

PEC1

Jesús Martínez López

5 de noviembre, 2024

Contents

Introducción	1
Objetivos	1
Materiales y Métodos	2
Materiales	2
Métodos	2
Resultados	3
1. Selección del dataset	3
2. Contenedor del tipo SummarizedExperiment	4
3. Exploración de los datos	8
Discusión	19
Referencias	20
Anexos	21
Anexo I - Código R	21
Anexo II - Imágenes	30

Introducción

La presente PEC está diseñada para consolidar los conocimientos adquiridos sobre las tecnologías ómicas y las herramientas de análisis de datos vistos hasta ahora en el curso. En particular, se centrará en el uso de **Bioconductor** y técnicas de exploración de datos.

El ejercicio propuesto implica seleccionar en primer lugar un conjunto de datos de metabolómica para realizar un análisis simplificado del mismo. Para ello, es fundamental familiarizarse previamente con aspectos clave como las tecnologías ómicas, la gestión de datos en **Bioconductor** y **GitHub**, y los contenedores de datos ómicos como **SummarizedExperiment**.

A través de esta PEC tendremos la oportunidad de planificar y ejecutar un proceso de análisis de datos ómicos que incluye la descarga de datos, la creación de un contenedor adecuado, y la exploración y documentación de los hallazgos. El objetivo final es elaborar un informe detallado que describa cada paso del proceso, así como presentar los resultados en un repositorio de **GitHub**.

Objetivos

El objetivo principal de este trabajo es realizar un **análisis exploratorio** de datos provenientes de estudios de metabolómica, utilizando herramientas y metodologías adecuadas para extraer información relevante de los datos seleccionados. Los objetivos específicos incluyen:

1. **Familiarización con las tecnologías ómicas:** comprender los principios y aplicaciones de las diferentes tecnologías utilizadas en metabolómica.
2. **Descarga y manejo de datos:** seleccionar un conjunto de datos de metabolómica de repositorios confiables, asegurando la correcta adquisición de los datos y su formato.

3. **Creación de contenedores de datos:** implementar un contenedor del tipo `SummarizedExperiment` que permita almacenar tanto los datos como los metadatos de manera estructurada, facilitando su posterior análisis.
4. **Exploración de datos:** realizar un análisis exploratorio de los datos mediante técnicas estadísticas y visualizaciones, que ofrezcan una visión general de las características y tendencias en los datos.
5. **Documentación y reporte:** elaborar un informe detallado que describa cada etapa del proceso de análisis, incluyendo la metodología, resultados y reflexiones, así como la reposición del trabajo en un repositorio de `GitHub` para promover la transparencia y reproducibilidad en la investigación.
6. **Desarrollo de habilidades técnicas:** fortalecer las competencias en el uso de herramientas de análisis de datos ómicos, como `Bioconductor` y `GitHub`, para la gestión y análisis de datos en el ámbito de la metabolómica.

Materiales y Métodos

Materiales

El conjunto de datos para la realización de esta PEC se ha escogido al azar a partir del archivo `Data_Catalog.xlsx` del repositorio de `github`: <https://github.com/nutrimetabolomics/metaboData/>

En concreto, se trata del dataset `2023-CIMCBTutorial`, constituido por 149 *features* (metabolitos) para 140 muestras. Este conjunto de datos metabolómicos se basa en el artículo de Chan et al. (2016), publicado en el *British Journal of Cancer*, en el que se analizaron muestras de orina de 43 pacientes con carcinoma gástrico, 40 con enfermedades gástricas benignas y 40 pacientes sanos mediante espectroscopia de resonancia magnética nuclear ((^1H)-NMR) con el objetivo de determinar un perfil metabolómico urinario distintivo en pacientes con carcinoma gástrico con potencial clínico para el diagnóstico temprano.

Los datos descargados directamente del repositorio de `GitHub` se procesaron para crear una estructura de datos de tipo `SummarizedExperiment`. Se amplió la información de los metadatos de muestras a través de un proceso de *web scraping*.

Métodos

Diferenciaremos dos grupos de métodos:

- **Métodos Bioinformáticos:** el principal método bioinformático utilizado fue la creación de un contenedor de datos mediante el paquete `SummarizedExperiment` de `Bioconductor`. Este enfoque permite organizar y almacenar tanto los datos de metabolitos como los metadatos asociados a las muestras de forma estructurada, facilitando el posterior análisis exploratorio. Para este último, nos apoyaremos en los métodos contenidos en el paquete `POMA`, que en su versión más reciente utiliza objetos de tipo `SummarizedExperiment` como argumento en sus funciones. Además de R base, otros paquetes de R y aplicaciones utilizados fueron:
 1. `readxl` para la lectura de datos
 2. `knitr` y `kableExtra` para elaboración del informe y de tablas de datos
 3. `plotly`, `ggplot2`, `ggraph` y `pactchwork` dependencias de `POMA` y para elaboración de gráficos
 4. `rvest` para web scraping
 5. `GitHub` para la creación del repositorio
- **Métodos Estadísticos:**
 1. **Análisis Univariado:** se elaboraron histogramas y boxplots para examinar la distribución general de los metabolitos. Se realizaron resúmenes estadísticos generales de los datos. Se comprobó la significación estadística y biológica de las concentraciones de los distintos metabolitos para las categorías GC con respecto a HE con el test de Welch y gráficamente mediante Volcano plot a partir de los datos normalizados, respectivamente.

2. Análisis Multivariante: se realizó un Análisis de Componentes Principales (PCA) y Agrupamiento Jerárquico para determinar si las categorías clínicas de las muestras se relacionaban con las fuentes de variabilidad del estudio (metabolitos).

Resultados

1. Selección del dataset

En primer lugar necesitamos obtener una copia del archivo `Data_Catalog.xlsx` del repositorio para poder explorarlo y seleccionar el estudio en cuestión. Si queremos hacerlo desde R, necesitaremos el enlace `RAW` para este archivo. Haciendo click sobre el archivo (Figura 3 del Anexo II) podemos copiar el enlace RAW haciendo click derecho del ratón en el botón con este mismo nombre (Figura 4 del Anexo II).

Una vez obtenido el enlace, lo usaremos en el siguiente script para descargar el archivo y leerlo:

```
# URL del archivo Data_Catalog.xlsx en GitHub
github.url <- "https://github.com/nutrimetabolomics/metaboData/raw/refs/heads/main"
file <- "Data_Catalog.xlsx"

# URL completa usando file.path()
datasets_catalog.url <- file.path(github.url, file)

# descargamos el archivo
download.file(datasets_catalog.url, destfile = "Data_Catalog.xlsx", mode = "wb")

# leemos el archivo .xlsx descargado
data <- read_excel("Data_Catalog.xlsx", sheet = 1)

# mostramos el contenido
kable(data[,1:ncol(data)-1]) # sin mostrar la descripción, última columna
```

Dataset	Samples	Features
2018-MetabotypingPaper	39	690
2018-Phosphoproteomics	12	1320
2023-CIMCBTutorial	140	149
2023-UGrX-4MetaboAnalystTutorial	24	145
2024-fobitools-UseCase_1	45	1541
2024-Cachexia	77	63

De esta forma podemos elegir un dataset al azar fijando una semilla aleatoria para reproducibilidad:

```
set.seed(123) # semilla aleatoria
selection <- sample(1:nrow(data), 1)

kable(data[selection, 1:ncol(data)-1])
```

Dataset	Samples	Features
2023-CIMCBTutorial	140	149

```
# descripción del estudio
cat(data[selection, ncol(data)]$Description)
```

```
## NMR data from a gastric cancer study used in a metabolomics data analysis tutorial ("Basic Metabolomics")
```

La propia descripción nos indica que son los datos de espectrometría de Resonancia Magnética Nuclear (NMR) en un estudio de cáncer gástrico utilizado para un tutorial de análisis metabolómico cuyo repositorio en Github se puede acceder mediante este enlace: <https://cimcb.github.io/MetabWorkflowTutorial/Tutorial1.html>.

Puesto que en la Introducción del repositorio de Github se indica que la estructura del mismo está basado en carpetas, es decir, una carpeta **Datasets** con directorios nombrados como en la columna **Dataset** del archivo **Data_Catalog.xlsx**, y dentro de cada uno de ellos, al menos, el conjunto de datos y un archivo **description.md** (Figura 5 del Anexo II).

Podemos obtener más información sobre este conjunto de datos en la descripción del estudio si leemos el archivo **description.md**:

```
## Dataset used in the CIMBC tutorial on ["Basic Metabolomics Data Analysis Workflow"](https://cimcb.github.io/MetabWorkflowTutorial/Tutorial1.html)
##
## The tutorial describes the data as follows:
##
## - The study used in this tutorial has been previously published as an open access article Chan et al. (2016)
##
## - The deconvolved and annotated data file have been deposited at the Metabolomics Workbench data repository
##
## - The data can be accessed directly via its project DOI:10.21228/M8B10B
##
## - 1H-NMR spectra were acquired at Canada's National High Field Nuclear Magnetic Resonance Centre (NANUC)
##
## - Spectral deconvolution and metabolite annotation was performed using the Chenomx NMR Suite v7.6.
##
## **Unfortunately, the Raw NMR data is unavailable.**
```

La descripción nos dice que el tutorial se basa en un estudio publicado por Chan et al. (2016) en el British Journal of Cancer. Los datos procesados y anotados fueron depositados en el repositorio Metabolomics Workbench (ID del proyecto PR000699) y son accesibles mediante el DOI: 10.21228/M8B10B. Los espectros (1)H-NMR fueron adquiridas en el Centro Nacional de Resonancia Magnética Nuclear de Alta Frecuencia de Canadá (NANUC), utilizando un espectrómetro Varian Inova de 600 MHz. El análisis espectral y la anotación de metabolitos se realizaron con la suite Chenomx NMR v7.6. Sin embargo, los datos crudos de NMR no están disponibles.

Dentro de esta misma carpeta en el repositorio nos encontramos con el dataset nombrado como **GastricCancer_NMR.xlsx**. Procedemos así a su descarga para el cumplimiento de los objetivos de la PEC:

```
file <- "GastricCancer_NMR.xlsx"

# URL completa usando file.path()
dataset.url <- file.path(dataset.folder.url, file)

# descargamos el archivo
download.file(dataset.url, destfile = "GastricCancer_NMR.xlsx", mode = "wb")
```

No hay que olvidar que el propio dataset se encuentra accesible para su descarga en el repositorio de *Metabolomics Workbench* accediendo al Project ID PR000699, en donde obtenemos el siguiente enlace para descargar: <https://www.metabolomicsworkbench.org/studydownload/ST001047.zip>

2. Contenedor del tipo **SummarizedExperiment**

Lo primero que vamos a hacer es determinar la estructura del archivo descargado, de cuántas hojas disponemos y cuáles son sus nombres:

```
## Número de hojas: 2
```

```
## Nombres de las hojas: Data Peak
```

Tenemos entonces 2 hojas en el archivo de Excel, con los nombres Data y Peak. La estructura de cada una de las hojas es la siguiente:

```
## Hoja de Data: Idx SampleID SampleType Class M1 M2 M3 M4 M5 M6 M7 M8 M9 M10 M11 M12 M13 M14 M15 M16 M17 M18 M19 M20 M21 M22 M23 M24 M25 M26 M27 M28 M29 M30 M31 M32 M33 M34 M35 M36 M37 M38 M39 M40 M41 M42 M43 M44 M45 M46 M47 M48 M49 M50 M51 M52 M53 M54 M55 M56 M57 M58 M59 M60 M61 M62 M63 M64 M65 M66 M67 M68 M69 M70 M71 M72 M73 M74 M75 M76 M77 M78 M79 M80 M81 M82 M83 M84 M85 M86 M87 M88 M89 M90 M91 M92 M93 M94 M95 M96 M97 M98 M99 M100 M101 M102 M103 M104 M105 M106 M107 M108 M109 M110 M111 M112 M113 M114 M115 M116 M117 M118 M119 M120 M121 M122 M123 M124 M125 M126 M127 M128 M129 M130 M131 M132 M133 M134 M135 M136 M137 M138 M139 M140 M141 M142 M143 M144 M145 M146 M147 M148 M149 M150 M151 M152 M153 M154 M155 M156 M157 M158 M159 M160 M161 M162 M163 M164 M165 M166 M167 M168 M169 M170 M171 M172 M173 M174 M175 M176 M177 M178 M179 M180 M181 M182 M183 M184 M185 M186 M187 M188 M189 M190 M191 M192 M193 M194 M195 M196 M197 M198 M199 M200 M201 M202 M203 M204 M205 M206 M207 M208 M209 M210 M211 M212 M213 M214 M215 M216 M217 M218 M219 M220 M221 M222 M223 M224 M225 M226 M227 M228 M229 M230 M231 M232 M233 M234 M235 M236 M237 M238 M239 M240 M241 M242 M243 M244 M245 M246 M247 M248 M249 M250 M251 M252 M253 M254 M255 M256 M257 M258 M259 M260 M261 M262 M263 M264 M265 M266 M267 M268 M269 M270 M271 M272 M273 M274 M275 M276 M277 M278 M279 M280 M281 M282 M283 M284 M285 M286 M287 M288 M289 M290 M291 M292 M293 M294 M295 M296 M297 M298 M299 M300 M301 M302 M303 M304 M305 M306 M307 M308 M309 M310 M311 M312 M313 M314 M315 M316 M317 M318 M319 M320 M321 M322 M323 M324 M325 M326 M327 M328 M329 M330 M331 M332 M333 M334 M335 M336 M337 M338 M339 M340 M341 M342 M343 M344 M345 M346 M347 M348 M349 M350 M351 M352 M353 M354 M355 M356 M357 M358 M359 M360 M361 M362 M363 M364 M365 M366 M367 M368 M369 M370 M371 M372 M373 M374 M375 M376 M377 M378 M379 M380 M381 M382 M383 M384 M385 M386 M387 M388 M389 M390 M391 M392 M393 M394 M395 M396 M397 M398 M399 M400 M401 M402 M403 M404 M405 M406 M407 M408 M409 M410 M411 M412 M413 M414 M415 M416 M417 M418 M419 M420 M421 M422 M423 M424 M425 M426 M427 M428 M429 M430 M431 M432 M433 M434 M435 M436 M437 M438 M439 M440 M441 M442 M443 M444 M445 M446 M447 M448 M449 M450 M451 M452 M453 M454 M455 M456 M457 M458 M459 M460 M461 M462 M463 M464 M465 M466 M467 M468 M469 M470 M471 M472 M473 M474 M475 M476 M477 M478 M479 M480 M481 M482 M483 M484 M485 M486 M487 M488 M489 M490 M491 M492 M493 M494 M495 M496 M497 M498 M499 M500 M501 M502 M503 M504 M505 M506 M507 M508 M509 M510 M511 M512 M513 M514 M515 M516 M517 M518 M519 M520 M521 M522 M523 M524 M525 M526 M527 M528 M529 M530 M531 M532 M533 M534 M535 M536 M537 M538 M539 M540 M541 M542 M543 M544 M545 M546 M547 M548 M549 M550 M551 M552 M553 M554 M555 M556 M557 M558 M559 M560 M561 M562 M563 M564 M565 M566 M567 M568 M569 M570 M571 M572 M573 M574 M575 M576 M577 M578 M579 M580 M581 M582 M583 M584 M585 M586 M587 M588 M589 M590 M591 M592 M593 M594 M595 M596 M597 M598 M599 M600 M601 M602 M603 M604 M605 M606 M607 M608 M609 M610 M611 M612 M613 M614 M615 M616 M617 M618 M619 M620 M621 M622 M623 M624 M625 M626 M627 M628 M629 M630 M631 M632 M633 M634 M635 M636 M637 M638 M639 M640 M641 M642 M643 M644 M645 M646 M647 M648 M649 M650 M651 M652 M653 M654 M655 M656 M657 M658 M659 M660 M661 M662 M663 M664 M665 M666 M667 M668 M669 M670 M671 M672 M673 M674 M675 M676 M677 M678 M679 M680 M681 M682 M683 M684 M685 M686 M687 M688 M689 M690 M691 M692 M693 M694 M695 M696 M697 M698 M699 M700 M701 M702 M703 M704 M705 M706 M707 M708 M709 M710 M711 M712 M713 M714 M715 M716 M717 M718 M719 M720 M721 M722 M723 M724 M725 M726 M727 M728 M729 M730 M731 M732 M733 M734 M735 M736 M737 M738 M739 M740 M741 M742 M743 M744 M745 M746 M747 M748 M749 M750 M751 M752 M753 M754 M755 M756 M757 M758 M759 M760 M761 M762 M763 M764 M765 M766 M767 M768 M769 M770 M771 M772 M773 M774 M775 M776 M777 M778 M779 M780 M781 M782 M783 M784 M785 M786 M787 M788 M789 M790 M791 M792 M793 M794 M795 M796 M797 M798 M799 M800 M801 M802 M803 M804 M805 M806 M807 M808 M809 M810 M811 M812 M813 M814 M815 M816 M817 M818 M819 M820 M821 M822 M823 M824 M825 M826 M827 M828 M829 M830 M831 M832 M833 M834 M835 M836 M837 M838 M839 M840 M841 M842 M843 M844 M845 M846 M847 M848 M849 M850 M851 M852 M853 M854 M855 M856 M857 M858 M859 M860 M861 M862 M863 M864 M865 M866 M867 M868 M869 M870 M871 M872 M873 M874 M875 M876 M877 M878 M879 M880 M881 M882 M883 M884 M885 M886 M887 M888 M889 M890 M891 M892 M893 M894 M895 M896 M897 M898 M899 M900 M901 M902 M903 M904 M905 M906 M907 M908 M909 M910 M911 M912 M913 M914 M915 M916 M917 M918 M919 M920 M921 M922 M923 M924 M925 M926 M927 M928 M929 M930 M931 M932 M933 M934 M935 M936 M937 M938 M939 M940 M941 M942 M943 M944 M945 M946 M947 M948 M949 M950 M951 M952 M953 M954 M955 M956 M957 M958 M959 M960 M961 M962 M963 M964 M965 M966 M967 M968 M969 M970 M971 M972 M973 M974 M975 M976 M977 M978 M979 M980 M981 M982 M983 M984 M985 M986 M987 M988 M989 M990 M991 M992 M993 M994 M995 M996 M997 M998 M999 M1000
```

```
## Hoja de Peak: Idx Name Label Perc_missing QC_RSD
```

En la primera hoja (**Data**) encontramos las siguientes columnas:

- Idx: índice
- SampleID: identificador de muestra
- SampleType: indica si la muestra fue un control de calidad agrupado (QC) o una muestra de estudio.
- Class: indica el resultado clínico observado para ese individuo: GC = cáncer gástrico, BN = tumor benigno, HE = control sano
- M1 ... M149: describen las concentraciones de metabolitos

Mientras que para la segunda hoja (**Peak**) están las columnas:

- Idx: índice
- Name: encabezado de la columna correspondiente a este metabolito en la tabla de **Data**
- Label: nombre único para el metabolito (o un identificador uNNN)
- Perc_missing: porcentaje de muestras no contienen una medición para este metabolito (datos faltantes)
- QC_RSD: puntuación de calidad que representa la variación en las mediciones de este metabolito en todas las muestras

Con toda esta información podemos generar un objeto de la clase **SummarizedExperiment**. Estos objetos tienen tres componentes principales (slots):

- Assays: lista de **matrices** que contienen los datos de expresión. Cada matriz debe tener las mismas dimensiones, con filas representando características (genes, proteínas o metabolitos) y columnas representando las muestras.
- rowData: **DataFrame** que almacena los metadatos de las características. Cada fila de rowData corresponde a una fila en las matrices de **assays**. Puede incluir información como el nombre, identificador o calidad para cada metabolito.
- colData: **DataFrame** que almacena los metadatos de las muestras. Cada fila de colData corresponde a una columna en las matrices de **assays**. Puede incluir información sobre el tipo de muestra, tratamiento aplicado o resultado clínico asociado con cada muestra.

Para acceder a la información de cada uno de estos slots basta con llamar a las funciones **assay()**, **rowData()** y **colData()**, respectivamente.

La hoja **Data** posee información tanto para la matriz de los datos de expresión como para **colData**, mientras que la hoja **Peak** contiene información para **rowData**. Para la generación del objeto **SummarizedExperiment** vamos a obviar los índices **Idx** y dividir **data**:

```
# extraemos los metadatos de las muestras (SampleID, SampleType, Class)
sample_metadata <- data[, 2:4]
# extraemos los datos de concentración de metabolitos (columnas M1 a M149) como matriz
concentration <- t(as.matrix(data[, 5:ncol(data)])) # trasponer con t() para dejar
                                                    # las concentraciones de metabolitos
                                                    # en filas

# podríamos renombrar los metabolitos en esta matriz por los Labels de sample_metadata
# pero vamos a dejarlo así teniendo en cuenta que podemos generar una función para
# obtener el nombre del metabolito posteriormente

colnames(concentration) <- paste0(sample_metadata$SampleID,
```

```

        "_", sample_metadata$Class) # renombramos las columnas por los
                                   # nombres de las muestras con su categoría clínica

# creamos el objeto DataFrame de rowData con los metadatos de los metabolitos de la hoja Peak
rowData <- DataFrame(
  Name = peak$Name,
  Label = peak$Label,
  Perc_missing = peak$Perc_missing,
  QC_RSD = peak$QC_RSD
)

# creamos el objeto data.frame de colData con los metadatos de las muestras
colData <- as.data.frame(sample_metadata[1:ncol(sample_metadata)])

# transformamos el tipo de dato de Class a factores para análisis con POMA
colData$Class <- as.factor(colData$Class)

# renombramos la columna SampleID por `Sample name`
colnames(colData)[colnames(colData) == "SampleID"] <- "Sample name"

# cambiamos el orden de las columnas de colData y nos quedamos sólo con
colData <- colData[c("Class", "SampleType", "Sample name")]

```

También podemos extraer información de los metadatos del experimento a partir del Project ID PR000699 en el repositorio *Metabolomics Workbench*: <https://www.metabolomicsworkbench.org/data/DRCCMetadat.a.php?Mode=Project&ProjectID=PR000699>

Además, podemos ver que el proyecto PR000699 se asocia al estudio ST001047. Realizando un procedimiento de *web scraping* con *rvest* podríamos obtener más información sobre las muestras para el slot `colData`.

Con todo ello, ya podemos crear nuestro objeto contenedor de la clase `SummarizedExperiment` y guardar el archivo binario `Rda`:

```

# creamos el objeto SummarizedExperiment
se <- SummarizedExperiment(
  assays = list(concentration = concentration),
  rowData = rowData,
  colData = colData,
  metadata = experiment_metadata
)

# guardamos el objeto en un archivo binario Rda
save(se, file = "SummarizedExperiment.Rda")

# guardamos los metadatos en archivos CSV
write.csv(colData, "sample_metadata.csv", row.names = FALSE)
write.csv(rowData, "variable_metadata.csv", row.names = FALSE)

# guardamos los datos en CSV
write.csv(as.data.frame(concentration), "data_assay_matrix.csv", row.names = TRUE)

# archivo md con metadatos
# inicializamos el contenido
contenido_md <- c(
  "# Resumen de los Metadatos\n",

```

```

"\n## Metadatos de Muestras\n",
knitr::kable(as.data.frame(colData), format = "markdown"),
"\n\n## Metadatos de Features\n",
knitr::kable(as.data.frame(rowData), format = "markdown")
)

# escribimos el archivo md
writeLines(contenido_md, "resumen_metadatos.md")

```

Hasta aquí el proceso de recogida de datos, de una forma manual, aprovechando que disponíamos de la información en el repositorio de *GitHub*. No obstante, el repositorio de *Metabolomics Workbench* posee una API REST que puede ser manejada fácilmente con el paquete *metabolomicsWorkbenchR* de Bioconductor. Tan solo hay que utilizar la función `do_query` con los argumentos apropiados para extraer la información que nos interese:

```

# opciones disponibles para un contexto de estudio concreto
# metabolomicsWorkbenchR::context_outputs(context = 'study')

mwb.summ <- do_query(context = 'study', input_item = 'study_id',
                     input_value = 'ST001047', output_item = 'summary') # resumen
mwb.data <- do_query(context = 'study', input_item = 'study_id',
                     input_value = 'ST001047', output_item = 'data') # datos
mwb.factors <- do_query(context = 'study', input_item = 'study_id',
                        input_value = 'ST001047', output_item = 'factors') # colData
mwb.metabolites <- do_query(context = 'study', input_item = 'study_id',
                             input_value = 'ST001047', output_item = 'metabolites') # colData

# no funciona la extracción directa del objeto SummarizedExperiment
# (Error en SummarizedExperiment(assays = list(X), rowData = VM, colData = SM, :
# the rownames and colnames of the supplied assay(s) must be NULL or identical to
# those of the SummarizedExperiment object (or derivative) to construct)

# mwb.se <- do_query(context = 'study', input_item = 'study_id',
#                     input_value = 'ST001047', output_item = 'SummarizedExperiment')

columns_to_select <- names(mwb.data$AN001711)[c(8:ncol(mwb.data$AN001711))]
assay.data <- as.data.frame(subset(mwb.data$AN001711, select = columns_to_select))
rownames(assay.data) <- mwb.data$AN001711$metabolite_name

# mwb.factors no dispone de las muestras QC. Este desacoplamiento produce el problema en
# la descarga directa del SummarizedExperiment con do_query()
qc.factors <- data.frame(
  "study_id" = rep("ST001047", 17),
  "local_sample_id" = c("sample_1", "sample_10", "sample_100", "sample_109", "sample_118", "sample_127",
                        "sample_140", "sample_19", "sample_28", "sample_37", "sample_46", "sample_55", "sample_82", "sample_91"),

  "sample_source" = rep("Urine", 17),
  "mb_sample_id" = c("SA070439", "SA070447", "SA070437", "SA070445", "SA070435", "SA070443", "SA070446",
                    "SA070434", "SA070432", "SA070433", "SA070444", "SA070442", "SA070440", "SA070441", "SA070438"),
  "raw_data" = rep("", 17),
  "subject_type" = rep(NA, 17),

```

```

"Sample_Type" = rep("QC", 17)
)

# unimos en un único data.frame los datos
factors <- rbind(qc.factors, mwb.factors$ST001047)

# creamos el objeto SummarizedExperiment
se2 <- SummarizedExperiment(
  assays = list(concentration = assay.data),
  rowData = mwb.metabolites,
  colData = factors,
  metadata = mwb.summ
)

se2

## class: SummarizedExperiment
## dim: 129 140
## metadata(15): study_id study_title ... study_summary subject_species
## assays(1): concentration
## rownames(129): 1_3-Dimethylurate 1_6-Anhydro- -D-glucose ...
##      -Methylhistidine -Methylhistidine
## rowData names(8): study_id analysis_id ... other_id other_id_type
## colnames(140): sample_1 sample_10 ... sample_98 sample_99
## colData names(7): study_id local_sample_id ... subject_type Sample_Type

# guardamos el objeto en un archivo binario Rda
save(se, file = "SummarizedExperiment_from_MWB.Rda")

```

3. Exploración de los datos

El flujo de trabajo que vamos a seguir para la exploración de los datos es el siguiente:

1. Exploración y Preprocesamiento
 - Exploración inicial: exploración global con análisis descriptivo de los datos.
 - Filtrado: filtraremos metabolitos con QC_RSD mayor al 20% y Perc_missing superior al 10% para eliminar aquellos con baja calidad de medición.
 - Imputación de valores faltantes: si existen valores faltantes, se usará el método KNN para imputarlos.
2. Transformación y normalización de los datos
 - Transformación logarítmica: aplicamos una transformación logarítmica para reducir la asimetría en la distribución de los datos.
 - Normalización de los datos: estandarizamos los datos del assay para su análisis.
 - Eliminación de outliers, si los hay.
3. Análisis exploratorio de los datos
 - Análisis estadístico global de los datos crudos. Comparación entre categorías GC y HE para los distintos metabolitos mediante test de Welch y gráficamente con Volcano plot con los datos normalizados.
 - Análisis de componentes principales (PCA): para evaluar la variación en los datos y visualizar la agrupación de las muestras por categoría clínica.

- Visualización de clusters: utilizaremos una agrupación jerárquica para explorar clusters en los datos que puedan orientarnos sobre un posible *efecto batch* de los datos.

Exploración y Preprocesamiento Antes de nada, vamos a realizar un análisis descriptivo para calcular estadísticos para cada metabolito. Empezaremos extrayendo los datos de concentración de los metabolitos del objeto `SummarizedExperiment`.

Resumimos los estadísticos descriptivos en un `DataFrame`:

```
summary.concentration.data <- data.frame(
  Mean = apply(concentration.data, 1, mean, na.rm=TRUE), # media
  Median = apply(concentration.data, 1, median, na.rm=TRUE), # mediana
  SD = apply(concentration.data, 1, sd, na.rm=TRUE), # desviación estándar
  Min = apply(concentration.data, 1, min, na.rm=TRUE), # mínimo
  Max = apply(concentration.data, 1, max, na.rm=TRUE), # máximo
  Q1 = apply(concentration.data, 1, quantile, probs = 0.25, na.rm=TRUE), # 1er cuartil
  Q3 = apply(concentration.data, 1, quantile, probs = 0.75, na.rm=TRUE), # 3er cuartil
  IQR = apply(concentration.data, 1, IQR, na.rm=TRUE), # rango intercuartílico
  perc_NA = apply(concentration.data, 1, function(x){ # porcentaje de missing values
    sum(is.na(x)/length(x)*100)
  })
)

head(summary.concentration.data, 10) # sólo mostramos las 10 primeras filas
```

##		Mean	Median	SD	Min	Max	Q1	Q3	IQR	perc_NA
##	M1	101.07097	60.35	123.61378	0.4	909.9	29.825	133.375	103.550	11.4285714
##	M2	641.99784	270.20	2397.53563	3.1	26195.8	140.900	480.950	340.050	0.7142857
##	M3	146.36692	105.10	131.85017	0.1	862.5	53.600	198.800	145.200	5.0000000
##	M4	43.83359	35.70	39.05195	0.1	242.5	18.775	51.325	32.550	8.5714286
##	M5	231.10797	160.35	337.54214	1.3	2503.0	67.000	253.075	186.075	1.4285714
##	M6	41.63383	25.90	48.40078	0.2	339.4	15.500	48.600	33.100	5.0000000
##	M7	74.11985	40.25	97.37999	4.6	492.6	19.650	65.550	45.900	2.8571429
##	M8	67.80929	51.00	61.13140	9.3	525.0	37.175	77.825	40.650	0.0000000
##	M9	64.09912	42.20	78.15066	0.7	612.1	20.100	75.300	55.200	19.2857143
##	M10	124.80930	80.80	205.71543	0.1	2026.8	38.400	148.000	109.600	7.8571429

En estos resultados observamos que los distintos metabolitos tienen una gran variabilidad en sus concentraciones, evidenciándose por las diferencias en la desviación estándar (SD) y el rango intercuartílico (IQR). Por ejemplo, M2 tiene una desviación estándar mucho mayor que M1, lo que sugiere una distribución más dispersa. Por otro lado, la diferencia observada entre media y mediana sugiere una distribución sesgada, con asimetría positiva. Estos resultados indican la necesidad de normalización para reducir la influencia de los metabolitos con valores altamente dispersos en el análisis multivariante.

Además, tendremos que realizar un manejo adecuado de los valores faltantes, con imputación de valores, ya que tenemos para algunos metabolitos encontramos elevados porcentajes de missing values (por ejemplo en M9).

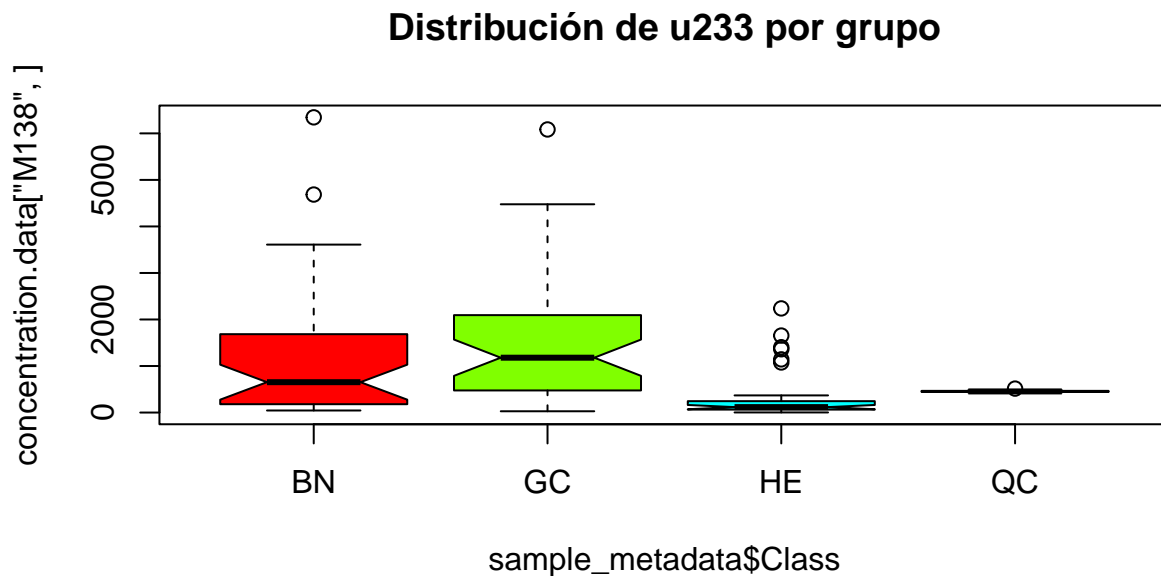
Por último, podríamos ver las distribuciones por grupos (QC, GC, BN, HE), para explorar posibles diferencias entre los grupos realizando un `boxplot` para cada uno de los metabolitos. Puesto que son en total 149 metabolitos, guardaremos para visualizar posteriormente los gráficos de cada uno de ellos en un archivo PDF nombrado como `metabolitos_boxplots.pdf`

Gráficos `boxplot` guardados en `metabolitos_boxplots.pdf`

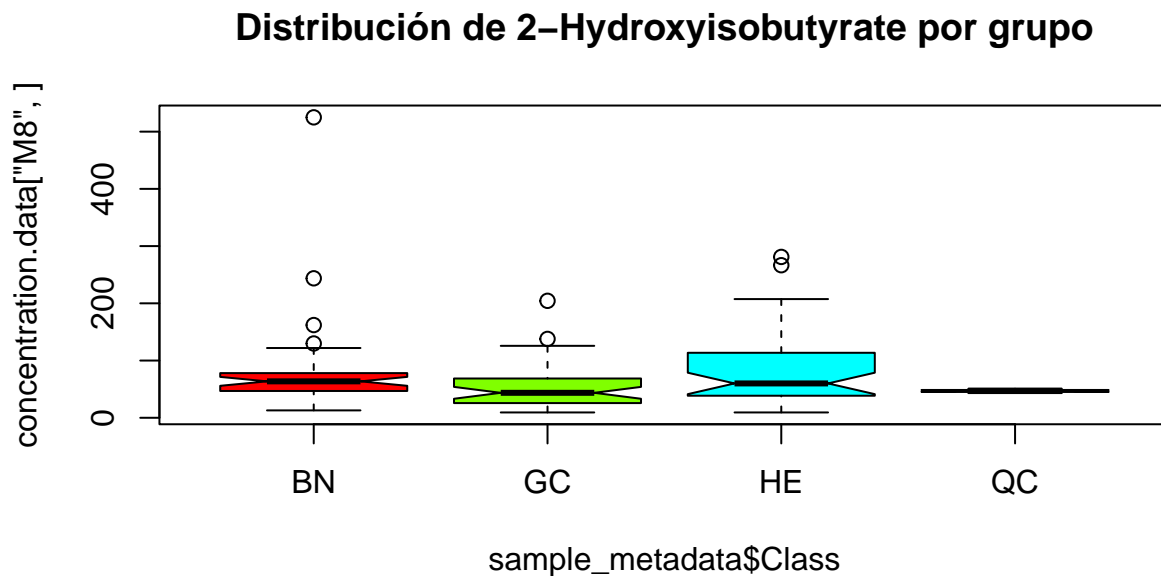
Similarmente, podemos crear histogramas de las distribuciones de los metabolitos y guardarlo en otro archivo PDF nombrado como `metabolitos_histograms.pdf`.

Gráficos de histogramas guardados en metabolitos_histograms.pdf

Las distribuciones son sesgadas en su mayoría, tal y como habíamos visto anteriormente. La inspección visual de los resultados nos muestra posibles diferencias entre los grupos clínicos para ciertos metabolitos y valores atípicos. Así pues, encontramos tanto metabolitos más concentrados en el grupo GC en comparación con el grupo HE:



Como menos concentrados:



Lo siguiente que vamos a hacer es un filtrado de datos. Observamos que tenemos unos estadísticos previamente calculados en el propio dataset para cada metabolito: QC-RSD y Perc_missing.

El *QC-RSD* (Relative Standard Deviation de los controles de calidad) es una medida de la variabilidad relativa de un metabolito a través de las muestras de control de calidad (QC). Se calcula como el coeficiente de variación (CV), expresado como porcentaje, para un metabolito específico en las muestras de control de calidad. Este valor indica la reproducibilidad de las mediciones y se utiliza comúnmente en metabolómica para evaluar la calidad de los datos; valores altos indican mayor variabilidad y menor reproducibilidad.

El cálculo se realiza con la siguiente fórmula:

$$QC_RSD = \left(\frac{SD}{media} \right) \times 100$$

Vamos a filtrar los metabolitos con *QC_RSD* mayor al 20% y *Perc_missing* superior al 10% para eliminar aquellos con baja calidad de medición, a la vez que imputamos los valores faltantes mediante el método basado en distancias K-Nearest Neighbor (KNN) con la función *PomaImpute*:

```
library("POMA")

## Welcome to POMA!
## Version 1.14.0
## POMAShiny app: https://github.com/pcastellanoescuder/POMAShiny

# recogemos los rowData de los metabolitos
rowData_se <- rowData(se)

# filtramos por QC_RSD < 20
filtered_indices <- which(rowData_se$QC_RSD < 20)

# creamos un nuevo objeto SummarizedExperiment con los metabolitos filtrados
se_filtered <- se[filtered_indices, ]

imputed <- PomaImpute(se_filtered, zeros_as_na = FALSE,
                      remove_na = TRUE, method = "knn", cutoff = 10)

# ver el número de metabolitos antes y después del filtrado
cat("Número de metabolitos originales:", nrow(se), "\n")

## Número de metabolitos originales: 149
cat("Número de metabolitos filtrados:", nrow(imputed), "\n")

## Número de metabolitos filtrados: 53
```

Hemos reducido de 149 a 53 metabolitos para su análisis. En este caso, nuestro conjunto de datos es algo menos común de lo que se esperaría para estudios ómicos típicos, donde suele haber un número muy elevado de variables que excede con creces el número de muestras. A tener en cuenta, la aplicación del método *PomaImpute* ha descartado el slot *rowData*. Vamos a crear una función que permita extraer el nombre del metabolito en cuestión dado un *Mx*:

```
# función para obtener el label de un metabolito específico
obtener_label_metabolito <- function(metabolito, se) {
  # extraemos la fila donde se encuentra el nombre del metabolito en el objeto original
  row <- rowData(se)[rowData(se)$Name == metabolito,]
  # extraemos el Label correspondiente para devolverlo
  label <- row$Label
  return(label)
}
```

Transformación y normalización de los datos Ahora es necesario transformar y normalizar los datos para el uso de algoritmos basados en distancias (como, por ejemplo, PCA), con el método `log_scaling` de `PomaNorm`:

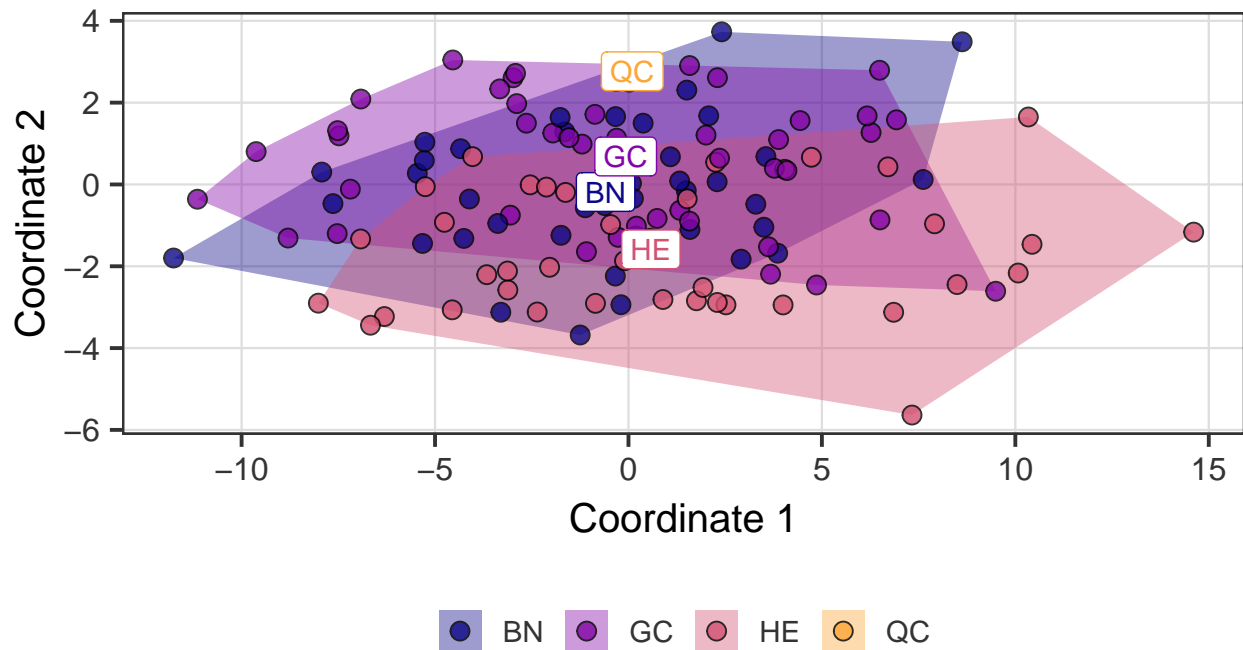
```
normalized <- PomaNorm(imputed, method = "log_scaling")
normalized
```

```
## class: SummarizedExperiment
## dim: 53 140
## metadata(0):
## assays(1): ''
## rownames(53): M4 M5 ... M148 M149
## rowData names(0):
## colnames(140): sample_1_QC_B1 sample_2_GC_B1 ... sample_139_HE_B4
##   sample_140_QC_B4
## colData names(5): Class SampleType Sample name mb_sample_id Batch
```

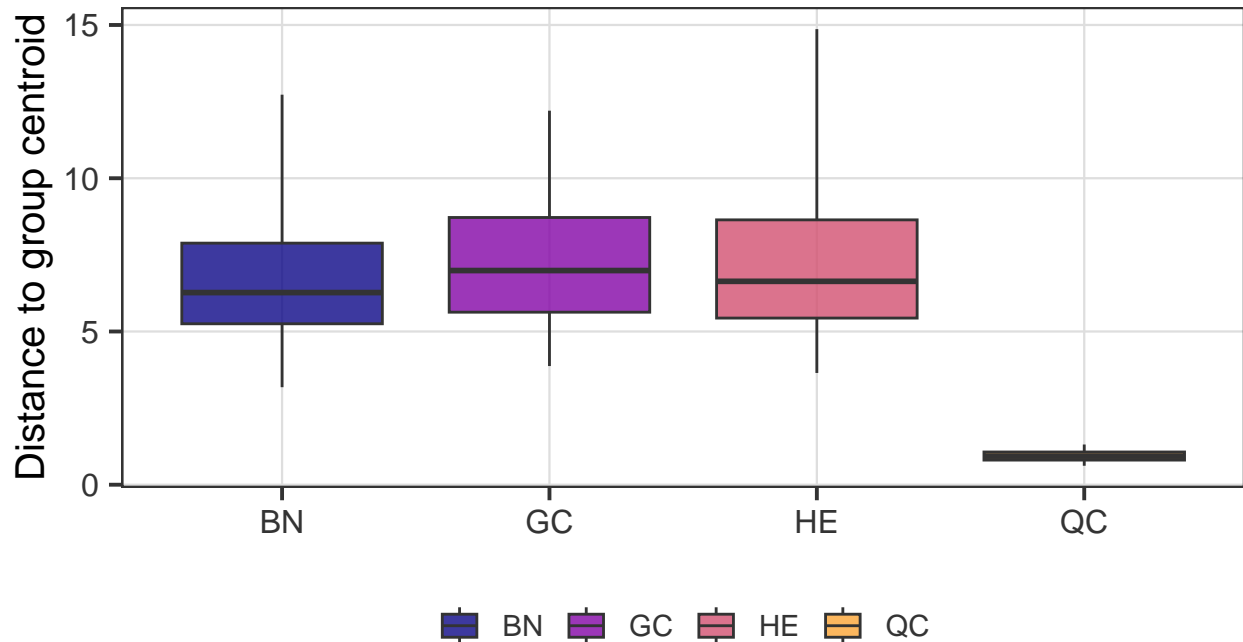
Comprobamos la existencia de posibles valores atípicos *outliers*:

```
PomaOutliers(normalized)
```

```
## $polygon_plot
```



```
##
## $distance_boxplot
```

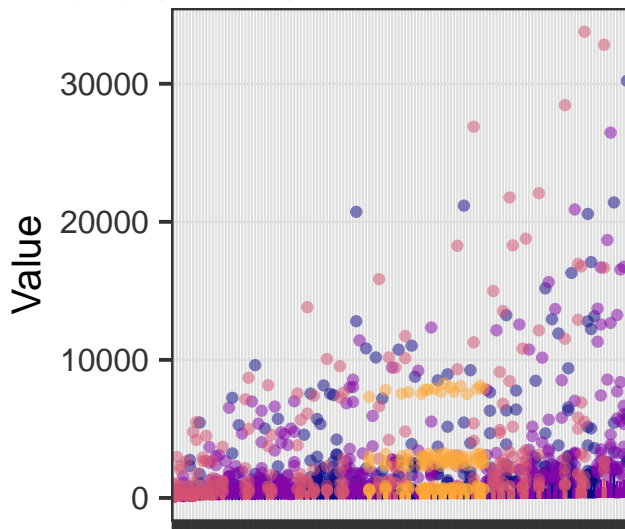


```
##
## $outliers
## # A tibble: 0 x 4
## #   i 4 variables: sample <chr>, groups <fct>, distance_to_centroid <dbl>,
## #   limit_distance <dbl>
##
## $data
## class: SummarizedExperiment
## dim: 53 140
## metadata(0):
## assays(1): ''
## rownames(53): M4 M5 ... M148 M149
## rowData names(0):
## colnames(140): sample_1_QC_B1 sample_2_GC_B1 ... sample_139_HE_B4
##   sample_140_QC_B4
## colData names(5): class sample_type sample_name mb_sample_id batch
```

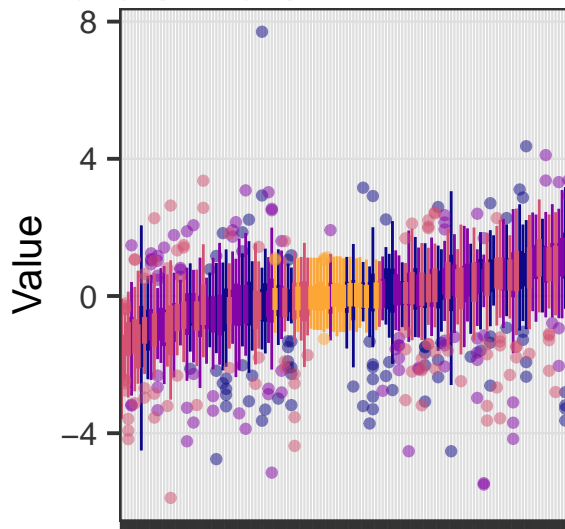
Aunque parece que sigue existiendo cierto sesgo en las distribuciones por categorías clínicas (fundamentalmente con HE), no encontramos muestras *outliers*.

Podemos ver el resultado de nuestra transformación de los datos, tanto para muestras como metabolitos, con PomaBoxplots:

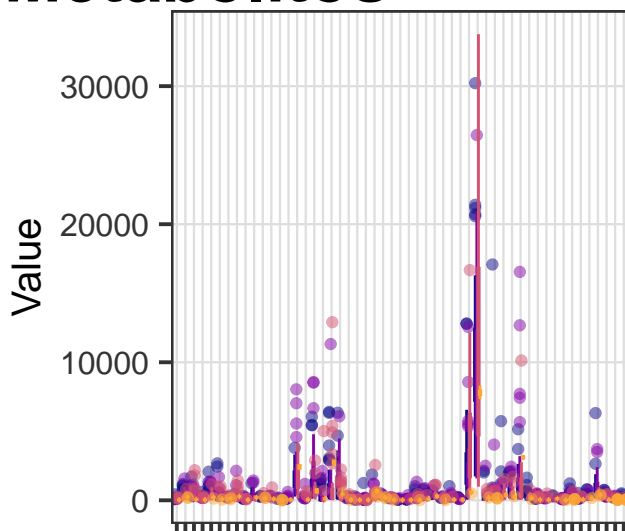
Sin normalizar – Muestras



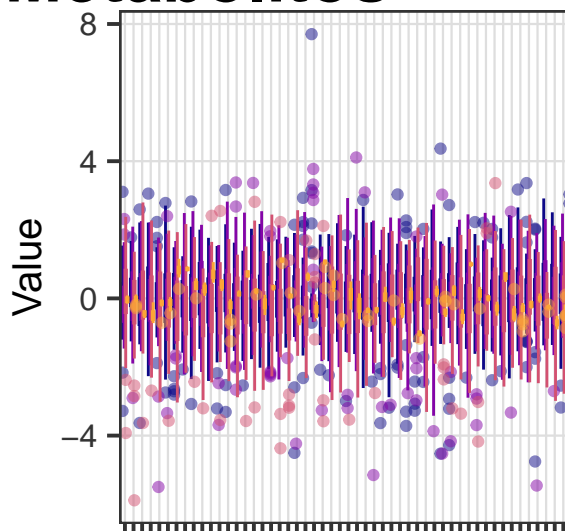
Normalizados – Muestras



Sin normalizar – Metabolitos

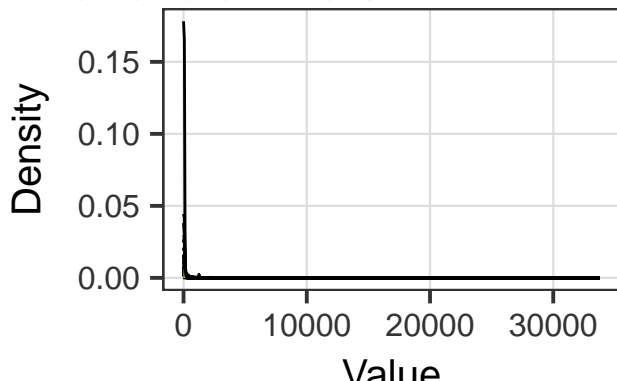


Normalizados – Metabolitos

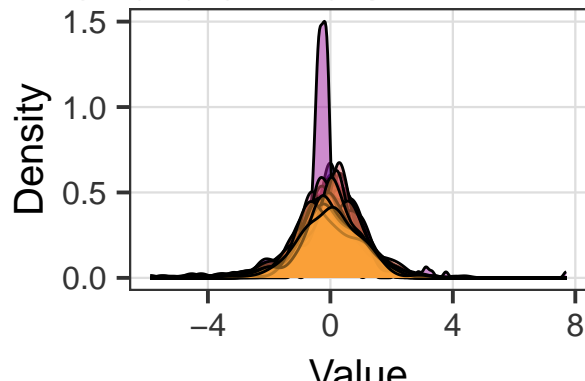


Por otro lado, con PomaDensity podemos obtener gráficos de densidad antes y después del proceso de normalización, tanto para metabolitos como muestras.

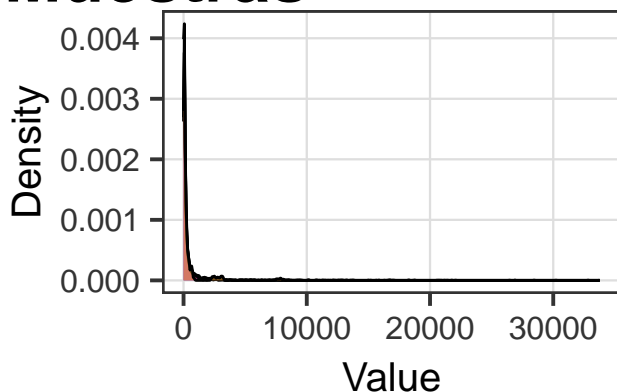
Sin normalizar – Metabolitos



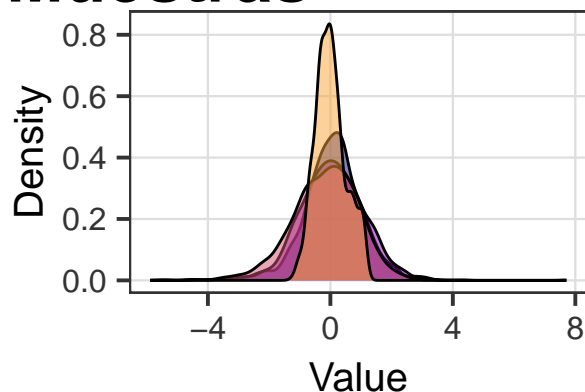
Normalizados – Metabolitos



Sin normalizar – Muestras



Normalizados – Muestras



En ambos gráficos (diagrama de cajas y gráficos de densidad), observamos que:

- Algunos metabolitos tienen valores notablemente altos en comparación con otros, lo que sugiere una distribución heterogénea de las concentraciones, como vimos anteriormente en la exploración global. La normalización después de la transformación logarítmica consigue que la mayoría de las concentraciones de metabolitos se agrupen en un rango mucho más homogéneo, haciendo que los metabolitos sean comparables entre sí. Aún así, se observan metabolitos con valores atípicos, como con el metabolito M144 (u87).
- En cuanto a las muestras, tras la transformación logarítmica se observan similares en cuanto a las distribuciones de los valores de los metabolitos, con algunos valores atípicos para ciertas categorías clínicas. En las muestras de la categoría QC es donde existe mayor similitud de distribución. Eliminaremos esta categoría para análisis sucesivos.

Análisis exploratorio de los datos Centraremos el análisis exploratorio univariado solamente a los datos de las categorías clínicas GC y HE.

Con `PomaUnivariate` podemos realizar pruebas estadísticas univariadas paramétricas y no paramétricas en

un objeto `SummarizedExperiment` para comparar grupos o condiciones:

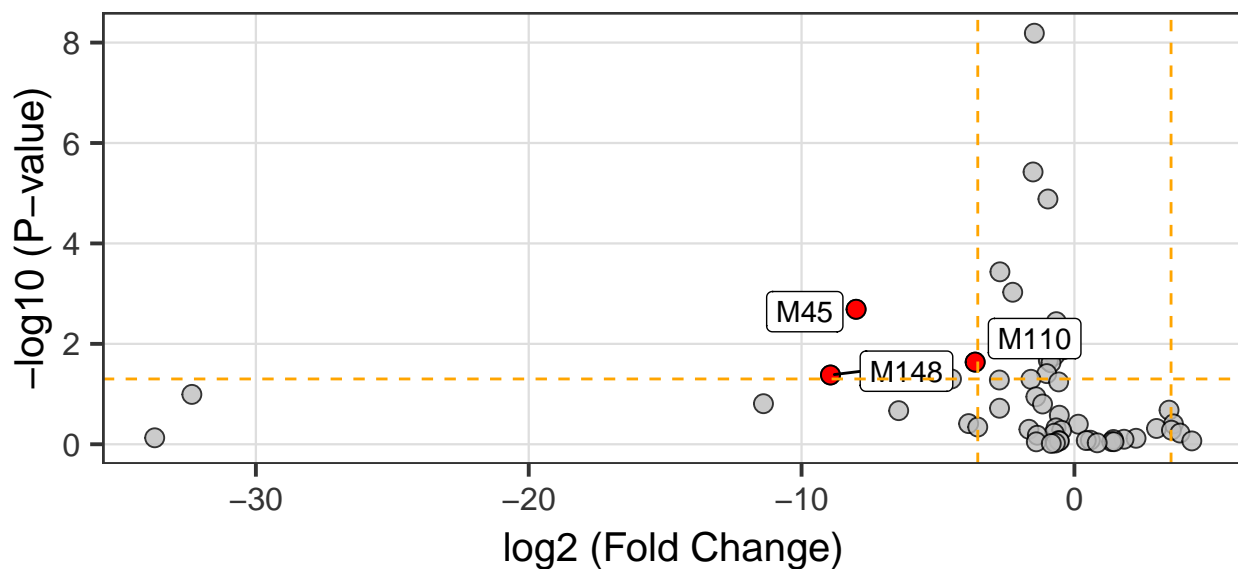
```
PomaUnivariate(normalized[, normalized$Class %in% c("GC", "HE")], method = "ttest",
  var_equal = FALSE, adjust = "fdr") # test t de Welch
```

```
## $result
## # A tibble: 53 x 9
##   feature fold_change diff_means pvalue adj_pvalue mean_GC mean_HE sd_GC sd_HE
##   <chr>      <dbl>      <dbl>   <dbl>   <dbl>   <dbl>   <dbl> <dbl> <dbl>
## 1 M138      -1.47      -1.32 6.51e-9 3.45e-7 0.534   -0.785 0.784 1.02
## 2 M89       -1.52      -1.06 3.77e-6 1.00e-4 0.420   -0.636 0.945 0.989
## 3 M134      -0.97      -1.00 1.30e-5 2.30e-4 0.510   -0.495 1.03 0.944
## 4 M118      -2.73      -0.834 3.68e-4 4.87e-3 0.224   -0.611 1.07 0.974
## 5 M142      -2.26      -0.74 9.39e-4 9.95e-3 0.227   -0.513 1.02 0.943
## 6 M45       -8.00        0.61 2.05e-3 1.81e-2 -0.0678 0.543 0.873 0.871
## 7 M7        -0.663     -0.702 3.61e-3 2.74e-2 0.422   -0.280 1.19 0.939
## 8 M4        -0.42        0.544 1.32e-2 8.74e-2 -0.383 0.161 0.971 0.982
## 9 M91       -0.725     -0.556 1.72e-2 1.01e-1 0.322   -0.234 0.914 1.14
## 10 M32      -0.956     -0.557 2.15e-2 1.05e-1 0.284   -0.272 1.27 0.862
## # i 43 more rows
```

Se observan diferencias significativas con respecto a distintos metabolitos entre ambas categorías clínicas, como con u233 (M138), N-AcetylglutamineDerivative (M89), u144 (M134), Tropate (M118) o u43 (M142). Sin embargo, lo que nos interesa conocer no es sólo las diferencias que puedan ser significativas, sino que puedan ser biológicamente relevantes priorizando metabolitos con alta significancia y cambios en magnitud importantes (*fold-change*).

Para ello, realizaremos un *volcano plot*, el cual representa el *fold-change* frente al logaritmo negativo de los p-valores para hallar los metabolitos que podrían estar contribuyendo a la separación observada entre “GC” y “HE” y que merecerían una mayor atención en estudios futuros para su validación como posibles biomarcadores:

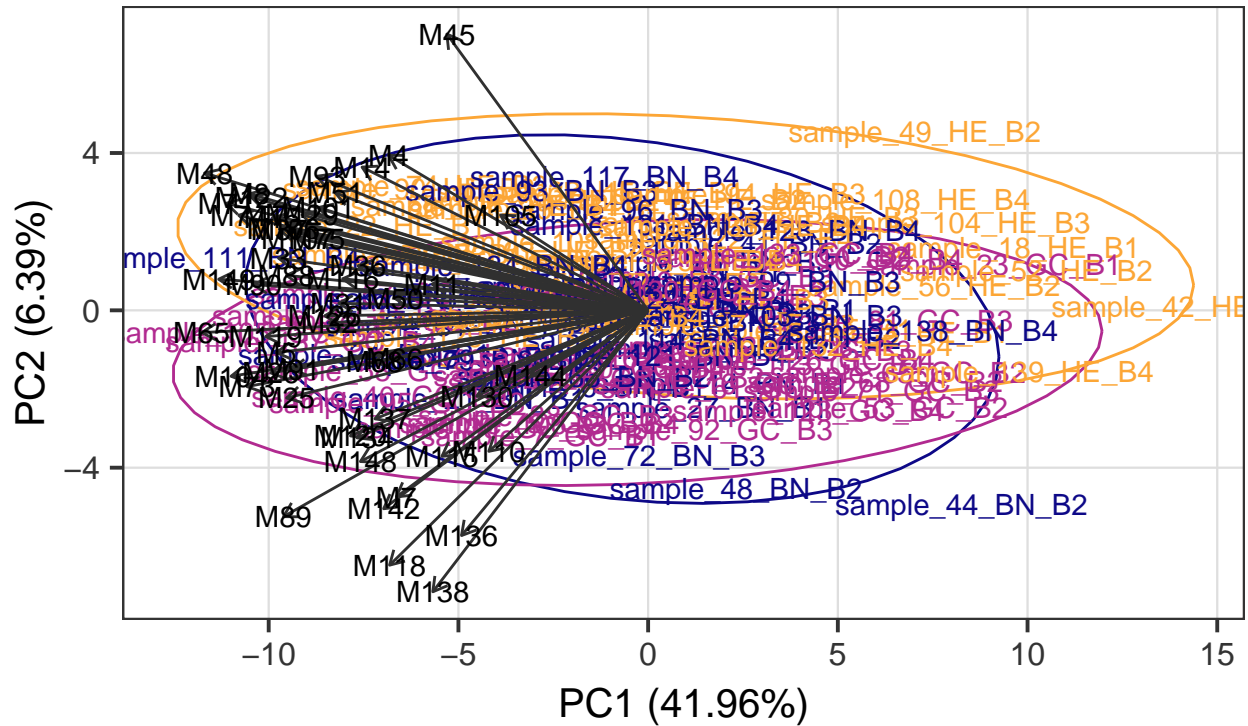
```
PomaUnivariate(normalized[, normalized$Class %in% c("GC", "HE")],
  method = "ttest", var_equal = FALSE, adjust = "fdr") %>%
  magrittr::extract2("result") %>%
  dplyr::select(feature, fold_change, pvalue) %>%
  PomaVolcano(labels=TRUE)
```



Observamos 3 metabolitos a considerar: Citrate (M45), -Methylhistidine (M148) y Serotonin (M110).

A continuación realizaremos un análisis de componentes principales (PCA) con las 3 categorías clínicas, pues visualizar los datos en dimensión reducida puede ayudar a detectar posibles patrones ocultos en los datos, con la función PomaPCA:

```
pca <- PomaPCA(normalized, ellipse = TRUE, labels=TRUE, load_length = 1.1)
pca$biplot
```



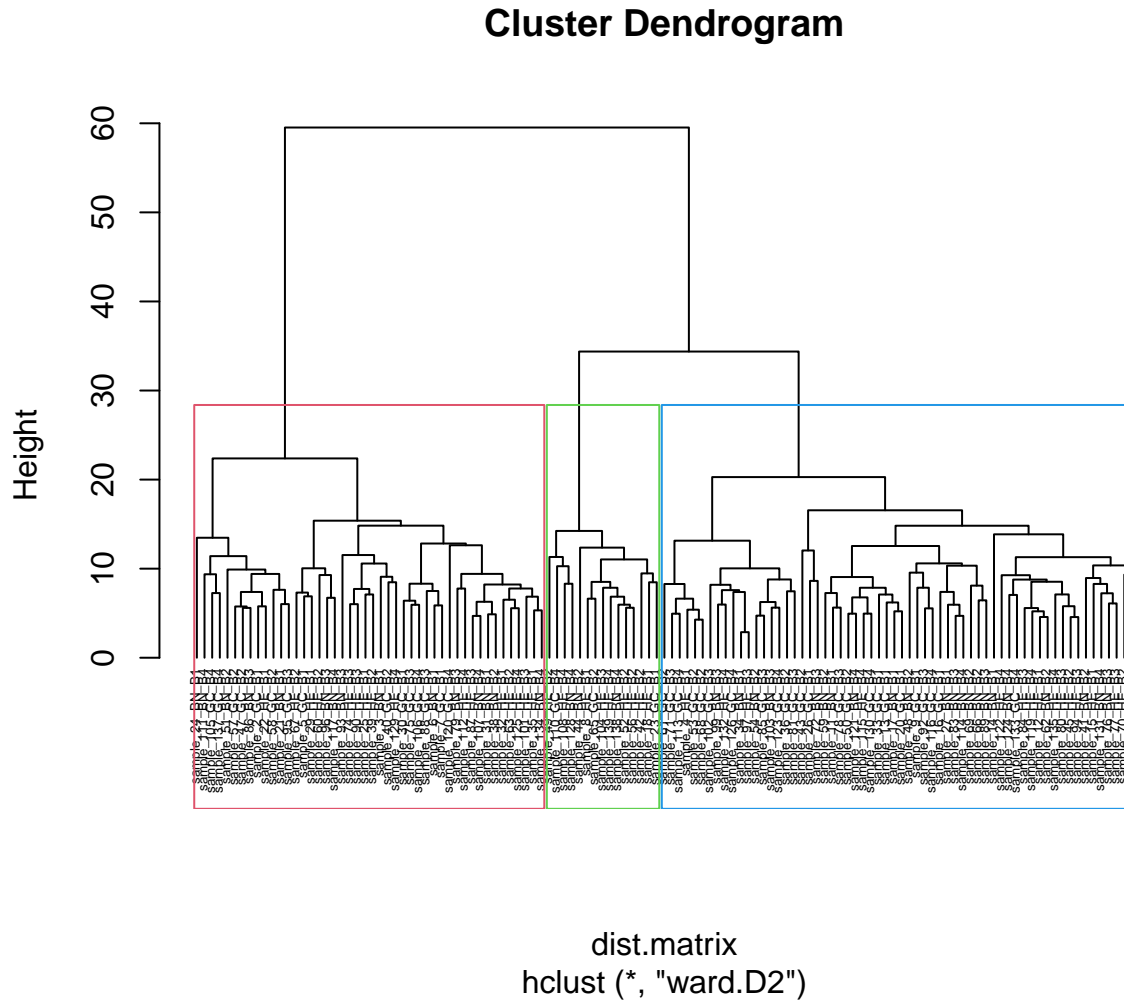
a BN a GC a HE

En este caso, con 2 componentes principales estamos explicando sólo un 48.35% de la variabilidad de los datos, aunque parece que la segunda componente está mayormente asociada a las distintas categorías clínicas, con GC en la parte inferior, HE en la superior y BN en posición intermedia, solapando con las categorías anteriores. En esta componente PC2 los 5 metabolitos que dominan las carga son: u233 (M138), Citrate (M45), Tropate (M118), u185 (M136) y N-AcetylglutamineDerivative (M89).

Por último, veamos si existe *efecto batch* mediante un agrupamiento jerárquico y visualización del dendrograma:

```
dist.matrix <- dist(t(assay(normalized)))
hc_res <- hclust(dist.matrix, method = "ward.D2")
sub_grp <- cutree(hc_res, k=3)

plot(hc_res, hang = -1, cex = 0.45)
rect.hclust(hc_res, k=3, border=2:(3+1))
```



No hay clusters que separen entre los distintos lotes, por lo que descartamos *efecto batch*. Tampoco hay un buen agrupamiento de las muestras según la categoría clínica.

Creación del repositorio Para la creación del repositorio y control de versiones en *GitHub* utilizaremos el método de *GitHub primero*. Para ello, en primer lugar hay que crear un nuevo repositorio en *GitHub*, desde el Dashboard principal haciendo click el botón verde *New*. Se abre entonces un formulario en donde rellenar el nombre del repositorio, en nuestro caso: `Martinez-Lopez-Jesus-PEC1` (Figura 6 del Anexo II). Una vez creado, copiamos al portapapeles la dirección del repositorio (Figura 7 del Anexo II).

Después tenemos que abrir un nuevo proyecto en *RStudio*, eligiendo la opción de *Version Control* y *Git*, en la ventana posterior (Figura 8 del Anexo II). En la nueva ventana que se abre habrá que pegar la dirección del repositorio del portapapeles (<https://github.com/GsusML84/Martinez-Lopez-Jesus-PEC1.git>) y crear el proyecto:

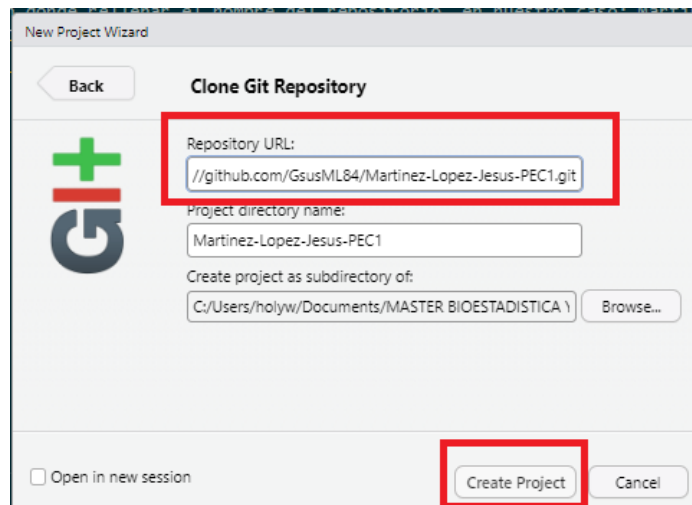


Figure 1: Proyecto R con control de cambios Git

A partir de ese momento el repositorio en *GitHub* estará conectado con *RStudio* gracias al *token* de acceso personal que tenemos que haber creado previamente. Podemos hacer *Push* y *Pull*, es decir, subir nuevas versiones modificadas o bajarlas desde, respectivamente, *GitHub*. En el panel derecho superior de *RStudio* encontramos un acceso rápido para el manejo de versiones en la pestaña *Git* (Figura 9 del Anexo II). Para hacer un *Push* hay que elaborar previamente un *commit*, esto es, elegir los archivos con los cambios que queremos subir, asociarlos a un mensaje y pulsar el botón de *Commit*:

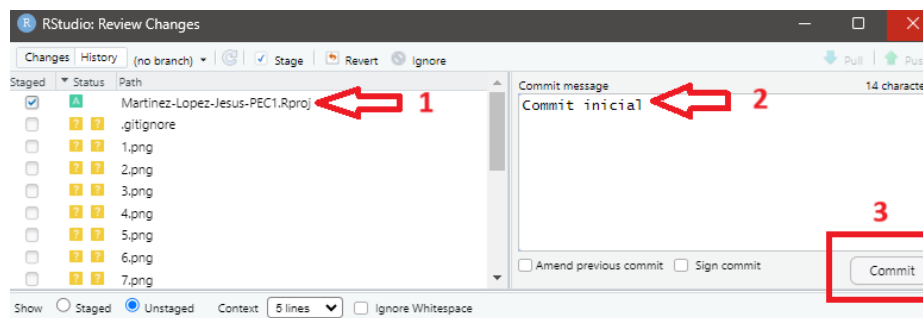


Figure 2: Commit

Una vez concluido el proceso de *commit* se muestran disponibles los botones de *Pull* (azul) y *Push* (verde) (Figura 10 del Anexo II). Pulsamos en el botón *Push* para subir la versión más reciente de los archivos en local al repositorio de *GitHub*. En la pestaña *History* podemos acceder al histórico del control de versiones (Figura 11 del Anexo II)

Discusión

El análisis exploratorio nos ha mostrado la siguiente información sobre los datos a destacar:

- Existen numerosos valores faltantes que hemos tenido que imputar con haciendo uso del algoritmo

KNN

- Las distribuciones de los metabolitos son asimétricas, razón por la cual hemos tenido que trabajar con ellos tras la transformación logarítmica y normalización.
- Hemos encontrado 3 metabolitos que podrían ser candidatos para su estudio en profundidad si queremos elaborar modelos predictivos para las categorías clínicas GC y HE.
- El Análisis de Componentes Principales nos muestra un solapamiento de la categoría clínica BN con HE y GC, más separadas entre sí.
- El lote no parece ser una fuente de variabilidad en los datos, por lo que descartamos *efecto batch*.

Por último, hemos visto que el control de versiones Git usando el esquema de *GitHub primero* es sencillo de aplicar en RStudio. El repositorio creado en *GitHub* está disponible públicamente en la dirección: **<https://github.com/GsusML84/Martinez-Lopez-Jesus-PEC1.git>**

Referencias

- Chan, Angela W., Pascal Mercier, Daniel Schiller, Robert Bailey, Sarah Robbins, Dean T. Eurich, Michael B. Sawyer, and David Broadhurst. 2015. "1H-NMR Urinary Metabolomic Profiling for Diagnosis of Gastric Cancer." *British Journal of Cancer* 114 (1): 59. <https://doi.org/10.1038/bjc.2015.414>.
- Metabolomics Workbench. 2018. "PR000699." Metabolomics Workbench. <https://doi.org/10.21228/M8B10B>.
- "Metabolomics Workbench : Home." n.d. Accessed November 2, 2024. <https://www.metabolomicsworkbench.org/>.
- "POMA. Bioconductor." n.d. Accessed November 2, 2024. <https://www.bioconductor.org/packages/devel/bioc/html/POMA.html>.

Anexos

Anexo I - Código R

```
## ## ----setup, include=FALSE-----
## required_packages <- c("BiocManager", "knitr", "kableExtra", "SummarizedExperiment", "metabolomicsWo
##                               "readxl", "ggplot2", "gggraph", "plotly", "patchwork", "rvest")
##
## if (!requireNamespace("BiocManager", quietly = TRUE)) {
##   install.packages("BiocManager")
## }
##
## for (package in required_packages) {
##   if (!requireNamespace(package, quietly = TRUE)) {
##     if (package == "SummarizedExperiment" | package == "metabolomicsWorkbenchR") {
##       BiocManager::install(package)
##     } else {
##       install.packages(package)
##     }
##   }
## }
##
## lapply(required_packages, library, character.only = TRUE)
##
## ## ----actualizacion R, include=FALSE-----
## # knitr::purl("PEC1.Rmd", output = "PEC1.R")
##
##
## ## ----carga DataCatalog.xlsx, cache=TRUE-----
## # URL del archivo Data_Catalog.xlsx en GitHub
## github.url <- "https://github.com/nutrimetabolomics/metaboData/raw/refs/heads/main"
## file <- "Data_Catalog.xlsx"
##
## # URL completa usando file.path()
## datasets_catalog.url <- file.path(github.url, file)
##
## # descargamos el archivo
## download.file(datasets_catalog.url, destfile = "Data_Catalog.xlsx", mode = "wb")
##
## # leemos el archivo .xlsx descargado
## data <- read_excel("Data_Catalog.xlsx", sheet = 1)
##
## # mostramos el contenido
## kable(data[,1:ncol(data)-1]) # sin mostrar la descripción, última columna
##
##
## ## ----seleccion aleatoria del dataset, cache=TRUE-----
## set.seed(123) # semilla aleatoria
## selection<- sample(1:nrow(data),1)
##
## kable(data[selection, 1:ncol(data)-1])
##
## # descripción del estudio
## cat(data[selection, ncol(data)]$Description)
```

```

##
##
## ## ----lectura de description.md, warning=FALSE, echo=FALSE-----
## dataset.folder.url <- file.path(github.url, "Datasets", data[selection,]$Dataset)
## file <- "description.md"
##
##
## description.url <- file.path(dataset.folder.url, file)
## description.md <- readLines(description.url)
##
## cat(description.md, sep = "\n")
##
##
## ## ----descarga del archivo dataset del repositorio de Github, cache=TRUE-----
## file <- "GastricCancer_NMR.xlsx"
##
## # URL completa usando file.path()
## dataset.url <- file.path(dataset.folder.url, file)
##
## # descargamos el archivo
## download.file(dataset.url, destfile = "GastricCancer_NMR.xlsx", mode = "wb")
##
##
## ## ----estructura del archivo dataset en Excel, echo=FALSE-----
## # vemos el número de hojas con la función excel_sheets()
## hojas <- excel_sheets(file)
##
## # número de hojas
## cat("Número de hojas:", length(hojas), "\n")
## # nombres de las hojas
## cat("Nombres de las hojas:", hojas)
##
##
## ## ----columnas de las hojas del dataset en Excel, echo=FALSE-----
## # leemos los datos de la primera hoja
## data <- read_excel(file, sheet = 1)
## cat("Hoja de Data:", colnames(data), "\n")
##
##
## # leemos los datos de la segunda hoja
## peak <- read_excel(file, sheet = 2)
## cat("Hoja de Peak:", colnames(peak))
##
##
## ## ----preparación del objeto SummarizedExperiment-----
## # extraemos los metadatos de las muestras (SampleID, SampleType, Class)
## sample_metadata <- data[, 2:4]
## # extraemos los datos de concentración de metabolitos (columnas M1 a M149) como matriz
## concentration <- t(as.matrix(data[, 5:ncol(data)])) # trasponer con t() para dejar
##                                                    # las concentraciones de metabolitos
##                                                    # en filas
##
## # podríamos renombrar los metabolitos en esta matriz por los Labels de sample_metadata
## # pero vamos a dejarlo así teniendo en cuenta que podemos generar una función para

```

```

## # obtener el nombre del metabolito posteriormente
##
## colnames(concentration) <- paste0(sample_metadata$SampleID,
##                                "_", sample_metadata$Class) # renombramos las columnas por los
##                                # nombres de las muestras con su categoría clínica
##
## # creamos el objeto DataFrame de rowData con los metadatos de los metabolitos de la hoja Peak
## rowData <- DataFrame(
##   Name = peak$Name,
##   Label = peak$Label,
##   Perc_missing = peak$Perc_missing,
##   QC_RSD = peak$QC_RSD
## )
##
## # creamos el objeto data.frame de colData con los metadatos de las muestras
## colData <- as.data.frame(sample_metadata[1:ncol(sample_metadata)])
##
## # transformamos el tipo de dato de Class a factores para análisis con POMA
## colData$Class <- as.factor(colData$Class)
##
## # renombramos la columna SampleID por `Sample name`
## colnames(colData)[colnames(colData) == "SampleID"] <- "Sample name"
##
## # cambiamos el orden de las columnas de colData y nos quedamos sólo con
## colData <- colData[c("Class", "SampleType", "Sample name")]
##
##
## ## ----metadatos del experimento, echo=FALSE-----
## # metadatos del experimento
## experiment_metadata <- list(
##   `Experiment data` = list(
##     `Experimenter name` = "Broadhurst David",
##     `Laboratory` = "University of Alberta",
##     `Contact information` = "d.broadhurst@ecu.edu.au",
##     `Title` = "1H-NMR urinary metabolomic profiling for diagnosis of gastric cancer",
##     `URL` = "https://pubmed.ncbi.nlm.nih.gov/26645240/",
##     `PMIDs` = "26645240",
##     `Abstract` = "Background: Metabolomics has shown promise in gastric cancer (GC) detection. This
##
## Methods: Urine from 43 GC, 40 BN, and 40 matched HE patients was analysed using (1)H nuclear magneti
##
## Results: GC displayed a clear discriminatory biomarker profile; the BN profile overlapped with GC an
##
## Conclusions: GC patients have a distinct urinary metabolite profile. This study shows clinical poten
##   )
## )
##
## ## ----web scraping, cache=TRUE, echo=FALSE-----
## # URL de la página con la tabla datatable de datos de muestra
## url <- "https://www.metabolomicsworkbench.org/data/subject_fetch.php?STUDY_ID=ST001047"
##
## # leemos la página
## webpage <- read_html(url)

```

```

##
## # extraemos y almacenamos la información en un data.frame
## df_samples <- webpage %>%
##   html_nodes(".datatable") %>%
##   html_table() %>%
##   .[[1]]
##
## # fusionamos por la columna Sample name
## colData <- merge(colData, df_samples, by = "Sample name", all.x = TRUE, sort=FALSE)
##
## # reordenamos y seleccionamos las columnas que queremos mantener
## colData <- colData[c("Class", "SampleType", "Sample name", "mb_sample_id", "Batch")]
##
## # añadimos el batch a los nombres de las columnas de concentration
## colnames(concentration) <- paste0(colnames(concentration),
##                                   "_B", colData$Batch) # muestra + cat clínica + batch
##
##
## ## ----construcción del objeto SummarizedExperiment y guardado del binario Rda-----
## # creamos el objeto SummarizedExperiment
## se <- SummarizedExperiment(
##   assays = list(concentration = concentration),
##   rowData = rowData,
##   colData = colData,
##   metadata = experiment_metadata
## )
##
## # guardamos el objeto en un archivo binario Rda
## save(se, file = "SummarizedExperiment.Rda")
##
## # guardamos los metadatos en archivos CSV
## write.csv(colData, "sample_metadata.csv", row.names = FALSE)
## write.csv(rowData, "variable_metadata.csv", row.names = FALSE)
##
## # guardamos los datos en CSV
## write.csv(as.data.frame(concentration), "data_assay_matrix.csv", row.names = TRUE)
##
## # archivo md con metadatos
## # inicializamos el contenido
## contenido_md <- c(
##   "# Resumen de los Metadatos\n",
##   "\n## Metadatos de Muestras\n",
##   knitr::kable(as.data.frame(colData), format = "markdown"),
##   "\n\n## Metadatos de Features\n",
##   knitr::kable(as.data.frame(rowData), format = "markdown")
## )
##
## # escribimos el archivo md
## writeLines(contenido_md, "resumen_metadatos.md")
##
##
## ## ----descarga con metabolomicsWorkbenchR-----
## # opciones disponibles para un contexto de estudio concreto
## # metabolomicsWorkbenchR::context_outputs(context = 'study')

```



```

##
## mwb.summ <- do_query(context = 'study', input_item = 'study_id',
##                      input_value = 'ST001047', output_item = 'summary') # resumen
## mwb.data <- do_query(context = 'study', input_item = 'study_id',
##                      input_value = 'ST001047', output_item = 'data') # datos
## mwb.factors <- do_query(context = 'study', input_item = 'study_id',
##                          input_value = 'ST001047', output_item = 'factors') # colData
## mwb.metabolites <- do_query(context = 'study', input_item = 'study_id',
##                              input_value = 'ST001047', output_item = 'metabolites') # colData
##
##
##
## # no funciona la extracción directa del objeto SummarizedExperiment
## # (Error en SummarizedExperiment(assays = list(X), rowData = VM, colData = SM, :
## #   the rownames and colnames of the supplied assay(s) must be NULL or identical to
## #   those of the SummarizedExperiment object (or derivative) to construct)
##
## # mwb.se <- do_query(context = 'study', input_item = 'study_id',
## #                     input_value = 'ST001047', output_item = 'SummarizedExperiment')
##
## columns_to_select <- names(mwb.data$AN001711)[c(8:ncol(mwb.data$AN001711))]
## assay.data <- as.data.frame(subset(mwb.data$AN001711, select = columns_to_select))
## rownames(assay.data) <- mwb.data$AN001711$metabolite_name
##
## # mwb.factors no dispone de las muestras QC. Este desacoplamiento produce el problema en
## # la descarga directa del SummarizedExperiment con do_query()
## qc.factors <- data.frame(
##   "study_id" = rep("ST001047", 17),
##   "local_sample_id" = c("sample_1", "sample_10", "sample_100", "sample_109", "sample_118", "sample_1",
##                         "sample_140", "sample_19", "sample_28", "sample_37", "sample_46", "sample_55",
##                         "sample_82", "sample_91"),
##   "sample_source" = rep("Urine", 17),
##   "mb_sample_id" = c("SA070439", "SA070447", "SA070437", "SA070445", "SA070435", "SA070443", "SA0704",
## "SA070434", "SA070432", "SA070433", "SA070444", "SA070442", "SA070440", "SA070441", "SA070438"),
##   "raw_data" = rep("", 17),
##   "subject_type" = rep(NA, 17),
##   "Sample_Type" = rep("QC", 17)
## )
##
## # unimos en un único data.frame los datos
## factors <- rbind(qc.factors, mwb.factors$ST001047)
##
##
## # creamos el objeto SummarizedExperiment
## se2 <- SummarizedExperiment(
##   assays = list(concentration = assay.data),
##   rowData = mwb.metabolites,
##   colData = factors,
##   metadata = mwb.summ
## )
##
## se2
##

```

```

## # guardamos el objeto en un archivo binario Rda
## save(se, file = "SummarizedExperiment_from_MWB.Rda")
##
##
## ## ----datos de concentraciones, echo=FALSE-----
## concentration.data <- assay(se)
##
##
## ## ----estadísticos resumen datos sin normalizar-----
## summary.concentration.data <- data.frame(
##   Mean = apply(concentration.data, 1, mean, na.rm=TRUE), # media
##   Median = apply(concentration.data, 1, median, na.rm=TRUE), # mediana
##   SD = apply(concentration.data, 1, sd, na.rm=TRUE), # desviación estándar
##   Min = apply(concentration.data, 1, min, na.rm=TRUE), # mínimo
##   Max = apply(concentration.data, 1, max, na.rm=TRUE), # máximo
##   Q1 = apply(concentration.data, 1, quantile, probs = 0.25, na.rm=TRUE), # 1er cuartil
##   Q3 = apply(concentration.data, 1, quantile, probs = 0.75, na.rm=TRUE), # 3er cuartil
##   IQR = apply(concentration.data, 1, IQR, na.rm=TRUE), # rango intercuartílico
##   perc_NA = apply(concentration.data, 1, function(x){ # porcentaje de missing values
##     sum(is.na(x)/length(x)*100)
##   })
## )
##
## head(summary.concentration.data, 10) # sólo mostramos las 10 primeras filas
##
##
## ## ----creación de boxplots de metabolitos, warning=FALSE, echo=FALSE-----
## crear_boxplot_metabolitos_pdf <- function(se_object, output_file = "metabolitos_boxplots.pdf") {
##   # extraer datos de concentración y metadatos
##   assay_data <- assay(se_object)
##   sample_metadata <- colData(se_object)
##   metabolitos <- rownames(assay_data)
##
##   # crear un archivo PDF para guardar los gráficos
##   pdf(output_file, width = 10, height = 10)
##
##   opt <- par(mfrow = c(3, 3))
##   # generar boxplots para cada metabolito
##   for (metabolito in metabolitos) {
##     # extraer los valores del metabolito específico y el grupo de muestra correspondiente
##     metabolito_data <- assay_data[metabolito, ]
##     grupo_data <- sample_metadata$Class # grupo a partir de Class
##
##     # crear boxplot del metabolito por grupo de muestra
##     boxplot(metabolito_data ~ grupo_data,
##             main = paste("Distribución de", metabolito, "por grupo"),
##             xlab = "Grupo",
##             ylab = "Concentración",
##             col = rainbow(length(unique(grupo_data))),
##             notch = TRUE,
##             las = 2) # etiquetas del eje X giradas
##   }
##   par(opt)
##   # cerrar el archivo PDF

```

```

## dev.off()
## message("Gráficos boxplot guardados en ", output_file)
## }
##
## # llamamos a la función con el objeto SummarizedExperiment
## crear_boxplot_metabolitos_pdf(se)
##
##
## ## ----creación de histogramas de metabolitos, echo=FALSE, warning=FALSE-----
## crear_histograma_metabolitos_pdf <- function(data, output_file = "metabolitos_histograms.pdf") {
##
##   pdf(output_file)
##
##   opt <- par(mfrow = c(3, 3))
##
##   # genera histogramas para cada metabolito
##   for (i in 1:ncol(data)) {
##     hist(data[, i], main = colnames(data)[i], xlab = "Valores", ylab = "Frecuencia")
##   }
##
##   par(opt)
##
##   dev.off()
##   message("Gráficos de histogramas guardados en ", output_file)
## }
##
## crear_histograma_metabolitos_pdf(t(assay(se)))
##
##
## ## ----M138 boxplot, warning=FALSE, echo=FALSE, fig.height=3.5-----
## sample_metadata <- colData(se)
## metabolito.name <- rowData[rowData$Name=="M138",]$Label
## boxplot(concentration.data["M138", ] ~ sample_metadata$Class,
##         notch = TRUE,
##         col=rainbow(length(unique(sample_metadata$Class))),
##         main = paste("Distribución de", metabolito.name, "por grupo"))
##
##
## ## ----M8 boxplot, warning=FALSE, echo=FALSE, fig.height=3.5-----
## sample_metadata <- colData(se)
## metabolito.name <- rowData[rowData$Name=="M8",]$Label
## boxplot(concentration.data["M8", ] ~ sample_metadata$Class,
##         notch = TRUE,
##         col=rainbow(length(unique(sample_metadata$Class))),
##         main = paste("Distribución de", metabolito.name, "por grupo"))
##
##
## ## ----filtrado de datos-----
## library("POMA")
##
## # recogemos los rowData de los metabolitos
## rowData_se <- rowData(se)
##
## # filtramos por QC_RSD < 20

```

```

## filtered_indices <- which(rowData_se$QC_RSD < 20)
##
## # creamos un nuevo objeto SummarizedExperiment con los metabolitos filtrados
## se_filtered <- se[filtered_indices, ]
##
## imputed <- PomaImpute(se_filtered, zeros_as_na = FALSE,
##                        remove_na = TRUE, method = "knn", cutoff = 10)
##
## # ver el número de metabolitos antes y después del filtrado
## cat("Número de metabolitos originales:", nrow(se), "\n")
## cat("Número de metabolitos filtrados:", nrow(imputed), "\n")
##
##
## ## ----función para obtener el label-----
## # función para obtener el label de un metabolito específico
## obtener_label_metabolito <- function(metabolito, se) {
##   # extraemos la fila donde se encuentra el nombre del metabolito en el objeto original
##   row <- rowData(se)[rowData(se)$Name == metabolito,]
##   # extraemos el Label correspondiente para devolverlo
##   label <- row$Label
##   return(label)
## }
##
##
## ## ----normalizado de datos-----
## normalized <- PomaNorm(imputed, method = "log_scaling")
## normalized
##
##
## ## ----outliers, fig.height=3.5-----
## PomaOutliers(normalized)
##
##
## ## ----PomaBoxplots, echo=FALSE, fig.height=8-----
## a <- PomaBoxplots(imputed,
##                   x = "samples") +
##   ggplot2::ggtitle("Sin normalizar - Muestras")+
##   ggplot2::theme(axis.text.x = ggplot2::element_blank())
## b <- PomaBoxplots(normalized,
##                   x = "samples") +
##   ggplot2::ggtitle("Normalizados - Muestras")+
##   ggplot2::theme(axis.text.x = ggplot2::element_blank())
##
## c <- PomaBoxplots(imputed,
##                   x = "features") +
##   ggplot2::ggtitle("Sin normalizar - Metabolitos")+
##   ggplot2::theme(axis.text.x = ggplot2::element_blank())
##
## d <- PomaBoxplots(normalized,
##                   x = "features") +
##   ggplot2::ggtitle("Normalizados - Metabolitos")+
##   ggplot2::theme(axis.text.x = ggplot2::element_blank())
##
## # mostramos los gráficos

```

```

## (a + b) / (c + d) + plot_layout(heights = c(1, 1))
##
##
## ## ----PomaDensity, echo=FALSE, fig.height=6-----
## a <- PomaDensity(imputed,
##                   x = "features",
##                   theme_params = list(legend_title = FALSE, legend_position = "none")) +
##   ggplot2::ggtitle("Sin normalizar - Metabolitos")
##
## b <- PomaDensity(normalized,
##                   x = "features",
##                   theme_params = list(legend_title = FALSE, legend_position = "none")) +
##   ggplot2::ggtitle("Normalizados - Metabolitos")
##
## c <- PomaDensity(imputed,
##                   x = "samples") +
##   ggplot2::ggtitle("Sin normalizar - Muestras")
##
## d <- PomaDensity(normalized,
##                   x = "samples") +
##   ggplot2::ggtitle("Normalizados - Muestras")
##
## # mostramos los gráficos
## (a + b) / (c + d) + plot_layout(heights = c(1, 1))
##
##
## ## ----metabolito con valores atípicos, echo=FALSE-----
## max_index <- which(assay(normalized) == max(assay(normalized)), arr.ind = TRUE)
## metabolito.max.name <- rownames(max_index)
##
##
## ## ----eliminación de QC, echo=FALSE-----
## # filtramos las muestras que no son QC
## normalized <- normalized[, colData(normalized)$SampleType != "QC"]
##
##
## ## ----PomaUnivariate-----
## PomaUnivariate(normalized[, normalized$class %in% c("GC", "HE")], method = "ttest",
##                 var_equal = FALSE, adjust = "fdr") # test t de Welch
##
##
## ## ----PomaVolcano, fig.height=3-----
## PomaUnivariate(normalized[, normalized$class %in% c("GC", "HE")],
##                 method = "ttest", var_equal = FALSE, adjust = "fdr") %>%
##   magrittr::extract2("result") %>%
##   dplyr::select(feature, fold_change, pvalue) %>%
##   PomaVolcano(labels=TRUE)
##
##
## ## ----PomaPCA-----
## pca <- PomaPCA(normalized, ellipse = TRUE, labels=TRUE, load_length = 1.1)
## pca$biplot
##
##

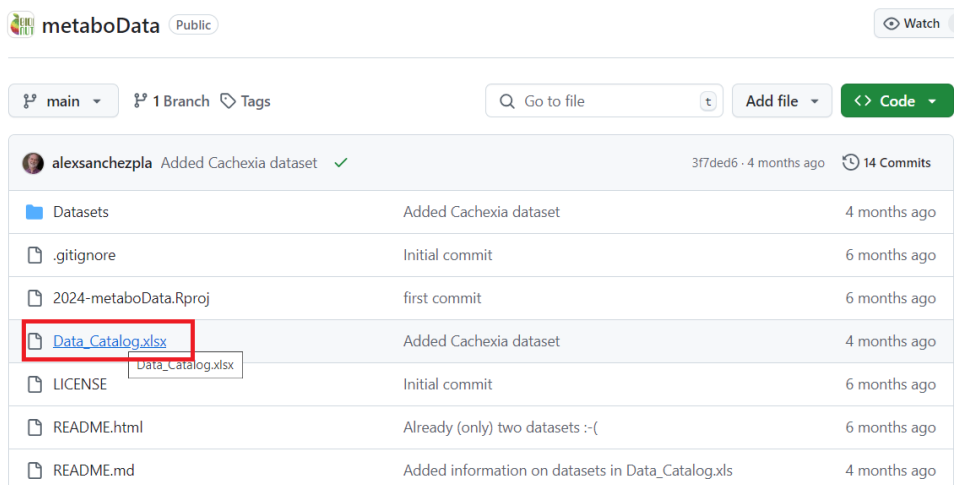
```

```

## ## ----cargas PCA, echo=FALSE-----
## pc2_values <- pca$loadings$PC2
##
## # índices de los 5 valores más grandes en valor absoluto
## top_indices <- order(abs(pc2_values), decreasing = TRUE)[1:5]
##
## # valores y features correspondientes
## top_values <- pc2_values[top_indices]
## top_features <- pca$loadings$feature[top_indices]
##
## # DataFrame para mostrar los resultados
## top_loads <- data.frame(
##   Feature = top_features,
##   Value = top_values,
##   AbsValue = abs(top_values)
## )
##
##
## ## ----cluster jerárquico de muestras, fig.asp=0.85, fig.align="center"-----
## dist.matrix <- dist(t(assay(normalized)))
## hc_res <- hclust(dist.matrix, method = "ward.D2")
## sub_grp <- cutree(hc_res, k=3)
##
## plot(hc_res, hang = -1, cex = 0.45)
## rect.hclust(hc_res, k=3, border=2:(3+1))
##
##
## ## ----mostrar_codigo, echo=FALSE-----
## # leemos el archivo generado con purl()
## codigo <- readLines("PEC1.R")
## # mostramos el código
## cat(codigo, sep = "\n")

```

Anexo II - Imágenes



File	Commit Message	Time
Datasets	Added Cachexia dataset	4 months ago
.gitignore	Initial commit	6 months ago
2024-metaboData.Rproj	first commit	6 months ago
Data_Catalog.xlsx	Added Cachexia dataset	4 months ago
LICENSE	Initial commit	6 months ago
README.html	Already (only) two datasets :-{	6 months ago
README.md	Added information on datasets in Data_Catalog.xls	4 months ago

Figure 3: repositorio

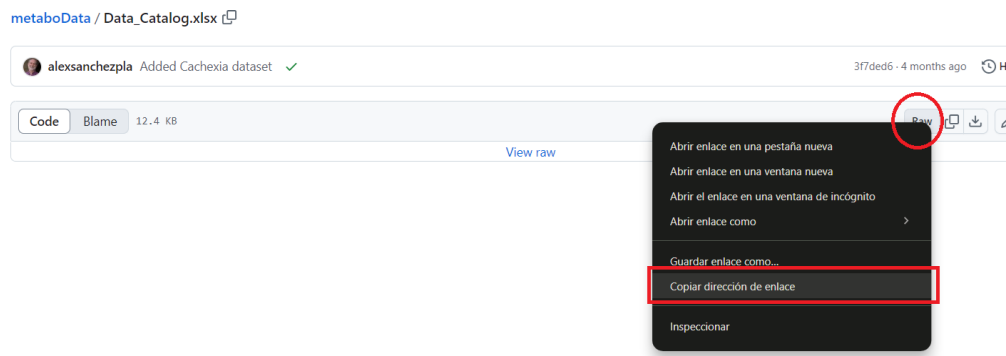


Figure 4: Data_Catalog.xlsx

'metaboData

Introduction

A repository with a few public metabolomics datasets borrowed from different public open sources.

While we don't come out with a better option the repository will be "folder-based". That is:

- each dataset is contained in a sub-folder of the "datasets" folder named with a short-descriptive name with no spaces and no special codes.
- Each folder contains a "description.md" document, written in markdown with information about the dataset.
- Eventually a template will be provided to fill the description.

The spreadsheet DataC_atalog.xls is an attempt to provide a quick description of each dataset

Figure 5: Introduction

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk (*)

Owner *

GsusML84

Repository name *

New repository name must not be blank

Great repository names are short and memorable. Need inspiration? How about [glowing-lamp](#) ?

Description (optional)

☒ Public

Anyone on the internet can see this repository. You choose who can commit.

☐ Private

You choose who can see and commit to this repository.

Initialize this repository with:

☐ Add a README file

This is where you can write a long description for your project. [Learn more about READMEs.](#)

Figure 6: Creación repositorio

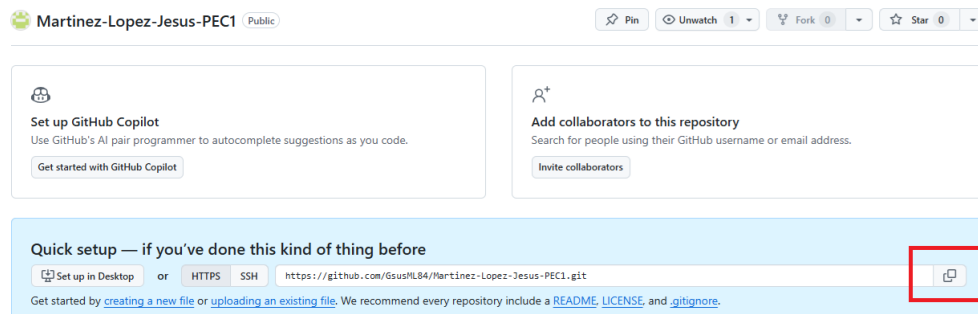


Figure 7: Obtención enlace git

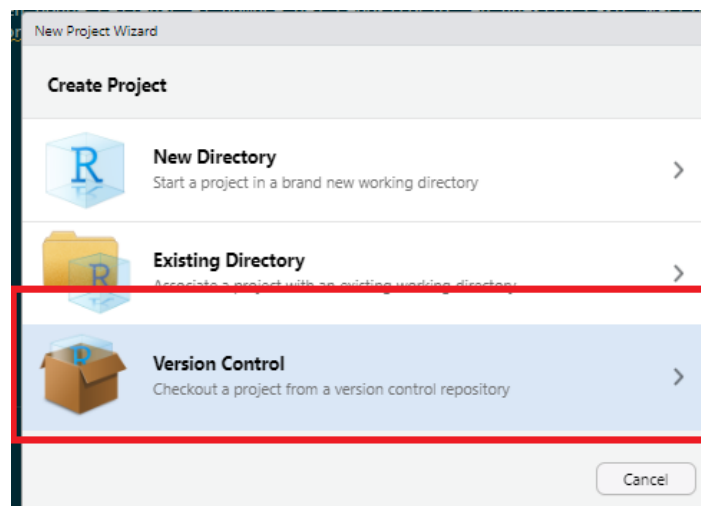


Figure 8: Obtención enlace git

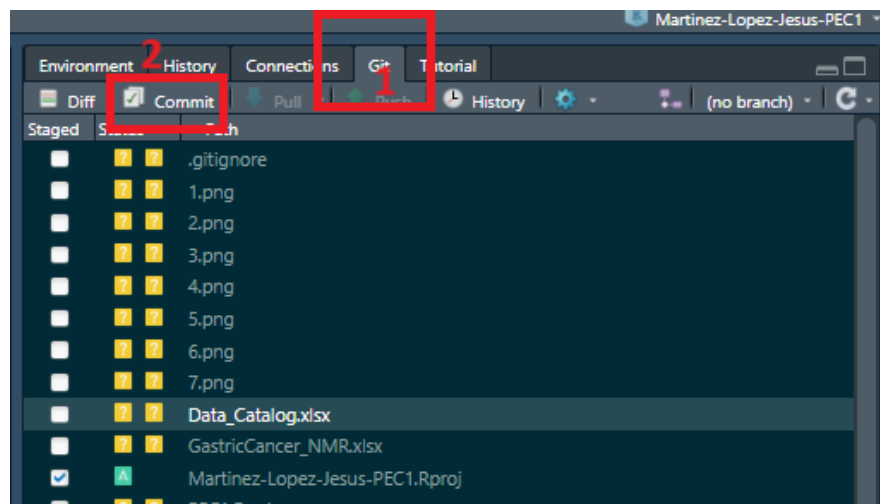


Figure 9: Pestaña Git RStudio

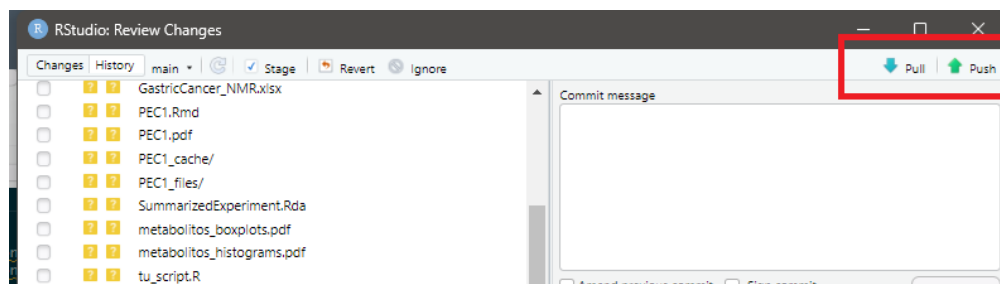


Figure 10: Botones Pull & Push

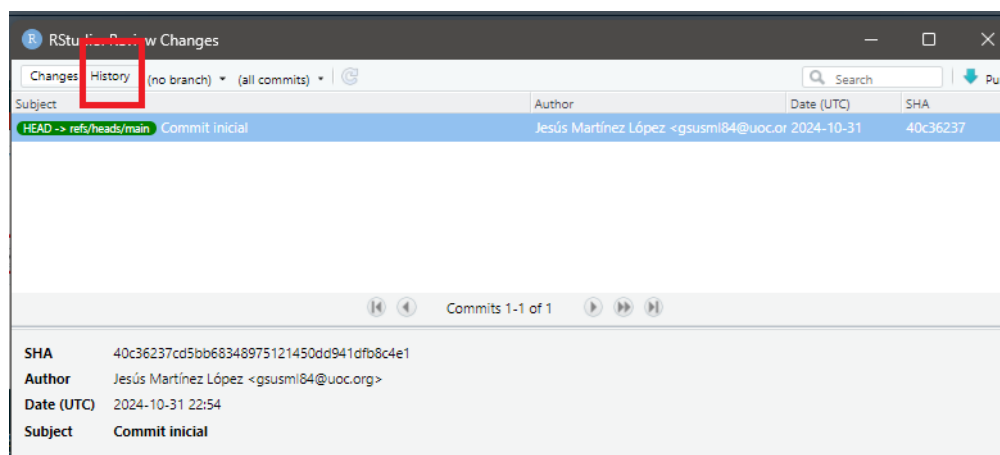


Figure 11: Histórico control de versiones