row.names = 1, head=TRUE)
myTargets <- targetsAll[mySelected,]

write.csv(myTargets, file = "./data/myTargets.csv")

library(GEOquery)
filePaths = getGEOSuppFiles("GSE46687")

library(oligo)
celFiles <- list.celfiles("./data2", full.names = TRUE)

library(Biobase)
my.targets <-read.AnnotatedDataFrame(file.path("./data2","myTargets.csv"),
                                     header = TRUE, row.names = 1, sep=",")

rawData <- read.celfiles(celFiles, phenoData = my.targets)
```

1. **Identificar qué grupos hay y a qué grupo pertenece cada muestra.**

```
colnames(rawData) <- my.targets@data$X.title.
head(rawData)
```

2. **Exploración y control de calidad de los datos crudos**

```
library(arrayQualityMetrics)
arrayQualityMetrics(rawData)

library(ggplot2)
library(ggrepel)
```

```
plotPCA3 <- function (datos, labels, factor, title, scale,colores, size = 1.5,
                      glineas = 0.25) {
   data <- prcomp(t(datos),scale=scale)
   # plot adjustments
   dataDf <- data.frame(data$x)
   Group <- factor
   loads <- round(data$sdev^2/sum(data$sdev^2)*100,1)
   # main plot
   p1 <- ggplot(dataDf,aes(x=PC1, y=PC2)) +
     theme_classic() +
     geom_hline(yintercept = 0, color = "gray70") +
     geom_vline(xintercept = 0, color = "gray70") +
     geom_point(aes(color = Group), alpha = 0.55, size = 3) +
     coord_cartesian(xlim = c(min(data$x[,1])-5,max(data$x[,1])+5)) +
     scale_fill_discrete(name = "Group")
   # avoiding labels superposition
   p1 + geom_text_repel(aes(y = PC2 + 0.25, label = labels),segment.size = 0.25,
                        size = size) +
     labs(x = c(paste("PC1",loads[1],"%")),y=c(paste("PC2",loads[2],"%"))) +
     ggtitle(paste("Principal Component Analysis for: ",title,sep=" "))+
     theme(plot.title = element_text(hjust = 0.5)) +
     scale_color_manual(values=colores)
}


plotPCA3(exprs(rawData), labels = myTargets$title, factor = myTargets$karyotype,
          title="Raw data", scale = FALSE, size = 3,
          colores = c("#9966CC", "#0095B6", "#960018"))
```

```
boxplot(rawData, cex.axis=0.5, las=2,  which="all",
        col = c(rep("#960018", 6), rep("#9966CC", 6), rep("#0095B6", 6)),
        main="Distribution of raw intensity values")
```

3. Normalización

```
eset_rma <- rma(rawData)
```

4. [Control de calidad de los datos normalizados] (opcional)

```
arrayQualityMetrics(eset_rma, outdir = file.path("./results", "QCDir.Norm"),
                    force=TRUE)

plotPCA3(exprs(eset_rma), labels = myTargets$title, factor = myTargets$karyotype,
          title="Normalized data", scale = FALSE, size = 3,
          colores = c("#9966CC", "#0095B6", "#960018", "yellow"))
```

```
boxplot(eset_rma, cex.axis=0.5, las=2,  which="all",
          col = c(rep("#960018", 6), rep("#9966CC", 6), rep("#0095B6", 6)),
          main="Boxplot for arrays intensity: Normalized Data")
```

5. Filtraje no específico [opcional]

```
library(genefilter)
# BiocManager::install("hgu133plus2.db")
library(hgu133plus2.db)
annotation(eset_rma) <- "hgu133plus2.db"
filtered <- nsFilter(eset_rma,
                     require.entrez = TRUE, remove.dupEntrez = TRUE,
                     var.filter=TRUE, var.func=IQR, var.cutoff=0.75,
                     filterByQuantile=TRUE, feature.exclude = "^AFFX")

print(filtered$filter.log)

eset_filtered <-filtered$eset
eset_filtered

# Guardemos los datos en este punto.
write.csv(exprs(eset_rma), file="./results/normalized.Data.csv")
write.csv(exprs(eset_filtered), file="./results/normalized.Filtered.Data.csv")
save(eset_rma, eset_filtered, file="./results/normalized.Data.Rda")
```

6. Identificación de genes diferencialmente expresados

```
if (!exists("eset_filtered")) load (file="./results/normalized.Data.Rda")
library(limma)
designMat<- model.matrix(~0+X.karyotype., pData(eset_filtered))
colnames(designMat) <- c("Xm45", "Xp45","XX46")
print(designMat)

cont.matrix <- makeContrasts (Xm45vsXp45 = Xm45-Xp45,
                              Xm45vsXX46 = Xm45-XX46,
                              Xp45vsXX46 = Xp45-XX46,
                              levels=designMat)
print(cont.matrix)

library(limma)
fit<-lmFit(eset_filtered, designMat)
fit.main<-contrasts.fit(fit, cont.matrix)
fit.main<-eBayes(fit.main)
class(fit.main)
```

```
topTab.Xm45vsXp45 <- topTable(fit.main, number=nrow(fit.main), coef="Xm45vsXp45",
                              adjust="fdr")
topTab.Xm45vsXX46 <- topTable(fit.main, number=nrow(fit.main), coef="Xm45vsXX46",
                              adjust="fdr")
topTab.Xp45vsXX46 <- topTable(fit.main, number=nrow(fit.main), coef="Xp45vsXX46",
                              adjust="fdr")
```

7. Anotación de los resultados

```
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)
}

topAnnotated.Xm45vsXp45 <- annotatedTopTable(topTab.Xm45vsXp45,
anotPackage="hgu133plus2.db")
topAnnotated.Xm45vsXX46 <- annotatedTopTable(topTab.Xm45vsXX46,
anotPackage="hgu133plus2.db")
topAnnotated.Xp45vsXX46 <- annotatedTopTable(topTab.Xp45vsXX46,
anotPackage="hgu133plus2.db")

write.csv(topAnnotated.Xm45vsXp45, file="./results/topAnnotated.Xm45vsXp45.csv")
write.csv(topAnnotated.Xm45vsXX46, file="./results/topAnnotated.Xm45vsXX46.csv")
write.csv(topAnnotated.Xp45vsXX46, file="./results/topAnnotated.Xp45vsXX46.csv")
```

8. Comparación entre distintas comparaciones (si hay más de una comparación, ver qué genes han sido seleccionados en más de una comparación)

```
library(limma)
res<-decideTests(fit.main, method="separate", adjust.method="fdr", p.value=0.1, lfc=1)
sum.res.rows<-apply(abs(res),1,sum)
res.selected<-res[sum.res.rows!=0,]
print(summary(res))

vennDiagram(res.selected, cex=0.9)
title("Genes in common between the three comparisons\n
      Genes selected with FDR < 0.1 and logFC > 1")

probesInHeatmap <- rownames(res.selected)
HMdata <- exprs(eset_filtered)[rownames(exprs(eset_filtered)) %in% probesInHeatmap,]
library(tidyselect)
geneSymbols <- select(hgu133plus2.db, rownames(HMdata), c("SYMBOL"))
```

```
SYMBOLS<- geneSymbols$SYMBOL
rownames(HMdata) <- SYMBOLS
write.csv(HMdata, file = file.path("./results/data4Heatmap.csv"))

my_palette <- colorRampPalette(c("#0095B6", "#9966CC"))(n = 299)
library(gplots)

heatmap.2(HMdata,
          Rowv = TRUE,
          Colv = TRUE,
          dendrogram = "both",
          main = "Differentially expressed genes \n FDR < 0,1, logFC >=1",
          scale = "row",
          col = my_palette,
          sepcolor = "white",
          sepwidth = c(0.05,0.05),
          cexRow = 0.5,
          cexCol = 0.9,
          key = TRUE,
          keysize = 1.5,
          density.info = "histogram",
          ColSideColors = c(rep("#960018",6),rep("#9966CC",6), rep("#0095B6",6)),
          tracecol = NULL,
          srtCol = 30)
```

9. Análisis de significación biológica ("Gene Enrichment Analysis")

```
listOfTables <- list(Xm45vsXp45 = topTab.Xm45vsXp45,
                     Xm45vsXX46  = topTab.Xm45vsXX46,
                     Xp45vsXX46 = topTab.Xp45vsXX46)

listOfSelected <- list()

for (i in 1:length(listOfTables)){
   # select the toptable
   topTab <- listOfTables[[i]]
   # select the genes to be included in the analysis
   whichGenes<-topTab["adj.P.Val"]<0.15
   selectedIDs <- rownames(topTab)[whichGenes]
   # convert the ID to Entrez
   EntrezIDs<- select(hgu133plus2.db, selectedIDs, c("ENTREZID"))
   EntrezIDs <- EntrezIDs$ENTREZID
   listOfSelected[[i]] <- EntrezIDs
   names(listOfSelected)[i] <- names(listOfTables)[i]
}

sapply(listOfSelected, length)
```

```
mapped_genes2GO <- mappedkeys(org.Hs.egGO)
mapped_genes2KEGG <- mappedkeys(org.Hs.egPATH)
mapped_genes <- union(mapped_genes2GO , mapped_genes2KEGG)

library(ReactomePA)

listOfData <- listOfSelected[2:3]
comparisonsNames <- names(listOfData)
universe <- mapped_genes

for (i in 1:length(listOfData)){
   genesIn <- listOfData[[i]]
   comparison <- comparisonsNames[i]
   enrich.result <- enrichPathway(gene = genesIn,
                                  pvalueCutoff = 0.05,
                                  readable = T,
                                  pAdjustMethod = "BH",
                                  organism = "human",
                                  universe = universe)

   cat("####################################")
   cat("\nComparison: ", comparison,"\n")
   print(head(enrich.result))

   if (length(rownames(enrich.result@result)) != 0) {
   write.csv(as.data.frame(enrich.result),
             file =paste0("./results/","ReactomePA.Results.",comparison,".csv"),
             row.names = FALSE)

   pdf(file=paste0("./results/","ReactomePABarplot.",comparison,".pdf"))
     print(barplot(enrich.result, showCategory = 15, font.size = 4,
             title = paste0("Reactome Pathway Analysis for ", comparison,
                            ". Barplot")))
   dev.off()

   pdf(file = paste0("./results/","ReactomePAcnetplot.",comparison,".pdf"))
     print(cnetplot(enrich.result, categorySize = "geneNum", schowCategory = 15,
          vertex.label.cex = 0.75))
   dev.off()
   }
 }
```