Universitat Oberta de Catalunya uoc.edu

Clinical Bioinformatics Area

Análisis de pathways y redes de interacción génica en enfermedades complejas

María Inmaculada Álamo Álvarez
Máster en Bioinformática y Bioestadística
Área 2, subárea 7: Estudios de asociación en genómica del cáncer e integración de datos ómicos

Tutores Área de Bioinformática Clínica:

María Peña Chilet
Investigadora posdoctoral
Joaquín Dopazo
Director del Área

Jaime Sastre Tomàs
Profesor Asociado
Carles Ventura Royo
Profesor responsable de la asignatura

junio de 2020

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Análisis de pathways y redes de interacción génica en enfermedades complejas*
Nombre del autor:	*María Inmaculada Álamo Álvarez*
Nombre del consultor/a:	*Jaime Sastre Tomás* *María Peña Chilet* *Joaquín Dopazo Blázquez*
Nombre del PRA:	*Carles Ventura Royo*
Fecha de entrega (mm/aaaa):	06/2020
Titulación:	*Máster Universitario en Bioinformática y Bioestadística*
Área del Trabajo Final:	*Área 2, subárea 7: Estudios de asociación en genómica del cáncer e integración de datos ómicos*
Idioma del trabajo:	***Castellano***
Palabras clave	*complex disease, mechanistic pathways, GWAS*

Resumen del Trabajo (máximo 250 palabras): *Con la finalidad, contexto de aplicación, metodología, resultados y conclusiones del trabajo.*

En este trabajo se ha creado un flujo de trabajo y una serie de scripts que sirven para asistir en la generación de mapas de enfermedad (mapa de enriquecimiento funcional y mapa de interacciones entre proteínas asociadas) partiendo de datos de estadísticas globales que provienen de estudios de asociación de genoma completo (GWAS). Se seleccionó el asma como ejemplo de enfermedad compleja para realizar una prueba de concepto. Se han obtenido datos de asociaciones de variantes de la base de datos GWAS Catalog, se ha realizado un análisis de enriquecimiento funcional con g:Profiler y se han visualizado las redes obtenidas en Cytoscape utilizando los plugins stringApp y del pipeline de EnrichmentMap.

El código utilizado (R y bash) está disponible en https://github.com/mialaalv/pathWAS.

Abstract (in English, 250 words or less):

This project consists of a pipeline and a series of scripts designed to assist in the generation of disease maps (functional enrichment maps and protein-protein interaction maps) based on global statistical data from genome-wide association studies (GWAS). Asthma was selected as an example of complex disease to conduct a proof of concept. Data on variant associations were obtained from the GWAS Catalog database, a functional enrichment analysis was performed with g:Profiler, and the networks obtained were displayed in Cytoscape using the stringApp and EnrichmentMap pipeline plugins. The code used (R and bash) is available at https://github.com/mialaalv/pathWAS.

Índice

Lista de figuras

1. Introducción

1.1. Contexto y justificación del Trabajo

1.1.1. Enfermedades complejas

A grandes rasgos, una enfermedad compleja es aquella cuyas características complican la identificación de los factores que la causan. Presentan el mismo comportamiento genético que rasgos continuos como el color de la piel o la estatura, que son caracteres cuantitativos (1). Generalmente se categorizan según unas características superficiales y clínicamente observables y, a nivel molecular, son de origen poligénico y surgen de la interacción de muchos factores genéticos entre sí y con el medio ambiente (2).

Identificar los factores que determinan la aparición de la enfermedad y caracterizar la contribución de estos a su desarrollo es todo un reto porque pueden estar enmascarados por otros factores. Además, algunas enfermedades serán heterogéneas y comprenderán diversos subtipos, o incluso abarcarán varios trastornos fenotípicamente similares pero cada uno con sus propias causas moleculares subyacentes (2,3).

1.1.2. Estudios de asociación genética

Los estudios de asociación alélica, como los GWAS (*Genome Wide Association Studies*, estudios de asociación de genoma completo) tienen como objetivo descubrir asociaciones estadísticas entre alelos utilizados como marcadores genéticos y alelos que producen susceptibilidad al fenotipo estudiado. Son un método ampliamente aceptado y eficaz para identificar los *loci* genéticos asociados a enfermedades o rasgos comunes (1). Los GWAS consisten en el análisis de cientos de miles de polimorfismos en todo el genoma en grandes cohortes de individuos para identificar las variantes asociadas con el rasgo de interés. La mayor parte de las variantes identificadas por los GWAS no son causales, sino que señalan una región con la que existe desequilibrio de ligamiento y que contiene una o más variantes que sí son funcionales. Se habla de desequilibrio de ligamiento entre dos variantes cercanas cuando los alelos de polimorfismos vecinos (en el mismo cromosoma) aparecen juntos en una población con mayor frecuencia que si no estuvieran vinculados (4).

El número de estudios de asociación de genoma completo ha aumentado mucho desde su aparición, y se pueden encontrar numerosos estudios y consorcios de GWAS. Esto hace que haya muchos datos disponibles y tamaños muestrales grandes. Además, es una tecnología bastante más asequible que, por ejemplo, las basadas en secuenciación del genoma completo. Sin embargo, a pesar del incremento en el tamaño muestral de GWAS, las asociaciones identificadas no explican el total de la heredabilidad de los rasgos estudiados (5). Por esto, se postula como herramienta mucho más potente el estudio a nivel de gen y de grupos de genes (6).

Los grandes volúmenes de datos disponibles dificulta su interpretabilidad, requiriéndose métodos cada vez más sofisticados para transformar los datos en conocimiento biológico. Debemos dar con una forma efectiva de explotarlos y ponerlos al servicio de las necesidades de la investigación.

1.1.3. GWAS Catalog

Es posible encontrar diversas herramientas que recopilan estudios de GWAS y sus resultados. Un ejemplo es GWAS Catalog. GWAS Catalog es un catálogo, de acceso público y curado manualmente, que contiene gran cantidad de estudios GWAS publicados y sus resultados, desarrollado por el NHGRI y EMBL-EBI.

A día 13 de junio de 2020, contiene datos y 187403 asociaciones SNP-atributo de 4580 estudios. Los datos de GWAS Catalog están mapeados al ensamblaje GRCh38.p13 del genoma humano y la versión 153 de la base de datos de polimorfismos de un solo nucleótido dbSNP (*The Single Nucleotide Polymorphism database*).

Hay diversas formas de acceder al contenido del catálogo:

- A través de la interfaz web en https://www.ebi.ac.uk/gwas.
- Mediante un volcado de base de datos en formatos TSV o RDF/OWL desde https://www.ebi.ac.uk/gwas/docs/file-downloads.
- De forma programática a través de su API REST, en https://www.ebi.ac.uk/gwas/rest/docs/api. Una API es un herramienta que permite el intercambio de información con su servidor sin necesidad de conocer en detalle cómo está implementada la base de datos. REST (REpresentational State Transfer) es un tipo de arquitectura basada en HTTP (Hypertext Transfer Protocol), y es la que la mayoría de servicios utilizan para implementar sus APIs (7). Puede utilizarse en cualquier dispositivo capaz de entender las peticiones HTTP, en las que una URL representa el recurso sobre el que se va a realizar determinada operación (8).

1.1.4. STRING

La base de datos STRING pretende recopilar, evaluar e integrar todas las fuentes de información disponibles públicamente de interacción entre proteínas, y complementarlas con predicciones computacionales. Su intención es crear una red mundial amplia y objetiva, que incluya tanto las interacciones directas , que corresponden a interacciones físicas, como las indirectas, que corresponden a interacciones funcionales (9). El sitio web de STRING se puede visitar en https://string-db.org/. También existen *plugins* que permiten el acceso a esta información desde otros entornos, como R o Cytoscape.

1.1.5. Módulos funcionales y mapas de enfermedad

Para favorecer un mejor entendimiento de las funciones celulares de un organismo a nivel de sistema, y permitiéndonos también dar explicaciones mecanísticas a trastornos complejos, se han desarrollado diversas iniciativas de recopilación de conocimiento sobre módulos funcionales, grandes redes de interacción molecular y

rutas en forma de representación gráfica de las interacciones funcionales entre ADN, ARN y proteínas (rutas biológicas). Algunos ejemplos de estas iniciativas son bases de datos como KEGG (2), Reactome (3) o Wikipathways (4), entre otras.

Una manera de abordar el estudio de las enfermedades complejas se basa en la creación de mapas de enfermedad completos que permitan su análisis mecanístico y funcional. En este sentido cabe destacar la iniciativa internacional Disease Maps (9). En medicina de sistemas, un mapa de enfermedad es una representación de los mecanismos moleculares que dan lugar a una enfermedad, un modelo conceptual detallado de rutas de señalización, metabólicas y reguladoras interconectadas. Se utilizan como recursos pedagógicos, pero también como base de modelos matemáticos predictores para la práctica clínica o para generar hipótesis de investigación (9,10).

Cabe esperar un mejor resultado a la hora de utilizar un enfoque mecanístico cuando la enfermedad que se estudie esté altamente relacionada con procesos de señalización celular. Por eso, para este trabajo se propone usar como prueba de concepto enfermedades complejas que cumplan esta premisa. Para ello, se usará como referencia la lista de enfermedades que *Berridge et al.* (11) sugiere que están directamente relacionadas con rutas de señalización, y se elige como enfermedad modelo para esta prueba de concepto para este trabajo el asma.

1.2. Objetivos del Trabajo

1. Búsqueda de datos de asociación genómica a enfermedades complejas en bases de datos públicas como GEO (12), SRA (13), o GWAS Catalog (14).
2. Seleccionar una enfermedad compleja con suficiente cantidad de datos para utilizar como prueba de concepto.
3. Obtener una lista de genes candidatos de enfermedad (*disease genes*, DG) a partir de los datos de asociación genómica utilizados. Se usará el software MAGMA (15) para transformar las asociaciones variante-enfermedad en asociaciones gen-enfermedad.
4. A partir de la lista de genes de enfermedad, generar una red de interacción entre estas proteínas con Cytoscape. Obtener las interacciones entre proteínas de la base de datos STRING (16).
5. Realizar un análisis de enriquecimiento funcional con g:Profiler (16).
6. Visualizar en Cytoscape los resultados del enriquecimiento funcional utilizando el pipeline de EnrichmentMap para Cytoscape (17).
7. Automatizar el proceso y generar una herramienta que pueda asistir en la creación de mapas de enfermedad para su posterior análisis.

1.1. Enfoque y método seguido

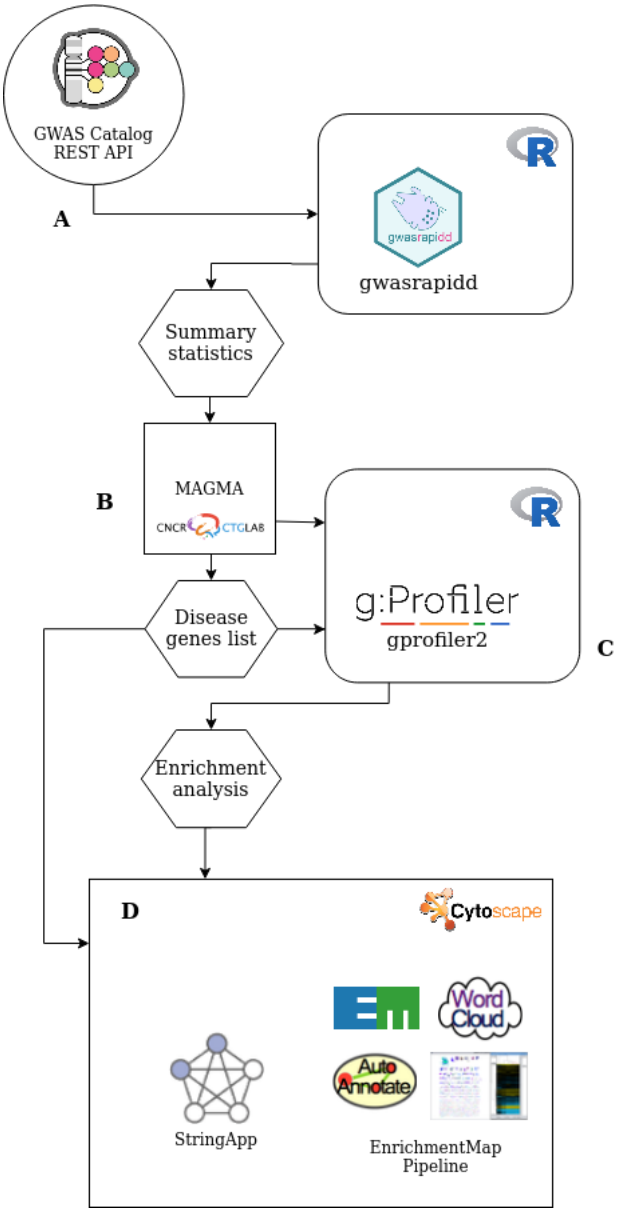

Figura 1. Esquema del flujo de trabajo propuesto. La herramienta sirve para asistir en la generación de mapas de enfermedad a partir de estadísticas globales que provienen de estudios de asociación de genoma completo. **(A)** Partiendo del nombre de una enfermedad o *trait* de GWAS Catalog se obtienen las asociaciones genéticas reportadas disponibles en la base de datos y **(B)** se realiza un análisis genético con la herramienta MAGMA para obtener una lista de genes asociados de forma significativa a la enfermedad. **(C)** De esta lista de genes se hace un análisis de enriquecimiento con g:Profiler, y **(D)** tanto la lista de genes como los resultados del enriquecimiento se utilizan para generar mapas de enfermedad en Cytoscape con los plugins de stringApp y los del *workflow* de EnrichmentMap.

1.2. Planificación del Trabajo

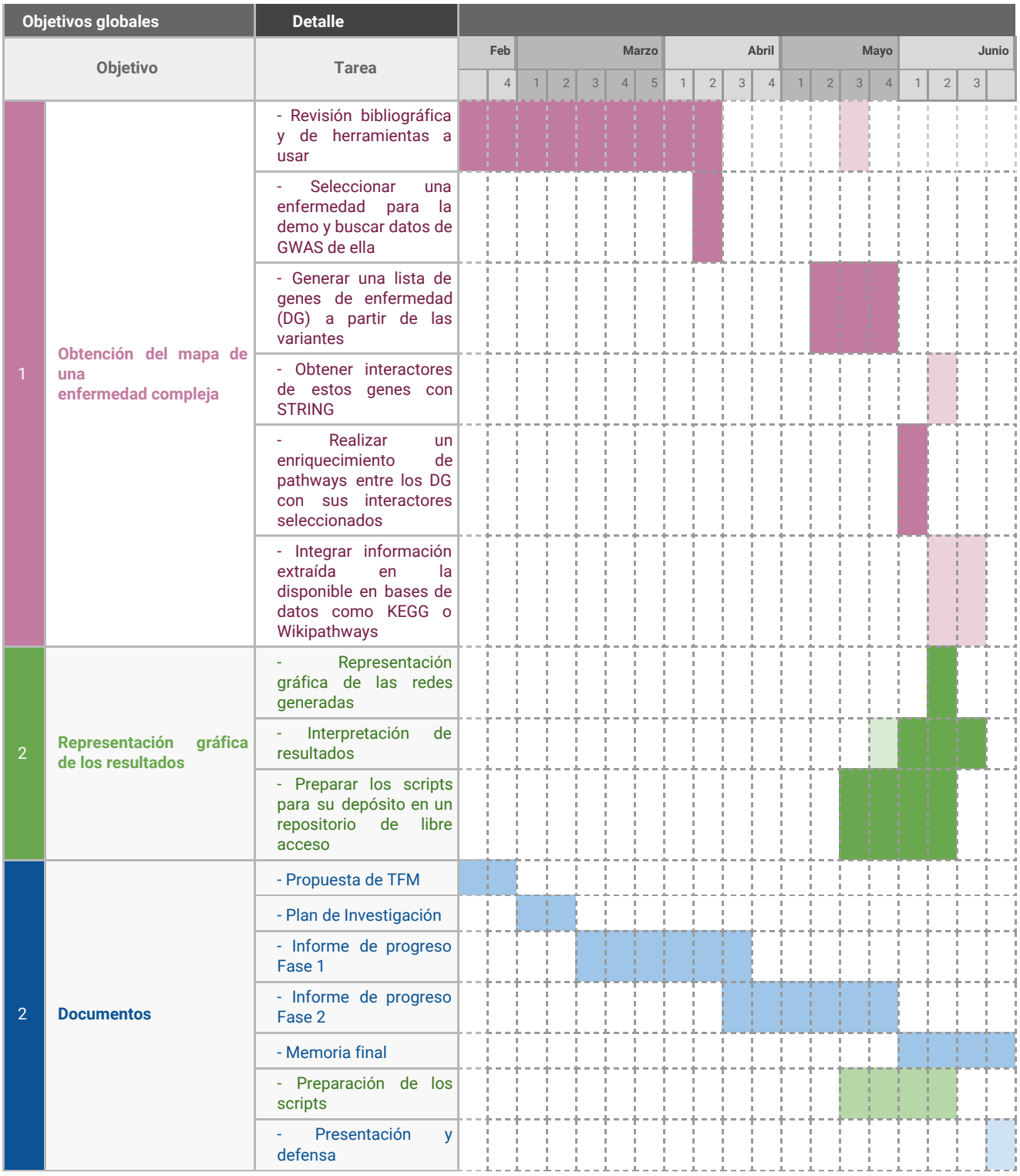

Figura 2. Diagrama Gantt del desarrollo del proyecto.

1.3. Breve sumario de productos obtenidos

Se ha obtenido una serie de scripts en R y bash que sirven para asistir a la generación de mapas de interacciones de proteínas asociadas a una enfermedad compleja y a la obtención y visualización de los resultados de un enriquecimiento funcional de estas. Los mapas se obtienen a partir de estadísticas globales que provienen de estudios de asociación de genoma completo. Partiendo del nombre de una enfermedad o *trait* de GWAS Catalog se obtienen las asociaciones genéticas reportadas disponibles en la base de datos y se realiza un análisis genético con la herramienta MAGMA para obtener una lista de genes asociados de forma significativa a la enfermedad. De esta lista de genes se hace un análisis de enriquecimiento con g:Profiler, y tanto la lista de genes como los resultados del enriquecimiento se utilizan para generar mapas de enfermedad en Cytoscape con los *plugins* de stringApp y los del *pipeline* de EnrichmentMap.

1.4. Breve descripción de los otros capítulos de la memoria

En el resto de capítulos de la memoria se introducen las herramientas utilizadas y los procedimientos seguidos para el desarrollo del trabajo, se realiza un recorrido por los pasos del pipeline propuesto para generar los mapas de enfermedad y se muestra un ejemplo de redes obtenidas con el asma como prueba de concepto.

2. Resto de capítulos

2.1. Pipeline propuesto

0) Preparación del entorno

Requisitos:

- Carpeta de scripts anexos, también disponibles en https://github.com/mialaalv/pathWAS.
- R (versión 4.0.0) (18)
- Entorno de programación de RStudio (versión 1.3.959).
- MAGMA versión 1.07 (descargado automáticamente con el script anexo).

  Para el funcionamiento de MAGMA (descargados automáticamente con el script anexo):

  - Archivo de definiciones de genes NCBI 37.3
  - Archivo de datos de GWAS de referencia para considerar el desequilibrio de ligamiento entre SNPs. Se descarga y usa automáticamente un panel de los 1000 genomas con población europea.
- Al menos 8 Gb de RAM para la visualización de las redes.
- Java 11
- Cytoscape 3.8.0. Plugins:
  - stringApp
  - EnrichmentMap Pipeline Collection

Se ha generado un script principal en bash que llama al resto de scripts (main.sh) y otros que se encargarán de preparar algunas dependencias del pipeline (magma_setup.sh y R_dependencies.R).

El flujo de trabajo propuesto se divide en 4 partes, como se puede ver en la Figura 1.

A) Obtención de las asociaciones genéticas con la enfermedad/rasgo de GWAS Catalog.

Partiendo del nombre de una enfermedad o característica (*trait*) de la ontología de GWAS Catalog (en este ejemplo el asma, con EFO_0000270, junto con sus términos hijos), se obtienen las asociaciones genéticas reportadas disponibles en la base de datos.

Se realiza desde R con el paquete *gwasrapidd* (19). *Gwasrapidd* permite acceder a los datos de GWAS Catalog a través de su API REST. La base de datos está organizada de forma que hay tres tablas principales: asociaciones variante-enfermedad, variantes y estudio del que provienen las asociaciones.

Debido a que en GWAS Catalog las posiciones genómicas están de acuerdo al ensamblaje GRCh38 y las necesitamos en GRCh37, las coordenadas se convierten utilizando la función *liftOver* del paquete *rtracklayer*.

- ❏ Script: gwas_data_fetching.R
- ❏ Input: el nombre de una enfermedad o característica de GWAS catalog. En el ejemplo se usa el asma (EFO_0000270) y sus términos hijos (*atopic asthma, Chronic Obstructive Asthma, adult onset asthma, childhood onset asthma*)
- ❏ Output: tabla de asociaciones variante-enfermedad con el nombre del SNP en la primera columna, en la segunda y tercera sus coordenadas genómicas (en el mismo ensamblaje que las referencias, en este caso GRCh37), en la tercera el p-valor de la asociación SNP-enfermedad y en la cuarta el número de participantes en el estudio del que proviene la asociación.

SNP	CHR	BP	P	NOBS
rs350729	2	52983773	2E-10	724
rs6924808	6	98358575	7E-16	120

B) Análisis genético con MAGMA.

MAGMA (*Multi-marker Analysis of GenoMic Annotation*) es un software dedicado al análisis de datos de GWAS a nivel de gen o de conjuntos de genes (20). El análisis de MAGMA a nivel de gen, que es el utilizado en este trabajo, está basado en un modelo de regresión lineal múltiple de componentes principales, y usa un test de Fisher para computar los p-valores de los genes. Este modelo proyecta la matriz de SNPs de un gen sobre sus componentes principales (PC), eliminando las componentes con valores propios muy pequeños, y luego utiliza esas componentes principales como predictores del fenotipo observado en el modelo de regresión lineal. Esto supone la eliminación de parámetros redundantes, mejora la potencia y garantiza que el modelo sea aplicable en presencia de SNPs con alta colinealidad. Por defecto, sólo se elimina el 0,1% de la varianza en la matriz de datos de los SNPs (20).

El análisis con MAGMA desde p-valores de GWAS consta de dos pasos:

- Anotación: se mapean las posiciones de los SNPs a cada uno de los genes de una referencia proporcionada, en este caso, los 19427 genes codificantes de proteínas incluidos en las definiciones de genes NCBI 37.3.

  Comando utilizado:

```
./magma/magma --annotate \
--snp-loc  data/GWAS_data/EFO_0000270/EFO_0000270_GWAS_summary_grch37_forMAGMA.tsv \
```

```
--gene-loc magma/res/NCBI37.3/NCBI37.3.gene.loc \
--out results/EFO_0000270
```

- Análisis de genes: computa los p-valores de asociación gen-enfermedad utilizando los p-valores proporcionados para cada SNP, los genes anotados con sus correspondientes SNPs y el número de individuos del estudio del que proviene cada asociación SNP-enfermedad. Se utiliza el modo por defecto.

  Comando utilizado:

```
./magma/magma --bfile ./magma/res/g1000_eur/g1000_eur \
--pval data/GWAS_data/EFO_0000270/EFO_0000270_GWAS_summary_grch37_forMAGMA.tsv \
ncol=NOBS --gene-annot results/EFO_0000270.genes.annot --out results/EFO_0000270
```

También debe realizarse un ajuste de los p-valores obtenidos. MAGMA devuelve unos p-valores no ajustados, así que se deben ajustar posteriormente para corregir por tests múltiples y reducir el número de falsos descubrimientos. En este caso, se ajustan en R con la función *p.adjust* del paquete *stats*, utilizando el método FDR (21). Posteriormente, se filtran los genes que tienen un p-valor ajustado mayor que 0.05 y aquellos con un Z-score menor que la media menos 0.9 veces la desviación estándar de la muestra. En la Figura 3 están representados los p-valores ajustados y los Z scores del output de MAGMA, y una línea horizontal en el punto donde se realiza el filtrado. Este último filtrado se realiza por ser un poco conservadores y descartar las peores asociaciones gen-enfermedad obtenidas.

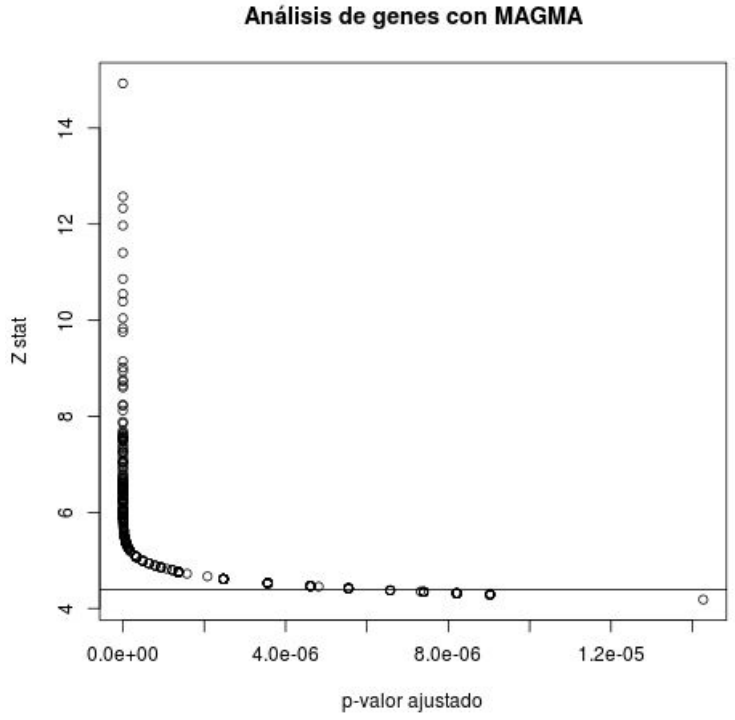

Figura 3. Gráfica de p-valor ajustado vs. Z score de las asociaciones gen-enfermedad que devuelve MAGMA. La línea horizontal muestra el punto de corte que se toma para eliminar las peores asociaciones obtenidas y reducir el número de posibles falsos positivos.

- ❏ Input: tabla de asociaciones variante-enfermedad generada en el apartado anterior.
- ❏ Output: tabla de asociaciones gen-enfermedad, von Z-score, p-valor y p-valor ajustado. Lista de genes de enfermedad.

C) Análisis de enriquecimiento funcional con g:Profiler

g:Profiler es una colección de herramientas comúnmente utilizadas en pipelines biológicos. La que se utiliza en este trabajo es g:GOSt. Sirve para realizar análisis de enriquecimiento funcional en una lista de genes que se proporcionan como input (16).

Los genes seleccionados en la etapa anterior se someten al análisis de enriquecimiento funcional con g:GOSt. En este, se consideran solo las anotaciones de pathways de Gene Ontology: Biological process y de Reactome. También se tienen en cuenta que estas anotaciones sean no inferida por anotación electrónica (anotaciones automáticas no revisadas por un humano), que las anotaciones correspondan a pathways de entre 5 y 350 nodos, ya que los pathways demasiado grandes tienen poco valor a la hora de interpretar los resultados, y demasiados pathways pequeños disminuyen el poder estadístico del enriquecimiento debido a que se realizan excesivas pruebas múltiples (17).

D) Visualización de los resultados en Cytoscape

tanto la lista de genes como los resultados del enriquecimiento se utilizan para generar mapas de enfermedad en Cytoscape con los plugins de stringApp y los del *workflow* de EnrichmentMap.

a) Red de interacciones entre proteínas

El objetivo es importar a Cytoscape una red de interacción de proteínas a través del plugin de stringApp (22), accediendo a las interacciones entre proteínas de la base de datos STRING. Para ello, se debe acceder al menú Archivo > Importar > Red desde bases de datos públicas (Figura 4).

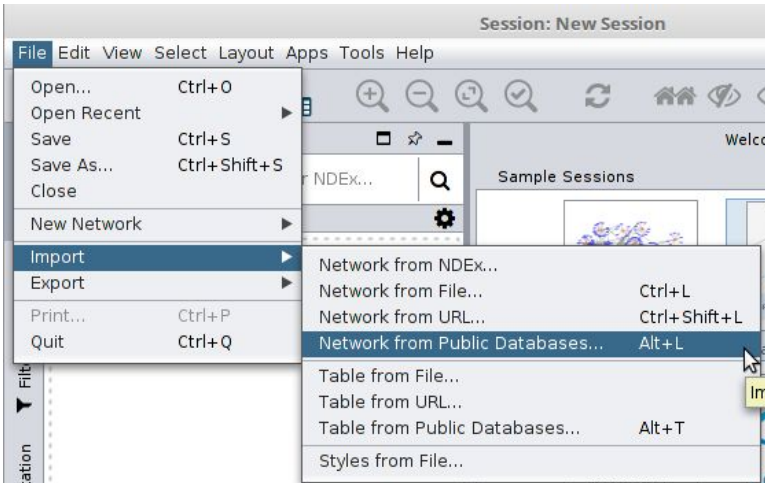

Figura 4. Importación de una red de interacción de proteínas a Cytoscape con el plugin *stringApp.*

Posteriormente, se debe seleccionar la opción “STRING: protein query” para hacer una búsqueda partiendo de la lista de entrez IDs generada en el archivo {efo}_entrezs_list.txt. Se debe copiar y pegar en el cuadro como en la Figura 5. Se ajusta el nivel de fiabilidad deseado para las interacciones entre proteínas que se van a importar a la red. En el ejemplo, se establece a 0.9. Se hace clic en “Import”, y probablemente aparezca una nueva ventana preguntando qué hacer en los casos en los que el gen o proteína introducido coincida con más de una entrada en la base de datos. Re revisa y se hace clic en Import de nuevo.

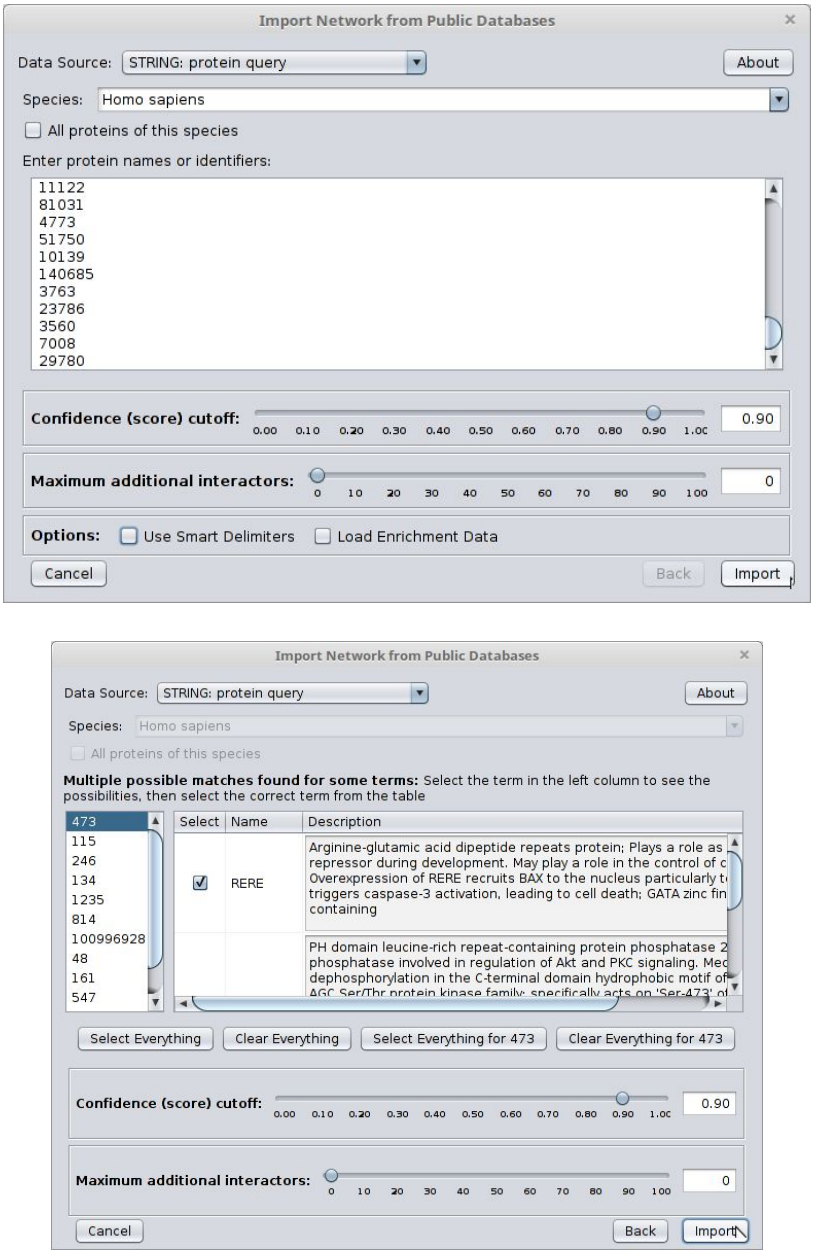

Figura 5. Generación de una red de interacción de proteínas en Cytoscape con el plugin *stringApp* a partir de un listado de entrez IDs. Se selecciona la opción “STRING: protein query” [1]. Se ajusta el grado de confianza deseado en las interacciones que se obtienen de la base de datos, en este caso 0.9. En el caso de que se encuentren múltiples resultados para la búsqueda, se selecciona el primero que aparece (marcado por defecto), ya que los

demás casos corresponden a coincidencias en el texto libre y no a identificadores de entrez, que son los deseados.

En la Figura 6 se puede observar la red importada, después de desactivar algunas opciones de visualización por defecto de STRING.

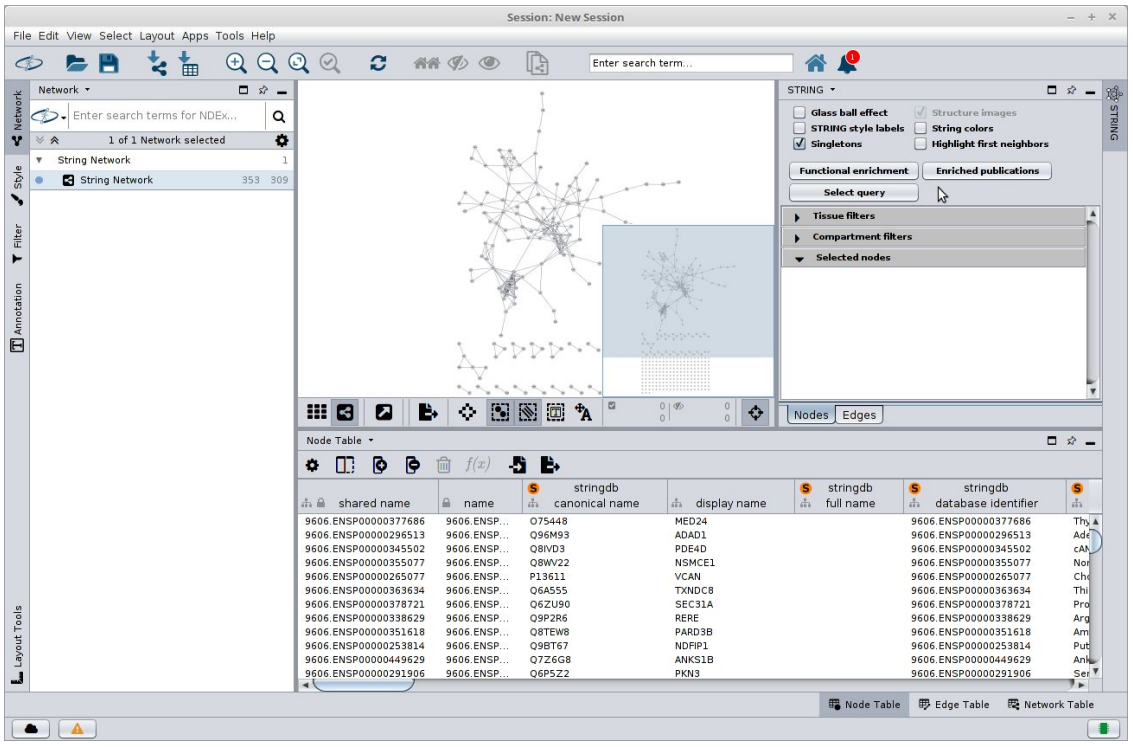

Figura 6. Captura de la sesión global de Cytoscape en la que se ha importado una red de interacciones entre proteínas a partir de una lista de genes con *stringApp*. Se han deseleccionado las opciones “glass ball effect”, “STRING style labels” y “string colors” para generar una red más nítida que se colorea posteriormente.

En la Figura 7, se muestran los parámetros modificados para colorear los nodos de la red en función del nivel de expresión de la proteína de ese nodo registrado en la base de datos Tissues (23) en tejido pulmonar. El amarillo corresponde a la menor expresión, y el morado a la mayor.

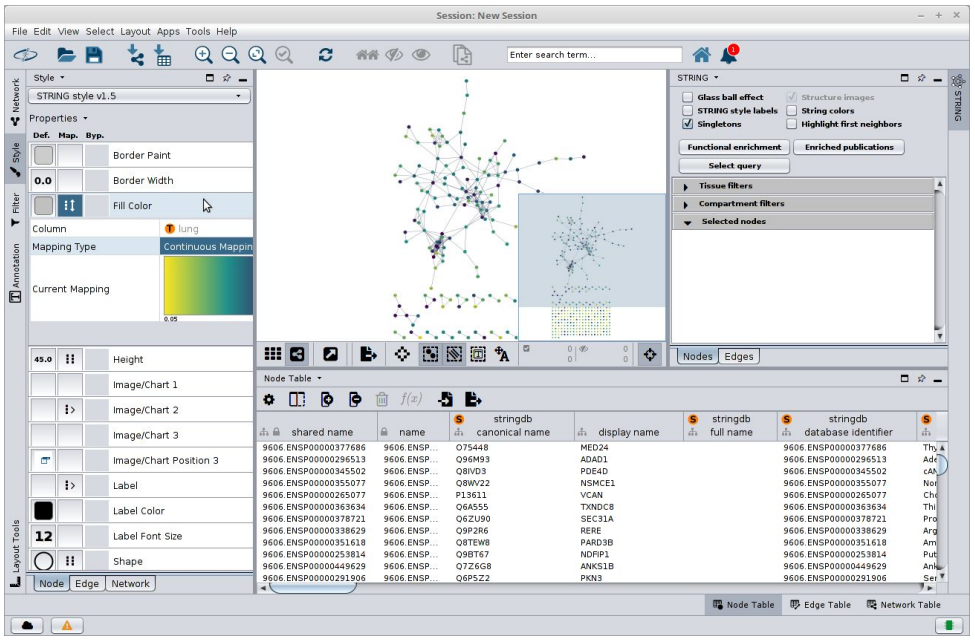

Figura 7. Ajuste del parámetro “color” en la pestaña “style” (estilo) para la visualización de la red de interacción de proteínas en la que se ha coloreado cada nodo en función de su nivel de expresión en el tejido de pulmón, siendo el amarillo la expresión más baja y el morado la más alta.

2.1.1. EnrichmentMap

En las siguientes capturas de pantalla se muestra el procedimiento a seguir para visualizar un mapa de enriquecimiento funcional a partir del enriquecimiento realizado con g:Profiler y de un archivo GMT disponible en la web de BaderLab, que actualizan periódicamente y ponen a disposición de la comunidad como recurso del pipeline de EnrichmentMap (17), y que contiene definiciones de los pathways. Tal archivo se descarga y adapta cuando se ejecuta el script prep_enrichmentmap.R.

El primer paso es abrir el plugin de EnrichmentMap en Cytoscape y seleccionar “Create Enrichment Map”. En esa ventana, deberemos cargar tanto el archivo GMT (mi_gmt.gmt) como el archivo de enriquecimiento generado en el script enrichment_gprofiler.R. Al seleccionar “Build”, se muestra en pantalla lo que se puede observar en la Figura 9.

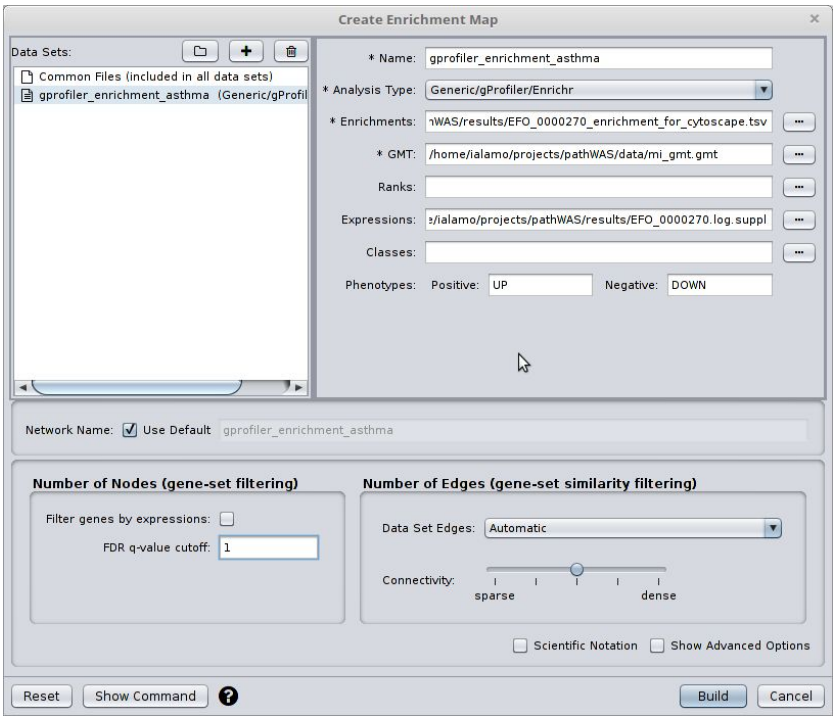

Figura 8. Creación del mapa de enriquecimiento funcional con EnrichmentMap. Como ya se han filtrado las anotaciones en el paso anterior, se puede poner un 1 en la celda de filtrado por FDR.

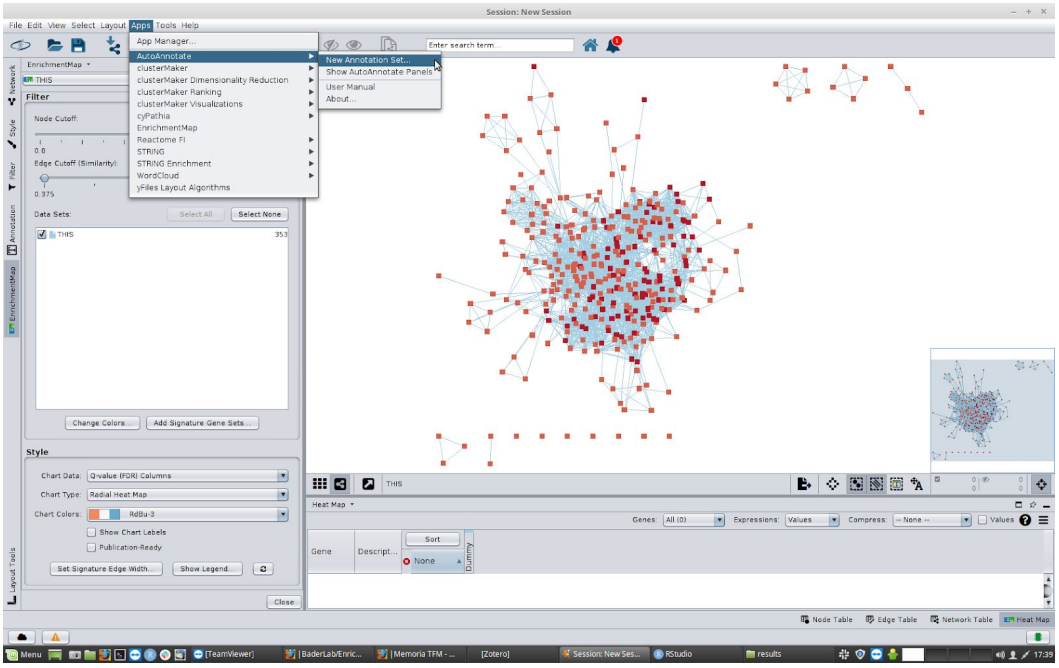

Figura 9. Creación de un nuevo mapa de enriquecimiento funcional con EnrichmentMap.

El siguiente paso es anotar el mapa de enriquecimiento funcional con los nombres de las funciones que más representadas se encuentran. En este paso, también se redistribuyen los nodos y se agrupan los términos que tienen algo en común. Para eso, se debe seleccionar en el menú Apps el plugin AutoAnnotate y utilizar la configuración que se muestra en la Figura 10. Si se marca la casilla “Layout network to prevent cluster overlap”, será más sencillo organizar los conjuntos de pathways de forma que no se solapen sus etiquetas y el mapa sea más fácilmente legible.

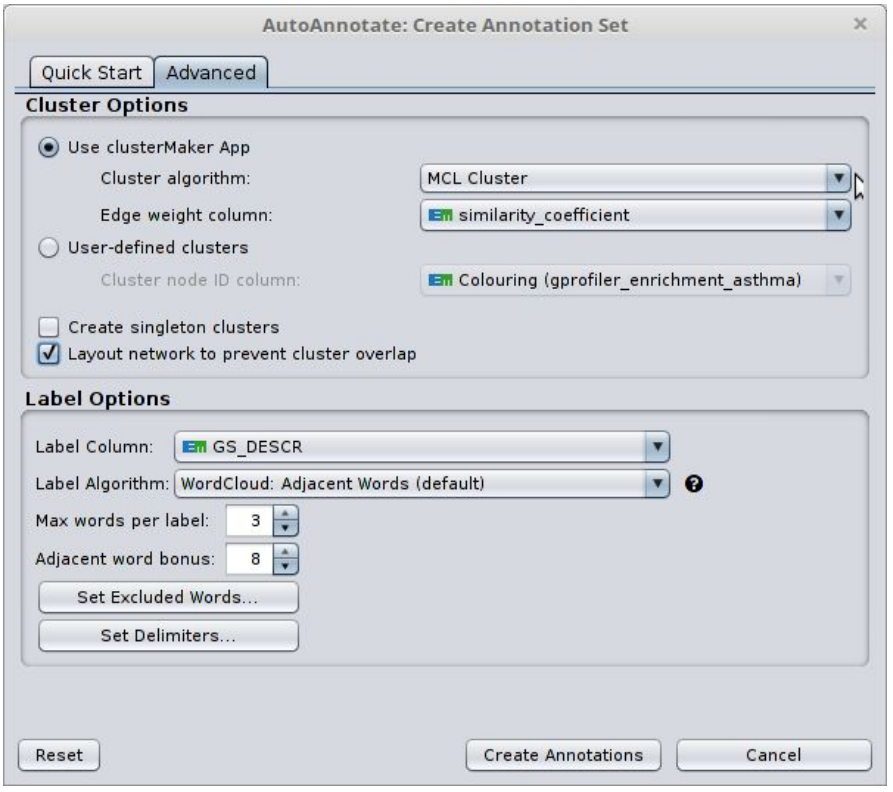

Figura 10. Opciones avanzadas seleccionadas para visualizar el mapa de enriquecimiento.

En redes grandes como estas, el layout automático no siempre va a devolver una red legible y útil, por lo que se requerirá de una ligera intervención manual para organizar en el espacio los nodos del mapa de enriquecimiento funcional, como en la Figura 11.

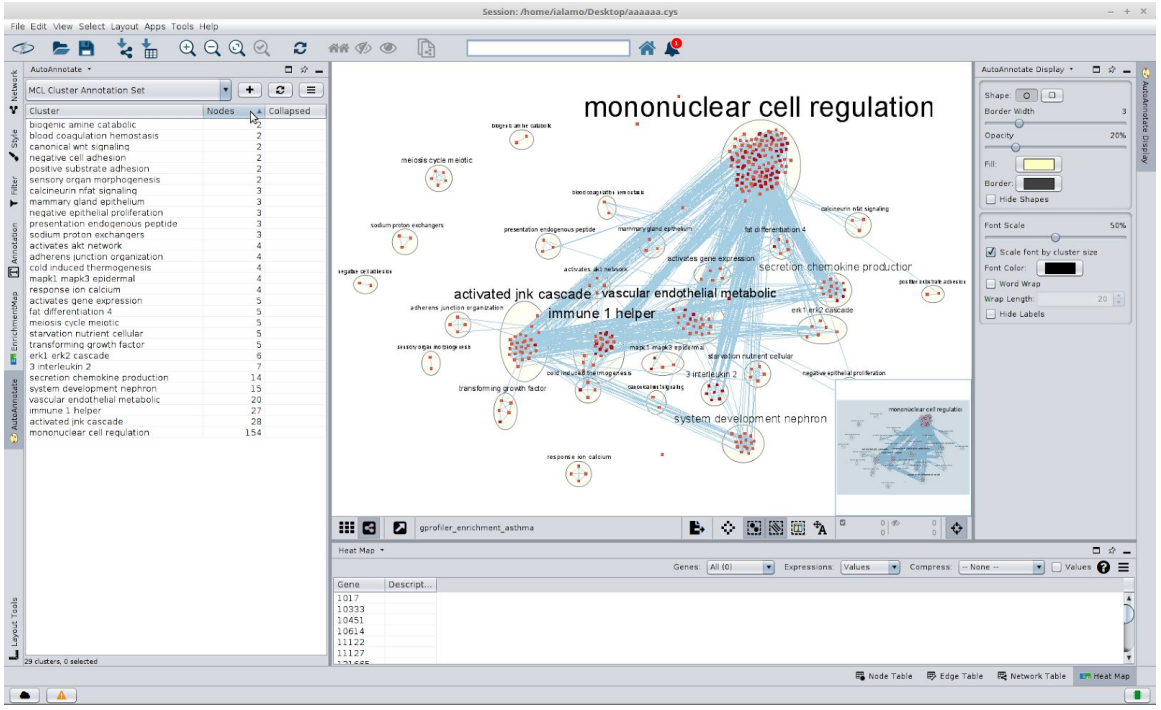

Figura 11. Vista global de la sesión de Cytoscape en la que se ha generado el mapa de enriquecimiento funcional.

2.1.2. Git

En el desarrollo de software, donde continuamente se añaden nuevas funcionalidades y cambios, es importante mantener un control de versiones para administrar el código fuente de nuestros programas. Git es el software de control de versiones más utilizado y GitHub es la plataforma más popular entre los desarrolladores. GitHub permite el desarrollo colaborativo de software y alojar proyectos utilizando Git como sistema de control de versiones. En el desarrollo de este proyecto se ha aprendido a utilizar esta plataforma y el código de este trabajo se encuentra disponible en www.github.com/mialaalv/pathwas.

2.2. Ejemplo de visualizaciones obtenidas

Como prueba de concepto se ha utilizado como enfermedad el asma, que en GWAS Catalog aparece con el código *EFO_0000270*. Se han incluido también sus términos hijos en la ontología, que son: asma obstructiva crónica, asma de inicio en la edad adulta, asma alérgica, asma atópica y asma de inicio en la infancia.

2.2.1. Red de interacción de proteínas

En la Figura 12 se puede observar un ejemplo de red de interacciones entre proteínas generada en Cytoscape con la herramienta desarrollada en el presente trabajo. La red se ha generado con el plugin stringApp a partir de una lista de genes asociados al asma obtenida de aplicar el software MAGMA a datos de asociaciones variante-enfermedad obtenidas de GWAS Catalog. Los nodos se han coloreado en función del nivel de expresión de esa proteína en tejido pulmonar según la base de datos Tissues (23), siendo el amarillo la menor expresión y el morado la mayor. Como se puede observar, la mayoría de las proteínas de la red, relacionadas con el asma, tienen un nivel de expresión medio-alto o alto, especialmente las que forman parte de la fracción de proteínas más interconectadas de la red.

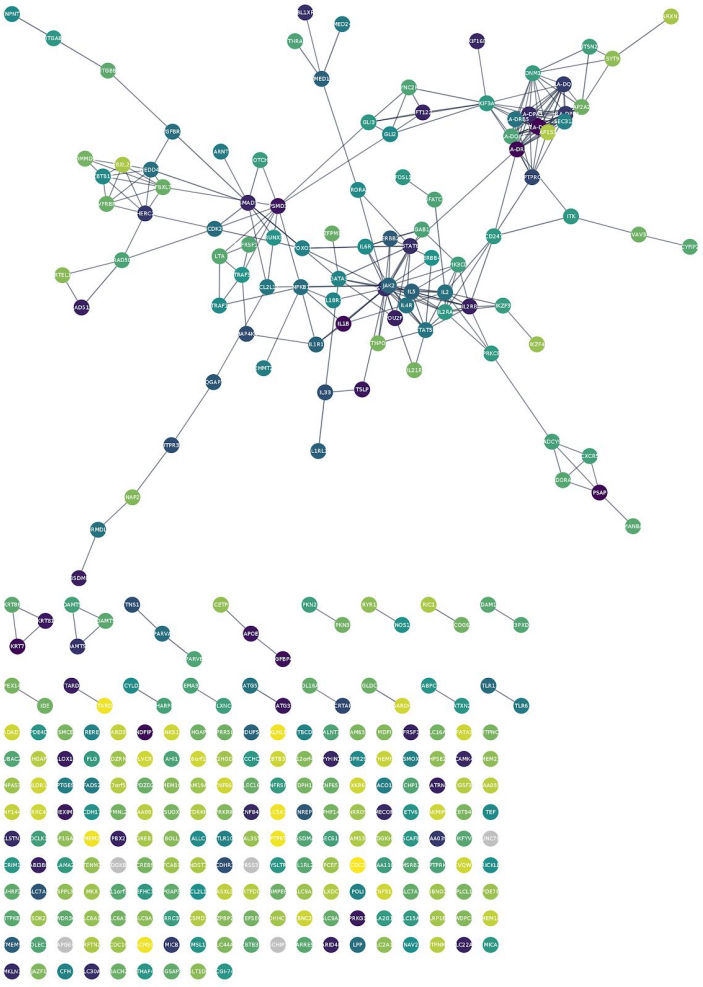

Figura 12. Visualización en Cytoscape de la red generada con el plugin *stringApp.*

2.2.2. Visualización del análisis de enriquecimiento funcional

En la Figura 13 se puede observar el mapa de enriquecimiento funcional generado con el pipeline de EnrichmentMap.

La información de pathways es redundante porque los genes a menudo participan en varias rutas. Además, en las bases de datos los pathways se organizan de forma jerárquica en ontologías, por lo que puede ocurrir que al hacer un enriquecimiento de pathways se estén mostrando como resultado diferentes versiones del mismo. EnrichmentMap combina estos pathways redundantes en “temas biológicos”, reduciendo la redundancia y facilitando la interpretación de los resultados.

Se muestra una visualización en la que los nodos son pathways, y se mostrarán ejes entre ellos cuando se compartan muchos genes entre pathways. El algoritmo de clustering (en este caso se ha usado MCL cluster) separa los pathways similares en grandes grupos temáticos, que se colorean de amarillo.

En el caso del asma, podemos ver que hay muchos grupos funcionales relacionados con la respuesta inmune, lo cual concuerda con la evidencia publicada (24,25)

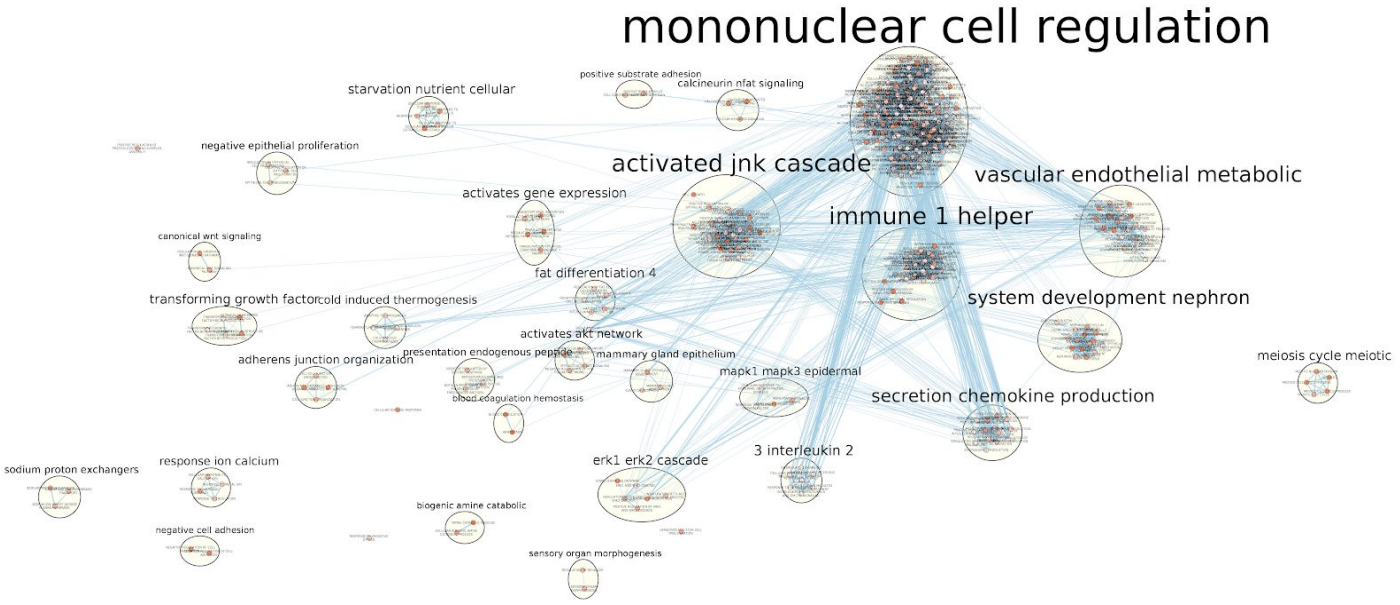

Figura 13. Visualización en Cytoscape (con *EnrichmentMap pipeline* (17)) del análisis de enriquecimiento funcional de g:Profiler sobre la lista de genes relacionados con el asma generada con MAGMA a partir de datos de asociaciones variante-enfermedad obtenidas de GWAS Catalog.

3. Conclusiones

- Se ha propuesto un pipeline para la creación de mapas de enfermedad a partir de datos de asociaciones de variantes de estudios de asociación de genoma completo (GWAS).
- Se pueden generar tanto mapas de interacción entre proteínas como mapas de enriquecimiento funcional.
- La visualización de un mapa de enriquecimiento funcional facilita la interpretación de los resultados al reducir la redundancia de estos, dada la redundancia intrínseca de los pathways.
- En general, se han cumplido los objetivos planteados originalmente. Se han tenido que tomar decisiones en el desarrollo del proyecto y priorizar unos sobre otros, pero aunque no se haya explorado todo lo planteado inicialmente, se ha establecido la metodología y la línea para hacerlo en los próximos meses. En el futuro próximo se pretende:
    - Explorar el pipeline desde datos de GWAS crudos en lugar de estadísticas de resumen.
    - Integrar direccionalidad de las interacciones en los mapas de interacción entre proteínas generados.

4. Glosario

Alelo/Variante: Variación de la secuencia de ADN en un determinado locus.

Atributo (*trait*): Característica o fenotipo determinada genéticamente.

Desequilibrio de ligamiento (*linkage disequilibrium,* LD): Se habla de desequilibrio de ligamiento entre dos variantes cercanas cuando los alelos de polimorfismos vecinos (en el mismo cromosoma) aparecen juntos en una población con mayor frecuencia que si no estuvieran vinculados.

Estudio de Asociación del Genoma Completo (*Genome Wide Association Studies*): análisis de una variación genética en todo el genoma humano para identificar su asociación a un rasgo observable.

Genoma de referencia: Base de datos digital de secuencias de material genético que se establece como ejemplo representativo de los genes de una especie.

GMT (*Gene Matrix Transposed file format*): Formato de archivo delimitado por tabulaciones que describe conjuntos de genes. Cada fila representa un conjunto de genes en el formato GMT. La primera columna son los nombres de los conjuntos, la segunda columna contiene una breve descripción del conjunto, y la tercera contiene el listado de genes del conjunto.

Locus/loci : una posición fija en un cromosoma, que determina la posición de un gen o de un marcador marcador genético. Se usa para identificar posiciones de interés sobre determinadas secuencias.

Pathway (ruta): una serie de interacciones entre moléculas que da lugar a un cambio o producto en la célula.

SNP (*single nucleotide polymorphism*): Variación en el material genético que afecta solo a un nucleótido.

5. Bibliografía

1. Carvallo P. Conceptos Sobre Genética Humana Para La Comprensión E Interpretación De Las Mutaciones En Cáncer Y Otras Patologías Hereditarias. Rev Médica Clínica Las Condes. 1 de julio de 2017;28(4):531-7.

2. Considine EC. The Search for Clinically Useful Biomarkers of Complex Disease: A Data Analysis Perspective. Metabolites. 2 de julio de 2019;9(7):126.

3. Lowe WL, Reddy TE. Genomic approaches for understanding the genetics of complex disease. Genome Res. octubre de 2015;25(10):1432-41.

4. Holloway JW, Prescott SL. Chapter 2 - The Origins of Allergic Disease. En: O'Hehir RE, Holgate ST, Sheikh A, editores. Middleton's Allergy Essentials [Internet]. Elsevier; 2017 [citado 24 de junio de 2020]. p. 29-50. Disponible en: http://www.sciencedirect.com/science/article/pii/B9780323375795000027

5. Manolio TA, Collins FS, Cox NJ, Goldstein DB, Hindorff LA, Hunter DJ, et al. Finding the missing heritability of complex diseases. Nature. 8 de octubre de 2009;461(7265):747-53.

6. Wang L, Jia P, Wolfinger RD, Chen X, Zhao Z. Gene set analysis of genome-wide association studies: methodological issues and perspectives. Genomics. julio de 2011;98(1):1-8.

7. Neumann A, Laranjeiro N, Bernardino J. An Analysis of Public REST Web Service APIs. IEEE Trans Serv Comput. 2018;1-1.

8. Hypertext Transfer Protocol -- HTTP/1.1 [Internet]. [citado 24 de junio de 2020]. Disponible en: https://www.w3.org/Protocols/rfc2616/rfc2616.html

9. Mazein A, Ostaszewski M, Kuperstein I, Watterson S, Novère NL, Lefaudeux D, et al. Systems medicine disease maps: community-driven comprehensive representation of disease mechanisms. Npj Syst Biol Appl. 2 de junio de 2018;4(1):1-10.

10. Disease Maps [Internet]. Disease Maps. [citado 24 de junio de 2020]. Disponible en: https://disease-maps.org/

11. Berridge MJ. Module 12: Signalling Defects and Disease. Cell Signal Biol. 1 de octubre de 2014;6:csb0001012.

12. Edgar R, Domrachev M, Lash AE. Gene Expression Omnibus: NCBI gene expression and hybridization array data repository. Nucleic Acids Res. 1 de enero de 2002;30(1):207-10.

13. Leinonen R, Sugawara H, Shumway M. The Sequence Read Archive. Nucleic Acids Res. enero de 2011;39(Database issue):D19-21.

14. Buniello A, MacArthur JAL, Cerezo M, Harris LW, Hayhurst J, Malangone C, et al. The NHGRI-EBI GWAS Catalog of published genome-wide association studies, targeted arrays and summary statistics 2019. Nucleic Acids Res. 8 de enero de 2019;47(D1):D1005-12.

15. MAGMA: Generalized Gene-Set Analysis of GWAS Data [Internet]. [citado 22 de abril de 2020]. Disponible en: https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1004219

16. Raudvere U, Kolberg L, Kuzmin I, Arak T, Adler P, Peterson H, et al. g:Profiler: a web server for functional enrichment analysis and conversions of gene lists (2019 update). Nucleic Acids Res. 2 de julio de 2019;47(W1):W191-8.

17. Reimand J, Isserlin R, Voisin V, Kucera M, Tannus-Lopes C, Rostamianfar A, et al. Pathway enrichment analysis and visualization of omics data using g:Profiler, GSEA, Cytoscape and EnrichmentMap. Nat Protoc. febrero de 2019;14(2):482-517.

18. R Core Team. R: A Language and Environment for Statistical Computing [Internet]. Vienna, Austria: R Foundation for Statistical Computing; 2020. Disponible en: https://www.R-project.org/

19. Magno R, Maia A-T. gwasrapidd: an R package to query, download and wrangle GWAS catalog data. Bioinformatics. 15 de enero de 2020;36(2):649-50.

20. Leeuw CA de, Mooij JM, Heskes T, Posthuma D. MAGMA: Generalized Gene-Set Analysis of GWAS Data. PLOS Comput Biol. 17 de abril de 2015;11(4):e1004219.

21. Benjamini Y, Hochberg Y. Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing. J R Stat Soc Ser B Methodol. 1995;57(1):289-300.

22. Doncheva NT, Morris JH, Gorodkin J, Jensen LJ. Cytoscape StringApp: Network Analysis and Visualization of Proteomics Data. J Proteome Res. 01 de 2019;18(2):623-32.

23. Palasca O, Santos A, Stolte C, Gorodkin J, Jensen LJ. TISSUES 2.0: an integrative web resource on mammalian tissue expression. Database J Biol Databases Curation [Internet]. 12 de febrero de 2018 [citado 25 de junio de 2020];2018. Disponible en: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5808782/

24. Krusche J, Basse S, Schaub B. Role of early life immune regulation in asthma development. Semin Immunopathol. febrero de 2020;42(1):29-42.

25. Lommatzsch M. Immune Modulation in Asthma: Current Concepts and Future Strategies. Respir Int Rev Thorac Dis. 8 de junio de 2020;1-11.

6. Anexos

6.1. Listado de enfermedades relacionadas con procesos de señalización celular

Listado extraido de *Berridge et al.* (11).

- Asma.
- Degeneración macular relacionada con la edad (AMD).
- Alzheimer.
- Cirrosis hepática.
- Síndrome de Cushing.
- Diabetes.
- Diabetes insípida (DI).
- Nefropatía diabética.
- Diarrea.
- Drogadicción.
- Disfunción eyaculatoria.
- Shock endotóxico.
- Enfermedad renal terminal (ERSD).
- Epilepsia.
- Disfunción eréctil.
- Enfermedad cardíaca.
- Hipercalcemia tumoral maligna.
- Hipertensión.
- Síndrome del intestino irritable.
- Síndrome metabólico.
- Migraña.
- Esclerosis múltiple.
- Nauseas.
- Obesidad.
- Osteoporosis.
- Dolor.
- Hiperparatiroidismo primario.
- Hiperparatiroidismo secundario.
- Enfermedad maniaco-depresiva.
- Parto prematuro.
- Artritis reumatoide.
- Esquizofrenia.
- Síndrome de muerte súbita del lactante (SIDS).
- Síndrome de Zollinger-Ellison.

6.2. Código desarrollado

El código utilizado en este trabajo se encuentra disponible en: https://github.com/mialaalv/pathWAS.

6.2.2. main.sh

```
#!/bin/bash

mkdir -p logs # create folder to store logs from executions
mkdir -p results # create folder to store result files

### Set up
echo MAGMA installation and download of resources
# Install MAGMA software from https://ctg.cncr.nl/software/magma
bash ./src/magma_setup.sh &> logs/magma_setup.log
echo DONE

echo Installation of R dependencies
# Install R dependencies
Rscript ./src/R_dependencies.R &> logs/R_dependencies.log
echo DONE

echo Fetching GWAS Catalog data
# fetch data from GWAS Catalog usin gwasrapidd bioconductor package. Lifts GRCh38 SNP locations to
GRCh37 build.
Rscript ./src/gwas_data_fetching_with_R.R &> logs/gwas_data_fetching_with_R.log
echo DONE

### MAGMA analysis
echo MAGMA Analysis
echo SNPs annotation
# SNPs annotation
./magma/magma --annotate --snp-loc
data/GWAS_data/EFO_0000270/EFO_0000270_GWAS_summary_grch37_forMAGMA.tsv --gene-loc
magma/res/NCBI37.3/NCBI37.3.gene.loc --out results/EFO_0000270
# make sure the SNP file and gene location file are in the same genome build
echo DONE

echo Gene analysis
# Gene analysis from SNP p-values
./magma/magma --bfile ./magma/res/g1000_eur/g1000_eur --pval
data/GWAS_data/EFO_0000270/EFO_0000270_GWAS_summary_grch37_forMAGMA.tsv ncol=NOBS
--gene-annot results/EFO_0000270.genes.annot --out results/EFO_0000270
# make sure the reference panel is in the same genome build as the SNP pval data
# NOBS is the initial sample size of the study the data were retrieved from
# *.genes.annot is the output from the annotation step.
echo DONE

echo P-val adjusting
# Adjusting pvalues for MAGMA output and filtering out the worst associations:
Rscript ./src/adjust_output_pvals.R &> logs/adjust_output_pvals.log
echo DONE

echo EnrichmentMap preparation
Rscript ./src/prep_enrichmentmap.R &> logs/prep_enrichmentmap.log
echo DONE
```

6.2.3. R_dependencies.R

```
#!/usr/local/bin/Rscript

#from github
if(!require(gwasrapidd)) remotes::install_github("ramiromagno/gwasrapidd")
#from bioconductor
if(!require(rtracklayer)) BiocManager::install("rtracklayer")
if(!require(GenomicRanges)) BiocManager::install("GenomicRanges")
if(!require(limma)) BiocManager::install("limma")
# from cran
if(!require(GSA)) install.packages("GSA")
if(!require(RCurl)) install.packages("RCurl")
if(!require(stringr)) install.packages("stringr")
if(!require(gprofiler2)) install.packages("gprofiler2")
if(!require(ggplot2)) install.packages("ggplot2")

sessionInfo()
```

6.2.4. magma_setup.sh

```
#!/bin/bash

# MAGMA v1.07bb installation and fetching of necessary resources

mkdir -p magma/res

# Install MAGMA
wget --no-check-certificate https://ctg.cncr.nl/software/MAGMA/prog/magma_v1.07bb.zip -P magma
echo YA
unzip magma/magma_v1.07bb.zip -d magma

# Gene locations, build GRCh37
wget https://ctg.cncr.nl/software/MAGMA/aux_files/NCBI37.3.zip -P magma/res
unzip magma/res/NCBI37.3.zip -d magma/res/NCBI37.3

# Reference population data (1000 genomes, GRCh37 genome build) to correct for LD
wget https://ctg.cncr.nl/software/MAGMA/ref_data/g1000_eur.zip -P magma/res
unzip magma/res/g1000_eur.zip -d magma/res/g1000_eur
```

6.2.5. gwas_data_fetching_with_R.R

```
#!/usr/local/bin/Rscript

library(gwasrapidd)
library(stringr)
library(rtracklayer)

##### get associations with trait asthma and its child terms:
# efo <- get_traits(efo_trait = "asthma")@traits$efo_id
efo <- "EFO_0000270" # asthma, used as example
```

```
##### path variables

associations_file <- file.path("data", "GWAS_data", efo, paste0(efo, "_associations.rds"))
studies_file <- file.path("data", "GWAS_data", efo, paste0(efo, "_studies.rds"))
variants_file <- file.path("data", "GWAS_data", efo, paste0(efo, "_variants.rds"))
traits_file <- file.path("data", "GWAS_data", efo, paste0(efo, "_traits.rds"))

summary_file <- file.path("data", "GWAS_data", efo, paste0(efo, "_GWAS_summary.rds"))
for_magma_file <- file.path("data", "GWAS_data", efo, paste0(efo,
"_GWAS_summary_grch37_forMAGMA.tsv"))

##### get data from GWAS Catalog

efos <- c(efo, get_child_efo(efo_id = efo)[[1]])

my_traits <- get_traits(efo_id = efos)

# summary of the traits considered:
my_traits@traits
# # A tibble: 5 x 3
#   efo_id        trait                  uri
#   <chr>         <chr>                  <chr>
# 1 EFO_0000270 asthma                    http://www.ebi.ac.uk/efo/EFO_0000270
# 2 EFO_0010638 atopic asthma             http://www.ebi.ac.uk/efo/EFO_0010638
# 3 EFO_0009759 Chronic Obstructive Asthma http://www.ebi.ac.uk/efo/EFO_0009759
# 4 EFO_1002011 adult onset asthma        http://www.ebi.ac.uk/efo/EFO_1002011
# 5 EFO_0004591 childhood onset asthma          http://www.ebi.ac.uk/efo/EFO_0004591

my_associations <- get_associations(efo_id = efos) # 2363 associations

association_ids <- my_associations@associations$association_id

my_variants <- get_variants(association_id = association_ids) # n 1541
my_studies <- get_studies(association_id = association_ids) # n 102

studies_from_associations <- sapply(association_ids, function(x) {
  a <- get_studies(association_id = x)@studies$study_id
  # if(length(a) > 1) print(paste0("looko at ", x, ":", a))
  return(a)
}) # checked that there's only 1 study per association

variants_from_associations <- sapply(association_ids, function(x) {
  a <- get_variants(association_id = x)@variants$variant_id
  # if(length(a) > 1) print(paste0("look at ", x, ":", a))
  return(a)
}) # returns a list with variants per association

# in case there is more than one variant per association, the rows are duplicated

# I only take the initial number of individuals (not replicas), ancestry_id = 1
nobs <- dplyr::filter(my_studies[studies_from_associations,]@ancestries, ancestry_id ==
1)$number_of_individuals

df <- data.frame(association_id = association_ids, study = studies_from_associations, NOBS = nobs)

n_vars <- sapply(variants_from_associations, length) # number of variants per accession
df <- df[rep(association_ids, n_vars),]                # repeat each row according to the number of variants
```

```
df$SNP <- unlist(variants_from_associations)      # add the variants to the df

df$P <- my_associations[df$association_id,]@associations$pvalue
df$CHR38 <- my_variants[df$SNP,]@variants$chromosome_name
df$BP38 <- my_variants[df$SNP,]@variants$chromosome_position

# some variants do not have specified positions, but they are specified in the name of the variant. it can
be parsed to rescue them:
# for the df rows with na in chr
for(x in 1:nrow(df)){
  if(is.na(df[x,"CHR38"]) & stringr::str_starts(string = df[x, "SNP"], pattern = "ch")){
        capture <- stringr::str_match(df[x, "SNP"], pattern = "ch[r]?(.*):([0-9]*)")

        df$CHR38[x] <- capture[1,2]
        df$BP38[x] <- capture[1,3]
  }
}

# now there are only two variants with no chromosome position annotations:
df[which(is.na(df$CHR38)),"SNP"]
# [1] "rs152271219" "rs67431028"

# they are dropped
df <- df[complete.cases(df),]


####### Lift genomic positions to GRCH38:

path = system.file(package="liftOver", "extdata", "hg38ToHg19.over.chain")
ch = import.chain(path)

ranges38 <- GRanges(seqnames = paste0("chr", df$CHR38), ranges = df$BP)
ranges37 <- liftOver(x = ranges38, chain = ch)

df$CHR <- stringr::str_replace(string = as.character(ranges37@unlistData@seqnames),pattern = "chr",
replacement = "")
df$BP <- as.character(ranges37@unlistData@ranges@start)

###### saving results and data for future reference #######

dir.create(file.path("data", "GWAS_data", efo), recursive = T)

saveRsink(file.path("logs", "gwas_data_fetching_with_R_log.txt"))
DS(my_associations, associations_file)
saveRDS(my_studies, studies_file)
saveRDS(my_variants, variants_file)
saveRDS(my_traits, traits_file)

saveRDS(df, summary_file)

# rearrange columns to meet requested order: SNP CHR BP P NOBS
write.table(x = df[, c("SNP", "CHR", "BP", "P", "NOBS")], file = for_magma_file, sep="\t", quote=F, row.names
= F, col.names = T)

##### Session Info
sessionInfo()
```

6.2.6. adjust_output_pvals.R

```
#!/usr/local/bin/Rscript

# get stats
efo <- "EFO_0000270"
res_file <- file.path("results", paste0(efo, ".genes.out"))
out_file <- file.path("results", paste0(efo, ".genes.out_with_FDR_correction.tsv"))
out_filtered_file <- file.path("results", paste0(efo, ".genes.out_with_FDR_correction_filtered_cutoff.tsv"))
list_entrezs_file <- file.path("results", paste0(efo, "_entrezs_list.txt"))

res <- read.delim(res_file, sep="")
res$P_adjust <- p.adjust(p = res$P, method = "fdr")

write.table(x = res, quote = F, file = out_file, sep="\t", row.names = F)

png("results/analisis_genes_cutoff.png")
plot(res$P_adjust, res$ZSTAT, xlab = "p-valor ajustado", ylab = "Z stat", main = "Análisis de genes con
MAGMA")
cut0 <- mean(res$ZSTAT) - 0.9*sd(res$ZSTAT)
abline(h =  cut)
dev.off()

res_filtered <- res[res$P_adjust<0.05,] # filter by adjusted p-value, consider valid only those <0.05

cut <- mean(res_filtered$ZSTAT) - 0.9*sd(res_filtered$ZSTAT)
res_filtered <- res_filtered[res_filtered$ZSTAT > cut,] # keep only values above cutoff, which is mean - 3
times the standard deviation

write.table(x = res_filtered, quote = F, file = out_filtered_file, sep="\t", row.names = F)

print("Number of genes dropped in filtering: ", sum(res$ZSTAT < cut))
write.table(x=res_filtered$GENE, list_entrezs_file, quote=F, row.names = F, col.names = F)

sessionInfo()
```

6.2.7. enrichment_gprofiler.R

```
#!/usr/local/bin/Rscript

library(gprofiler2)
library(ggplot2)
library(stringr)

efo <- "EFO_0000270"
res_file <- file.path("results", paste0(efo, ".genes.out_with_FDR_correction_filtered_cutoff.tsv"))
fig1_file <- file.path("results", paste0(efo, "_gprofiler_unfiltered_plot.jpeg"))
fig2_file <- file.path("results", paste0(efo, "_gprofiler_filtered_plot.jpeg"))

pre_out_file <- file.path("results", paste0("gprofiler2_", efo, ".tsv"))
out_file <- file.path("results", paste0(efo, "_enrichment_for_cytoscape.tsv"))

res <- read.delim(res_file)

genes <- as.character(res$GENE[order(res$ZSTAT, decreasing = T)]) # ordered list of genes by
decreasing Z-score of association to disease trait
```

```
### in asthma example, none are filtered out

# perform functional enrichment of the gene list
gprofres <- gprofiler2::gost(query = genes,
                    organism = "hsapiens",
                    ordered_query = T, # if the ids are ordered by biological importance (decreasing importance == increasing p-value)
                    correction_method = "fdr",  # if commented, corrects with default method developed by g:Profile team
                    exclude_iea = T,   # excludes automatic annotations (in silico)
                    sources = c("GO:BP", "REAC"), # only searches in GO biological process and Reactome
                    evcodes = T,                  # give extra columns with list of intersections trait-gene
                    numeric_ns = "ENTREZGENE_ACC" # entry id is entrez
)

p <- gostplot(gostres = gprofres, capped = FALSE, interactive = FALSE)
ggsave(plot = p, filename = fig1_file, device = "jpeg")

# keep only the annotations to pathways that are not too big or too small
ind_size <- gprofres$result$term_size >= 5 & gprofres$result$term_size <= 350
print(paste0("Dropped ", sum(!ind_size), " items. Kept annotations to pathways with sizes 3-350 items."))
gprofres2 <- gprofres$result[which(ind_size),]

# keep only annotations with intersection >3
ind_inter <- sapply(gprofres2$intersection, function(x) length(strsplit(x, split = ",")[[1]])>=3 )
print(paste0("Dropped ", sum(!ind_inter), " items. Kept annotations to pathways with 3 or more genes in our input."))
gprofres2 <- gprofres2[which(ind_inter),]

gprofres3 <- gprofres2
gprofres3$parents <- paste(gprofres3$parents, collapse = ";")

# make table with appropriate format:
gprofres3 <- data.frame(source = gprofres3$source,term_name = gprofres3$term_name, term_id = gprofres3$term_id, adjusted_p_value=gprofres3$p_value, negative_log10_of_adjusted_p_value = -log(base = 10, x = gprofres3$p_value), term_size = gprofres3$term_size, query_size = gprofres3$query_size, intersection_size = gprofres3$intersection_size, effective_domain_size = gprofres3$effective_domain_size, intersections = gprofres3$intersection)

write.table(gprofres3, pre_out_file, sep="\t", quote = F, row.names = F)

#repeat plot after filtering
mock <- list(gprofres2, gprofres$meta)
names(mock) <- c("result", "meta")
q <- gostplot(mock, capped = FALSE, interactive = FALSE)
ggsave(plot = p, filename = fig2_file, device = "jpeg")

##### make table appropriate for EnrichmentMap
cytoscape_enrichment <- data.frame(GO.ID = gprofres3$term_id, Description = gprofres3$term_name, p.Val = gprofres3$adjusted_p_value, FDR = gprofres3$adjusted_p_value, Phenotype = 1, Genes = stringr::str_replace_all(string = gprofres3$intersections, pattern = " ", replacement = ","))
write.table(cytoscape_enrichment, out_file, sep="\t", quote = F, row.names = F)


sessionInfo()
```

6.2.8. prep_enrichmentmap.R

```
#!/usr/local/bin/Rscript

# download UP-TO-DATE gmt gene sets file from baderLab's repo
library(limma)
library(GSA)
library(RCurl)

working_dir <- getwd()
gmt_url = "http://download.baderlab.org/EM_Genesets/current_release/Human/Entrezgene/"
#list all the files on the server
filenames = getURL(gmt_url)
tc = textConnection(filenames)
contents = readLines(tc)
close(tc)

#get the gmt that has all the pathways and does not include terms inferred from electronic
annotations(IEA)
rx = gregexpr("(?<=<a href=\")(.*.GOBP_AllPathways_no_GO_iea.*.)(.gmt)(?=\">)",
            contents, perl = TRUE)
gmt_file = unlist(regmatches(contents, rx))

dest_gmt_file <- file.path(working_dir, "data", paste("Supplementary_Table3_", gmt_file, sep="") )

download.file(
  paste(gmt_url,gmt_file,sep=""),
  destfile=dest_gmt_file
)

##################
gmt_file <- file.path("data",
"Supplementary_Table3_Human_GOBP_AllPathways_no_GO_iea_June_01_2020_entrezgene.gmt") #
downloaded in previous script
my_gmt_file <- file.path("data", "mi_gmt.gmt")

# get gene set file
migmt <- read.delim(gmt_file, sep = "#", header = F)

# parse it to turn it into a data frame
mi_gmt <- apply(migmt, MARGIN = 1, FUN=function(x){
  a <- strsplit(x, "\t")[[1]]
  return(data.frame(name = a[1], desc = a[2], genes = paste(a[3:length(a)], collapse = " ")))
})

mi_gmt <- do.call("rbind", mi_gmt)

replacement <- sapply(as.character(mi_gmt$name), function(x){
   a <- strsplit(x, "%")[[1]]
  return(a[length(a)])
})

mi_gmt$name <- replacement

write.table(x = mi_gmt, file = my_gmt_file, quote=F, row.names = F, col.names = T)

sessionInfo()
```