**Guía para la realización de análisis de enriquecimiento funcional en R (parte uno).**

**Simposio anual Ómicas - Modalidad talleres**

1. **Análisis de enriquecimiento funcional para una lista de genes única (Instalación y carga de archivos)**

En esta sesión se realizará análisis de enriquecimiento funcional para una única lista de genes usando R.

* El archivo para usar es el mismo que el que se ha usado en la sesión anterior. Este se encuentra disponible en el siguiente enlace: <https://raw.githubusercontent.com/ccsosa/R_Examples/master/GIM.csv>, sin embargo, se llamará directamente desde R.
* Para esta sesión necesitará instalar las siguientes librerías
  1. **gprofiler2**
  2. **biomaRt**
  3. **topGO**
  4. **GOsummaries**
  5. **Enrichplot**
  6. **org.Hs.eg.db**

if (!requireNamespace("BiocManager", quietly = TRUE))

install.packages("BiocManager")

BiocManager::install("topGO")

BiocManager::install("biomaRt")

BiocManager::install("org.Hs.eg.db")

BiocManager::install("enrichplot")

install.packages("gprofiler2")

install.packages("ggnewscale")

* Para este ejercicio, el script con las instrucciones se puede descargar del enlace: <https://raw.githubusercontent.com/ccsosa/TALLER_OMICAS/master/Taller_8_GO_1.R> y puede descargarse localmente dando clic derecho y luego usando la opción Guardar como. Posteriormente guarde el archivo como "Taller\_8\_GO\_1.R" entre comillas.Interfaz de usuario gráfica, Texto, Aplicación

  Descripción generada automáticamente
* Abra el archivo Taller\_8\_GO\_1.R usando el icono

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

* Al abrir el archivo con extensión .R en RStudio, este será abierto en el área del editor, como se ve a continuación:

Captura de pantalla de computadora

Descripción generada automáticamente

* Para correr los análisis propuestos, primero se deben correr las líneas 2 y 3, en las cuales se cargarán las librerías gprofiler2, biomaRt, topGO, clusterProfiler y GOsummaries.

library (gprofiler2);library(biomaRt);library(topGO),

library (clusterProfiler); library (GOsummaries)

* Posteriormente se cargará el archivo con la lista de genes pertenecientes a inestabilidad genómica en humano que se encuentra localizado en GitHub (líneas 5 a la 7)

#cargar ejemplo desde CSV

url\_file = "https://raw.githubusercontent.com/ccsosa/R\_Examples/master/GIM.csv"

x <- read.csv(url\_file,header = T)

1. **Análisis de enriquecimiento funcional para una lista de genes única**
2. **gprofiler2**

* Ahora que ha cargado el archivo con la lista de genes, haremos un análisis de enriquecimiento funcional con el paquete gprofiler2, usaremos como organism= “hsapiens”, un punto de corte de 0.05, como método de corrección false\_discovery\_rate. Este análisis se guardará en el objeto x\_s y se encuentra entre las líneas 13 a 21 del script

#Analisis de enriquecimiento funcional

x\_s <- gprofiler2::gost(query = x[,1],

organism = "hsapiens", ordered\_query = FALSE,

multi\_query = FALSE, significant = TRUE, exclude\_iea = FALSE,

measure\_underrepresentation = FALSE, evcodes = FALSE,

user\_threshold = 0.05, correction\_method = "false\_discovery\_rate",

domain\_scope = "annotated", custom\_bg = NULL,

numeric\_ns = "", as\_short\_link = FALSE,

sources="GO:BP")

**NOTA**: Recuerde que para correr esta línea de código debe señarla y dar clic en Run, como se muestra abajo:

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

* Ahora haremos un *Manhattan plot,* en este grafico se usa el para ver fácilmente que términos GO tuvieron un valor de p más cercano a 0 de manera más fácil (líneas 22 a la 24)

#GO PLOT

p <- gprofiler2::gostplot(x\_s, capped = F, interactive = FALSE)

p

* El resultado del *Manhattan plot,* debe verse así, si quiere un grafico interactive, por favor cambie interactive=FALSE a interactive=TRUE.

Gráfico, Gráfico de dispersión

Descripción generada automáticamente

* Si desea un gráfico en el que solo se muestre el top diez de GO enriquecidos, corra las líneas 27 a 29, como se muestra a continuación. El grafico resultante mostrará que el termino GO más enriquecido fue respuesta fue respuesta celular a daño de ADN

#GO PLOT con el top diez

gprofiler2::publish\_gostplot(p, highlight\_terms = x\_s$result$term\_id[1:10])

p

Gráfico, Gráfico de dispersión

Descripción generada automáticamente

1. **topGO**

* Para correr un análisis de enriquecimiento funcional en topGO, primero necesitará obtener los términos GO desde ENSEMBL, para esto deberá usar las líneas 35 a la 40 en donde se usará la librería biomaRt para obtener esa información. La primera parte del código permitirá establecer una conexión con el servidor de ENSEMBL. Ya que usaremos la información de humano, en el parámetro biomart, estará la opción 'ENSEMBL\_MART\_ENSEMBL' y en el parámetro dataset usaremos la opción 'hsapiens\_gene\_ensembl'.

**NOTA**: Si desea hacer extraer la información en plantas, debe usar la siguiente línea de código en azul y cambie el dataset dependiendo de que especie guste usar.

db= biomaRt::useMart(biomart="plants\_mart",host="plants.ensembl.org", dataset= "athaliana\_eg\_gene")

#Obtener los GO desde ENSEMBL

db= biomaRt::useMart('ENSEMBL\_MART\_ENSEMBL',dataset='hsapiens\_gene\_ensembl', host="www.ensembl.org")

go\_ids= biomaRt::getBM(attributes=c('go\_id', 'external\_gene\_name', 'namespace\_1003'),

filters='external\_gene\_name',

values=x[,1],

mart=db)

**Nota**: Si en su computador surge un error a la hora de obtener la información de los GO asociados a la lista de genes, esto se debe a las restricciones que posee el muro de fuego de la universidad, por favor corra la siguiente línea de código (línea 42 a la 44) para corregirlo.

go\_ids <- read.csv("https://raw.githubusercontent.com/ccsosa/TALLER\_OMICAS/master/go\_ids.csv",header = T)

* Corra la línea 46 para crear una lista de los términos GO por cada gen

gene\_2\_GO=unstack(go\_ids[,c(1,2)])

* Las líneas 49 a la 54 le permitirán remover genes que no tengan anotaciones y marcar genes de interés como 1 en una lista

#Remover genes sin anotacion

keep = x[,1] %in% go\_ids[,2]

keep =which(keep==TRUE)

candidate\_list=x[,1][keep]

geneList=factor(as.integer(x[,1] %in% candidate\_list),levels = c(0,1))

names(geneList)= x[,1]

* Posteriormente las líneas 56 a la 59 le permitirán crear un objeto, donde se obtendrán procesos biológicos mediante el parámetro ontology='BP'

#crear un objeto topGOdata

GOdata=new('topGOdata', ontology='BP', allGenes = geneList, annot = topGO::annFUN.gene2GO, gene2GO = gene\_2\_GO)

* Las líneas 63 a 72 le permitirán calcular las pruebas estadísticas de Fisher, Kosmogorov y eliminación combinada con Kosmogorov respectivamente y reportar los diez nodos más relevantes en una tabla resumen que puede llamar usando View(allRes)

#Run stastical tests

resultFisher <- runTest(GOdata, algorithm = "classic", statistic = "fisher")

resultKS <- runTest(GOdata, algorithm = "classic", statistic = "ks")

resultKS.elim <- runTest(GOdata, algorithm = "elim", statistic = "ks")

#summarize in a table

allRes <- topGO::GenTable(GOdata, classicFisher = resultFisher,

classicKS = resultKS, elimKS = resultKS.elim,

orderBy = "elimKS", ranksOf = "classicFisher", topNodes = 10)

Una captura de pantalla de un videojuego

Descripción generada automáticamente con confianza media

* Si desea obtener los valores p asociados a las pruebas estadísticas realizadas podrá usar la función score, a su vez, podrá ver los resultados de todos los términos GO mediante la función termStat (líneas 75 a 78).

#diversas formas de obtener p valores

pValue.classic <- score(resultKS)

pValue.elim <- score(resultKS.elim)[names(pValue.classic)]

gstat <- topGO::termStat(GOdata, names(pValue.classic))

* El paquete topGO, le permitirá obtener un grafo dirigido con los nodos más significantes, en este caso usaremos tres (líneas 80 y 81).

par(cex = 1)

showSigOfNodes(GOdata, score(resultKS.elim), firstSigNodes = 3)

Diagrama

Descripción generada automáticamente

1. **GOsummaries**

* Una última opción para análisis de enriquecimiento funcional es presentada a través del paquete GOSummaries. Para esta opción usaremos los datos originales que se cargaron en el CSV y usaremos la función gosummaries (líneas 88-92).

#alternativa usando gosummaries

gl = list(GIM = x[,1]) # Two lists per component

gs = GOsummaries::gosummaries(gl)

plot(gs, fontsize = 8)

Interfaz de usuario gráfica, Aplicación, Tabla

Descripción generada automáticamente