Anexo

Análisis de variantes en esclerosis múltiple en el sur de Europa

Adrián Valls Carbó

24 de diciembre, 2021

Índice

Introducción

Los estudios de asociación genética son un tipo de estudios en los que se realiza un genotipado de pacientes y controles para intentar determinar qué genes se encuentran asociados a enfermedad. Este tipo de estudios sirven para detectar variantes asociadas a enfermedades relativamente frecuentes en la población. Para ello se realiza un genotipado de determinados SNPs en pacientes y en controles para determinar qué SNPs pueden estar asociados a enfermedad.

El pipeline que seguiremos se encuentra referenciado en el siguiente esquema:

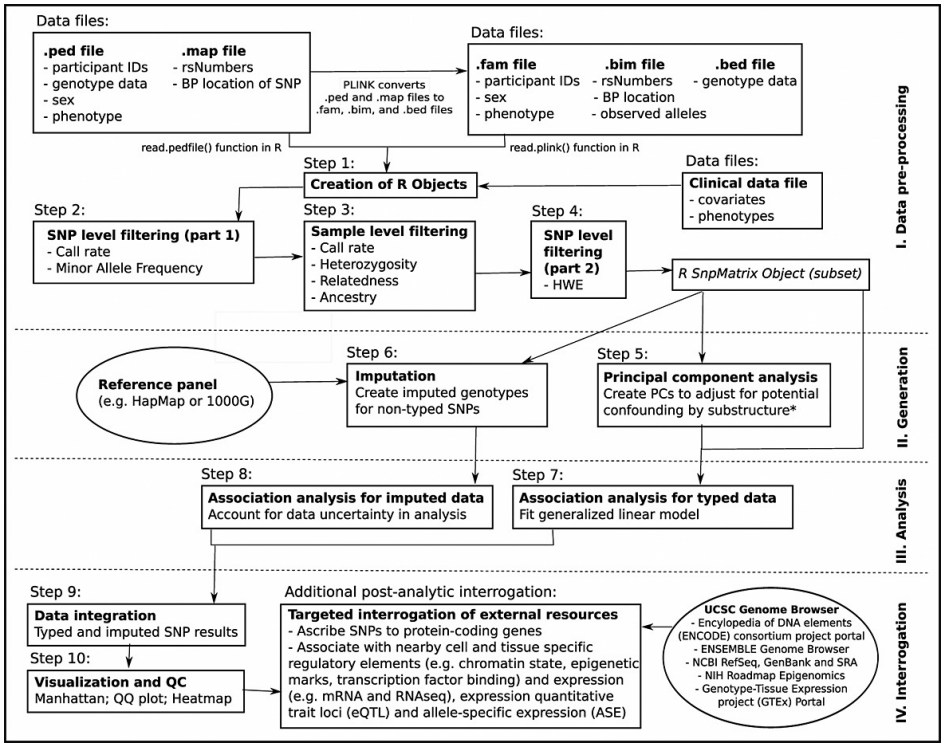

Figura 1: Pipeline (imagen tomada de Reed et al. (2015))

Cargando los datos

```
# Definimos los archivos
file_tabla<-"../Data_in"
file_graph<-"../Graficos"
file_data.out<-"../Data_output"
file_temp<-file.path(file_data.out, "file_temp")
file_resul<-file.path(file_data.out, "resultados")

# Creamos las carpetas si no existen
if (!dir.exists(file_graph)){dir.create(file_graph)}
if (!dir.exists(file_data.out)){dir.create(file_data.out)}
if (!dir.exists(file_temp)){dir.create(file_temp)}
if (!dir.exists(file_resul)){dir.create(file_resul)}

# Anotamos el archivo .gds
gds.file<-file.path(file_resul, "datos.gds")
```

```
# Anotamos donde se encuentran los diferentes archivos
archivo<-"Barcelona_PreQC"

# Definimos los archivos
bed <- file.path(file_tabla, paste0(archivo, ".bed"))
fam <- file.path(file_tabla,paste0(archivo, ".fam"))
bim <-file.path(file_tabla,paste0(archivo, ".bim"))
```

```
# Llamamos a la funcion seqBED2GDS del paquete SeqVarTools
# Solo ejecutaremos este fragmento la primera vez para procesar los .bim
seqBED2GDS(bed, fam, bim,  gds.file, verbose = FALSE)
```

Una vez definidos los campos, podemos abrir el archivo. Sin embargo los nombres de los rsID no han sido bien definidos, en los archivos por lo que debemos modificarlo

```
# Abrimos el archivo
gds <- seqOpen(gds.file)

# Vemos que todos tienen 2 alelos
#table(nAlleles(gds))

# Podemos extraer la posicion de los SNP
#head(seqGetData(gds, "position"))

# También podemos extraer los nombres de las muestras
#head(seqGetData(gds, "sample.id"))

# Otra opcion es extraer datos sobre la anotacion
#head(seqGetData(gds, "variant.id"))

# En primer lugar extraemos los nombres de la anotacion
rsID <- seqGetData(gds, "annotation/id")
sample.id <- seqGetData(gds, "sample.id")
sexo<-seqGetData(gds, "sample.annotation/sex")

# Cerramos el archivo
seqClose(gds)

# Cambiamos el archivo y le anotamos los nuevos nombres
setVariantID(gds.file, rsID)

# Podemos realizar una tabla en la que se especifiquen los alelos
#geno <- getGenotype(gds)

#geno <- getGenotypeAlleles(gds)
```

Para el estudio disponemos de 319950 SNPs, 464 individuos de los cuales 232 son controles, y 232 son casos.

Control de calidad

En el filtrado de los datos utilizaremos los siguientes parámetros:

Filtrado de los SNP: primera parte

Eliminaremos de acuerdo a 3 categorías

1.-Aquellas que tengan un call rate <99.5 % y minor allele frequency (MAF) <=5 %, es decir aquellos SNPs en las que todos los individuos sean totalmente iguales

2.-Aquellas con un call rate <99 % con un MAF entre el 5 y el 10 %

3.-Aquellas con un call rate inferior al 98 % con un MAF superior al 10 %

```
gds <- seqOpen(gds.file)

# Para realizar esto primero debemos determinar el call rate
miss.snp <- missingGenotypeRate(gds, margin="by.variant")


# Estimamos la frecuencia de cada alelo
afreq <- alleleFrequency(gds)

# Cerramos el archivo
seqClose(gds)

# Calculamos la frecuencia del alelo y su variante menor, obteniendo el menor
maf <- pmin(afreq, 1-afreq)

# Aplicamos los filtros
f1<-maf<=0.05 & miss.snp > 0.005
f2<- maf>0.05 & maf<=0.1 & miss.snp > 0.01
f3<-maf>0.1 & miss.snp>0.02
var.call<-!ifelse(f1+f2+f3==0, FALSE, TRUE)

# Seleccionamos aquellas variantes que quedan dentro del análisis
filt.snp.call<-rsID[!var.call]
snp.sel=rsID[var.call]
```

De este modo eliminamos 5511 variantes por encontrarse en alguna de las categorías anteriores

Filtrado de las muestras

1. Call rate <98 %: excluiremos aquellas muestras que presenten menos de un 98 % de los SNPs de la muestreados
2. Discrepancia de género: si el género reportado es femenino pero en los datos de genotipado muestra genes correspondientes al cromosoma Y, estas muestras serán eliminadas.
3. Heterocigosidad autosómica: aquellos individuoes que presenten una heterocigosidad autosómica mayor a 3 desviaciones estandar de la media serán excluidos del análisis. Además aquellas que tengan un inbreeding superior a 0.1
4. IBD (Identity by descendent) excesiva: aquellas muestras que presenten una mayor correlación serán eliminados, pues se consideran duplicados genéticos

```
# Determinamos los limites
call.rate.sample = 0.98
inbreeding= 0.1
```

Filtrado de call rate >98 % por muestra

El siguiente paso es filtrar las muestras que tengan un bajo índice de genotipado, por debajo del call rate que hemos definido con anterioridad

```
gds<-seqOpen(gds.file)

# Establecemos un filtro
seqSetFilter(gds, variant.id=snp.sel, verbose=FALSE)

# Calculamos los fallos de genotipado por muestra
miss.var <- missingGenotypeRate(gds, margin="by.sample")

# Sin embargo en nuestra muestra solo hay una que muestra baja calidad
filt.samp.call<-sample.id[miss.var>(1-call.rate.sample)]

# Seleccionamos las muestras
sample.sel<-sample.id[!sample.id%in%filt.samp.call]
seqClose(gds)
```

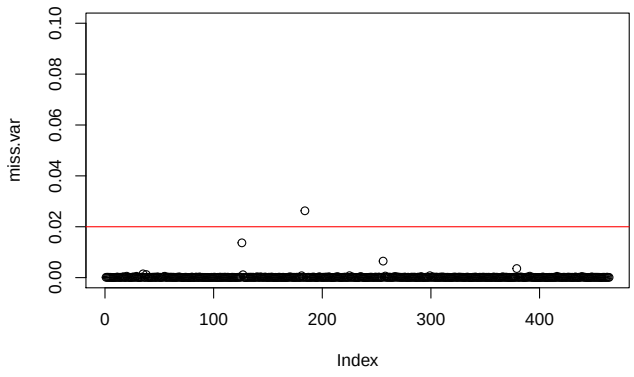

Figura 2: Muestras con bajo call rate

Vemos que existe 1 muestra que presente un call rate inferior al 98 %.

Podemos ver tambien los perdidos segun los cromosomas sean autosómicos o sexuales

```
gds<-seqOpen(gds.file)

# Establecemos un filtro
seqSetFilter(gds, variant.id=snp.sel, verbose=FALSE)

# Extraemos los cromosomas
chr<- seqGetData(gds, "chromosome")
chr.X<- seqGetData(gds, "variant.id")[chr=="X"]
chr.Y<-seqGetData(gds, "variant.id")[chr=="Y"]
chr.auto<- seqGetData(gds, "variant.id")[chr<25]

# Establecemos el filtro
seqSetFilter(gds, variant.id=chr.auto, verbose=FALSE)
miss.var <- missingGenotypeRate(gds, margin="by.sample")

# Vemos que la muestra muestra una tasa de bajo genotipado

length(sample.id[miss.var>(1-call.rate.sample)])
```

```
## [1] 1
```

```
seqClose(gds)
```

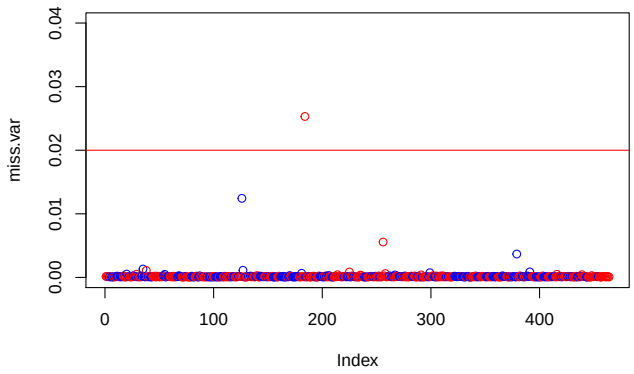

Figura 3: Muestras autosómicas con bajo call rate

Se aprecia que en los cromosomas autosómicos la tasa de perdidos es baja, y que los perdidos son principalmente en cromosomas sexuales

Comprobando la concordancia de sexo genómico y sexo anotado

En muchas ocasiones el sexo genotipado y el referido por el paciente no coincide. Debemos conseguir que ambos coincidan. Por lo general la heterocigosidad de los genes contenidos en el cromosoma X es la que nos sirve para comprobarlo, ya que los hombres deberían ser todos homocigotos para los genes del cromosoma X y las mujeres deben tener una heterocigosis mayor a 0 (alrededor de 0.3)

```
gds<-seqOpen(gds.file)

# Establecemos el filtro para el cromosoma X
seqSetFilter(gds, variant.id=chr.X, sample.id=sample.sel, verbose=FALSE)
sexo<-seqGetData(gds, "sample.annotation/sex")

# Determinamos la heterocigosidad por muestra
```

```
het.var <- heterozygosity(gds, margin="by.sample")

# Calculamos aquellas muestras femeninas que se salen de los límites
# tomando 3 desv respecto a la media
media.het.f<-mean(het.var[sexo=="F"])
desv.f<-sd(het.var[sexo=="F"])
lim.f<-media.het.f+c(3,-3)*desv.f

# Realizamos lo mismo para los varones
media.het.m<-mean(het.var[sexo=="M"])
desv.m<-sd(het.var[sexo=="M"])
lim.m<-media.het.m+c(3,-3)*desv.m

# Realizamos un flag para aquellas muestras que se salgan de los limites
flag.f<-!between(het.var[sexo=="F"], lim.f[2], lim.f[1])
flag.m<-!between(het.var[sexo=="M"], lim.m[2], lim.m[1])

# Seleccionamos las muestras
samp.fem.err<-seqGetData(gds, "sample.id")[sexo=="F"][flag.f]
samp.masc.err<-seqGetData(gds, "sample.id")[sexo=="M"][flag.m]

# Guardamos las que hemos eliminado
filt.sexo.erroneo<-c(samp.fem.err, samp.masc.err)

# Seleccionamos aquellas muestras
sample.sel=sample.sel[!sample.sel%in%filt.sexo.erroneo]

seqClose(gds)
```

Teoricamente la heterocigosidad del cromosoma X en mujeres debe de ser en torno al 30 %, aunque en nuestra muestra se encuentra en 9 %. Sin embargo vemos en el gráfico que existen varias muestras pertenecientes a mujeres que tienen una heterocigosidad baja por lo que eliminaremos los datos que se encuentren 3 desviaciones fuera de la media. En los hombres vemos que ocurre lo mismo (las muestras eliminadas son 201416607, 201416644, 201416710, 201416743, 201416787, 201416795, 201416807, 201416811, 201416855, 201416874, 201416875, 201416678, 201416716, 201416726, 201416771, 201416888, 201416915, 201416969, 201417005, 201417107)

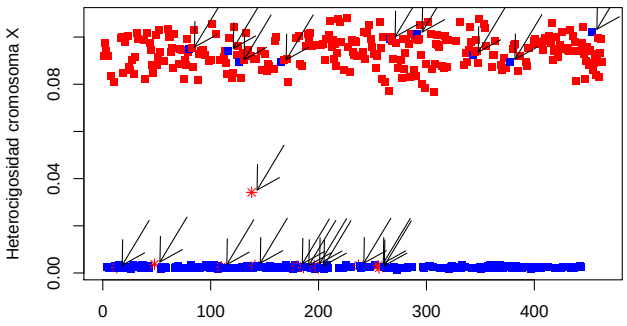

Figura 4: Muestras con fallo de sexo

Eliminando los outliers de heterocigosidad

Los SNPs que tienen más de un 0.1-0.15 de heterocigosidad por lo general son outliers y se suelen eliminar. Además según lo reflejado en la introducción eliminaremos aquellas muestras que se alejen más de 3 desviaciones típicas de la media de heterocigosidad de la muestra

```
gds<-seqOpen(gds.file)

# Aplicamos el filtro
seqSetFilter(gds, sample.id=sample.sel, variant.id=snp.sel, verbose=FALSE)

# Calculamos la heterocigosidad
het.var <- heterozygosity(gds, margin="by.sample")

# Definimos los límites aceptables
media<-mean(het.var)
desviacion<-media+c(1,-1)*3*sd(het.var)



# Seleccionamos aquellas muestras que se encuentran dentro de los límites
flag.het<-!between(het.var, desviacion[2], desviacion[1])
filt.het.baja<-seqGetData(gds, "sample.id")[flag.het]

# Aplicamos cuales son las muestras seleccionadas
sample.sel=sample.sel[!sample.sel%in%filt.het.baja]

seqClose(gds)
```

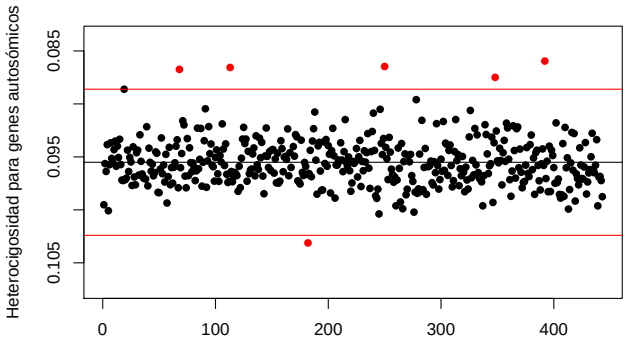

Figura 5: Muestras con extrema heterocigosidad

Se aprecia que hay un par de muestras que tienen una mayor heterocigosidad que la aceptada. Aunque por lo general se cumple que es inferior al 0.15 de heterocigosidad

También deseamos desechar aquellas muestras que presenten cifras elevadas de endogamia (inbreeding). De forma genérica tomaremos un límite de endogamia del 0.1

```
gds<-seqOpen(gds.file)

# Aplicamos el filtro
seqSetFilter(gds, sample.id=sample.sel, variant.id=snp.sel, verbose=FALSE)

# Calculamos el inbreed
inbreed<-inbreedCoeff(gds, margin="by.sample")
```

```
# Seleccionamos aquellas muestras que se encuentran dentro de los límites
filt.inbred<-seqGetData(gds, "sample.id")[inbreed>0.1]

seqClose(gds)

# Aplicamos cuales son las muestras seleccionadas
sample.sel=sample.sel[!sample.sel%in%filt.inbred]
```

Detección de muestras genéticamente relacionadas

Los estudios GWAS son estudios basados en muestras poblacionales, por lo que relaciones familiares entre individuos no es representativo de la muestra. Por este motivo tenemos que eliminar aquellas muestras que se encuentren relacionadas entre ellas. En primer lugar eliminamos aquellos SNPs que se encuentran correlacionados con la funcion `snpgdsLDpruning()` que elimina aquellos SNPs que se encuentran altamente correlacionados. Posteriormente podemos calcular la correlacion de las muestras con la funcion `snpgdsMoM` que calcula la relacion de las muestras. Eliminaremos aquellas muestras que tengan una relación mayor a 0.1

```
gds <- seqOpen(gds.file)

set.seed(1000)

# Realizamos el corte por 0.1 como indica en el archivo word
snpset <- snpgdsLDpruning(gds, ld.threshold=0.1, verbose = FALSE,
                          sample.id = sample.sel, snp.id = snp.sel)

# Seleccionamos los ID
snpset.id <- unlist(snpset)

# Calculamos la relacion entre las muestras
ibd<-snpgdsIBDMoM(gds, kinship = T, snp.id = snpset.id, sample.id = sample.sel,
                  num.thread = 1, verbose = FALSE)
seqClose(gds)

# Seleccionamos las muestras
ibd.sel<-subset(snpgdsIBDSelection(ibd), kinship>0.1)

# Seleccionamos las muestras
filt.rela<-as.integer(related(ibd.sel))

# Las aplicamos al filtro
sample.sel=sample.sel[!sample.sel%in%filt.rela]
```

Se aprecia que el número de muestras relacionadas es 6.

```
gds <- seqOpen(gds.file)

# Realizamos el análisis de los componentes principales
pca <- snpgdsPCA(gds, snp.id=snpset.id,sample.id = sample.sel,
                 num.thread=2, verbose=FALSE)

seqClose(gds)
```

Filtrado de las muestras por los componentes principales Vemos que a partir de 2 componentes principales la variabilidad recogida es baja. Podemos representar estas dos primeras componentes

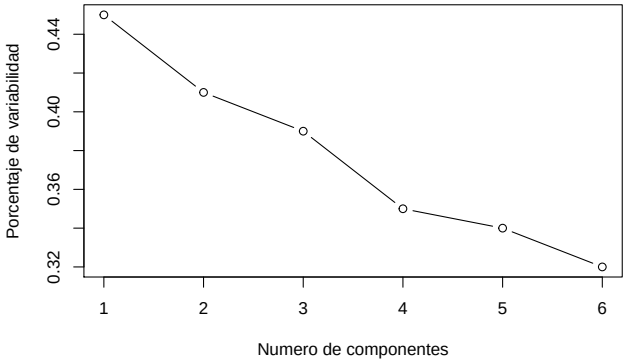

Figura 6: Aportacion de cada componente principal

```
gds <- seqOpen(gds.file)

# Aplicamos el filtro
seqSetFilter(gds, sample.id=sample.sel, variant.id=snpset.id, verbose=F)

# Extraemos el fenotipo
pheno_code <- seqGetData(gds,"sample.annotation/phenotype")
sexo <- seqGetData(gds,"sample.annotation/sex")

seqClose(gds)

# Creamos una tabla
tab <- data.frame(sample.id = pca$sample.id,
    pheno = factor(pheno_code)[match(pca$sample.id, sample.id)],
    sexo = factor(sexo)[match(pca$sample.id, sample.id)],
    EV1 = pca$eigenvect[,1],
    EV2 = pca$eigenvect[,2],
    EV3 = pca$eigenvect[,3],
    EV4 = pca$eigenvect[,4],
    EV5 = pca$eigenvect[,5],
    EV6 = pca$eigenvect[,6],
    EV7 = pca$eigenvect[,7],
    EV8 = pca$eigenvect[,8],
    EV9 = pca$eigenvect[,9],
    EV10 = pca$eigenvect[,10],
    stringsAsFactors = FALSE)
```

Vemos en el siguiente gráfico que existen algunos casos que parecen algo extremos

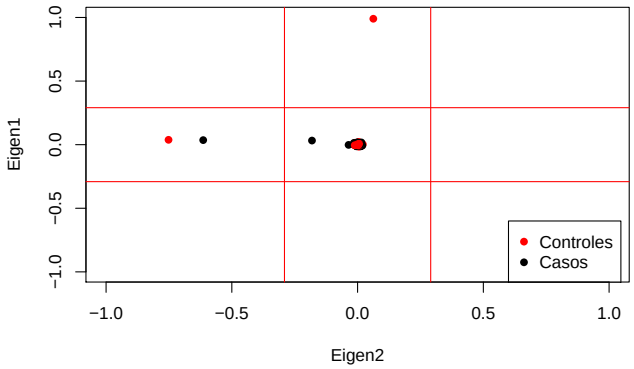

Figura 7: Componentes principales

Eliminaremos aquellas muestras que sobrepasan 6 desviaciones típicas de la media en alguno de los componentes principales. Aplicaremos este método.

```
# Podríamos computar el valor para cada eigenvector pero son todos iguales
#apply(pca$eigenvect,2, function (x) mean(x)+6*sd(x))

# Calculamos el límite
valor<-mean(pca$eigenvect[,1])+6*sd(pca$eigenvect[,1])

# Las muestras eliminadas son
flag<-ifelse(apply(abs(tab[,4:13])>valor,1, sum)==0, TRUE, FALSE)
filt.pca<-tab[!flag,1]

# Seleccionamos de la tabla dichos registros
sample.sel<-sample.sel[!sample.sel%in%filt.pca]

gds <- seqOpen(gds.file)

# Volvemos a computar los componentes principales con las muestras seleccionadas
pca <- snpgdsPCA(gds, sample.id = sample.sel, snp.id=snpset.id,
                 num.thread=2, verbose = F)

seqClose(gds)
```

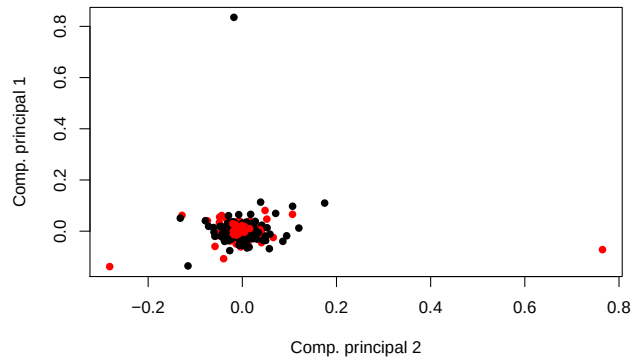

Figura 8: Grafico de componentes principales corregido

Filtrado de los SNP: segunda parte

Si la frecuencia de los genotipos observados de una variante en una poblacion puede ser derivada de las frecuencias observadas de los alelos, se dice que la variante genética está en equilibrio de Hardy-Weinberg. El test de Hardy Weimberg se utiliza para filtrar errores de genotipado. Este equilibrio se tiene que hacer solo sobre los controles, en aquellos cuya p<0.000001

```
# Establecemos los límites
p.valor.hwe=0.00001
log.p.hwe<--log10(p.valor.hwe)

gds <- seqOpen(gds.file)

# Aplicamos el filtro con las muestras seleccionadas
seqSetFilter(gds, sample.id=sample.sel, variant.id=snp.sel, verbose=FALSE)

# Extraemos el fenotipo
fenotipo<- seqGetData(gds, "sample.annotation/phenotype")

# Seleccionamos a continuacion solo a los controles
seqSetFilter(gds, sample.id=sample.sel[fenotipo==1],
             variant.id=snp.sel, verbose=FALSE)

# Calculamos el valor del hwe
hw <- hwe(gds)

# Eliminamos aquellos valores que excedan el p.valor
filt.snp.hwe<-hw[which(hw$p<p.valor.hwe),1]

# Seleccionamos aquellos SNPs que cumplen el criterio
snp.sel<-snp.sel[!snp.sel%in%filt.snp.hwe]



# Computamos el valor de p
pval <- -log10(sort(hw$p))

# Calculamos el HWE permutado
hw.perm <- hwe(gds, permute=TRUE)
seqClose(gds)

# Calculamos el valor de p
x <- -log10(sort(hw.perm$p))
```

```
# Podemos realizar el mismo gráfico con los dos, cromosomas autosómicos y los
# sexuales
gds <- seqOpen(gds.file)

# Seleccionamos a continuacion solo los genes de los cromosomas autosómicos
seqSetFilter(gds, sample.id=sample.sel[fenotipo==1],
             variant.id=snp.sel[!chr%in%c("X", "Y", "XY", "MT")],
             verbose=FALSE)

# Calculamos el valor del hwe
hw <- hwe(gds)
```

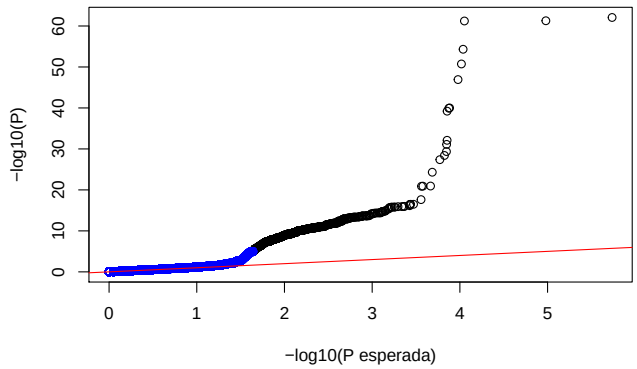

Figura 9: Equilibrio de Hardy Weinberg

```
# Computamos el valor de p
pval2 <- -log10(sort(hw$p))

# Calculamos el HWE permutado
hw.perm <- hwe(gds, permute=TRUE)

seqClose(gds)

# Calculamos el valor de p
x2 <- -log10(sort(hw.perm$p))
```

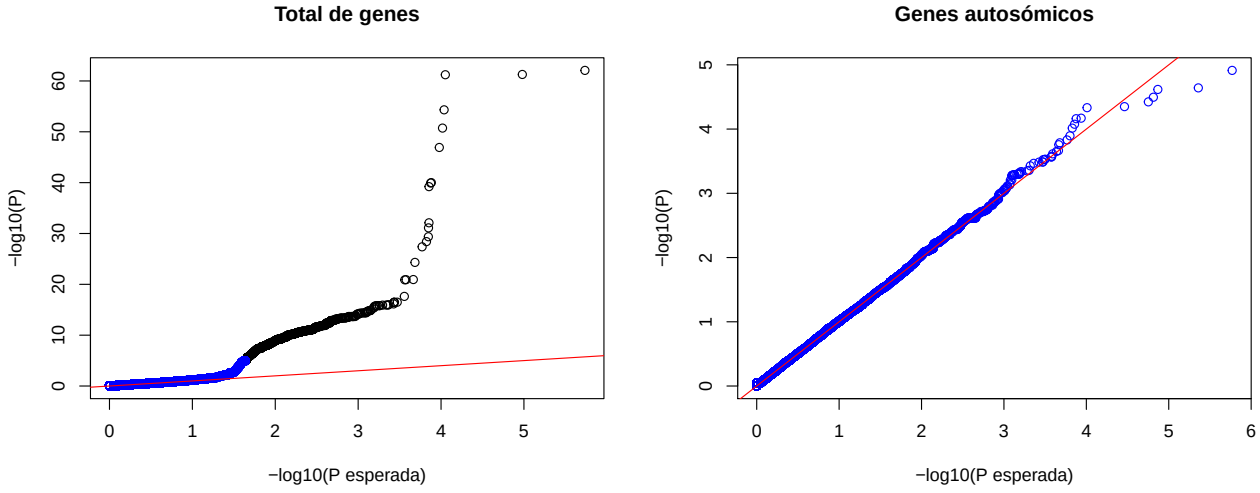

Figura 10: Equilibrio de Hardy Weinberg total

Otros análisis del control de calidad

Podemos realizar un análisis de otros parámetros.

La relación de transicion/transversion por lo general suele ser una medida de la calidad de los datos. Por lo general en los estudios de genoma humano el ratio suele encontrarse alrededor de 3

Otro parámetro es el porcentaje de datos que se encuentran en equilibrio de genes heterocigotos y homocigotos, que debe seguir una distribución normal

```
gds <- seqOpen(gds.file)

seqSetFilter(gds, sample.id=sample.sel, variant.id=snp.sel, verbose=FALSE)
# El ratio transicion/transversion
titv(gds)
```

```
## [1] 3.159536
```

```
# Estimamos el ratio transicion transversion por muestra
titv.s<-titv(gds, by.sample=TRUE)

# Podemos calcular tambien el grado de homocigosidad
hethom <- hethom(gds)

seqClose(gds)
```

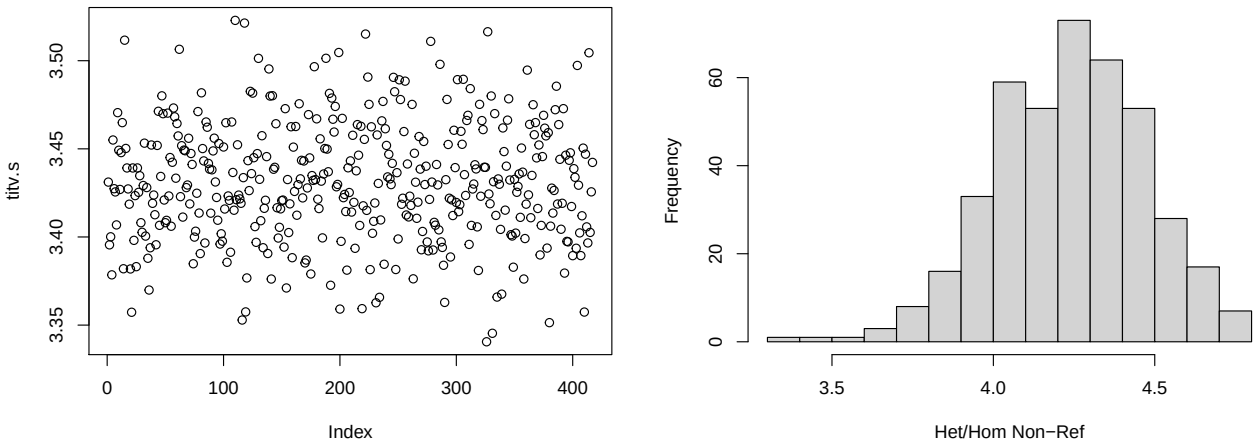

Figura 11: Indice transicion transversion y hetero/homocigosidad

Recapitulación del control de calidad

De una muestra inicial de 464 muestras con 319950 SNPs:

- Eliminamos 5511 SNPs por presentar criterios de MAF y call rate que ya detallamos con anterioridad. Nos quedan 314439 SNPs
- Eliminamos 1 muestras por presentar un call rate inferior al 98 %. Nos quedan 463
- Eliminamos 20 muestra por presentar discordancia entre el sexo cromosómico y el reportado. Nos quedan 443 muestras
- Eliminamos 7 muestras por presentar un índice de heterocigosidad fuera de los límites establecidos. Nos quedan 436 muestras
- Eliminamos 3 muestras por presentar alta endogamia y 6 presentar correlacion entre ellos. Nos quedan 427 muestras
- Eliminamos 10 muestras por ser extremos en los componentes principales. Nos quedan 417
- Se eliminaron 1512 SNPs por presentar valores extremos del equilibrio de Hardy Weinberg en los controles. Por lo tanto nos quedan 312927

Para el análisis final disponemos de 312927 SNPs para el análisis y 417 muestras

Análisis estadístico

Creando componentes principales que capturen la subestructura de la poblacion

Una vez que tenemos los datos con los que vamos a trabajar podemos en primer lugar recoger la variabilidad de los datos

```
gds <- seqOpen(gds.file)

set.seed(1000)

# Seleccionamos
snpSub<-snpgdsLDpruning(gds, ld.threshold = 0.1,
                        sample.id=sample.sel,
                        snp.id=snp.sel, verbose = FALSE)

snpset<-unlist(snpSub)

# Calculamos los componentes principales
pca<-snpgdsPCA(gds, sample.id=sample.sel, snp.id = snpset,
               num.thread = 4, verbose=FALSE)

seqClose(gds)

# Los guardamos
pca<-data.frame(sample.id=as.integer(pca$sample.id), pca$eigenvect[,1:10])

# Cambiamos el nombre
colnames(pca)[2:11]<-paste("pc", 1:10, sep = "")
```

Preparación de los datos

En primer lugar debemos adoptar todos los datos proporcionados para así ajustar los datos

```
# Cargamos el archivo con los datos fenotípicos
scan<-read.xlsx(file.path(file_tabla, "scan.xlsx"))

# Cargamos los datos de anotación en la memoria
anotacion<-read.csv(file.path(file_tabla, "anot.csv"), sep=",")

# Renombramos algunas variables
names(scan)[names(scan) == 'Sample_ID'] <- 'sample.id'
scan$sample.id<-as.integer(substring(scan$sample.id, 1,9))

# Fusioamos los datasets
scan<-merge(scan, pca, all.x = TRUE, by="sample.id")

gds <- seqOpen(gds.file)

# Obtenemos los ids
ids<-seqGetData(gds, "sample.id")
scan<-scan[match(ids, scan$sample.id),]

# Recodificamos los datos
scan$Status<-recode(scan$Status, "1"="0", "2"="1")
```

```
scan$Status<-factor(scan$Status)
scan$Sex <- as.factor(scan$Sex)

seqClose(gds)
```

Una vez que hemos realizado estos pasos inciales de cargado de los datos y de recategorización, el siguiente paso es convertirlos en un formato para el análisis. Por lo general se suele usar la estructura de `SeqVarData` del paquete `GWAStools` sobre el que se pueden aplicar diversos test estadísticos

```
# Lo guardamos como un AnnotatedDataframe
sample.data <- AnnotatedDataFrame(scan)

gds <- seqOpen(gds.file)

# Generamos un dato de secuenciacion
seqData <- SeqVarData(gds, sample.data)
```

Computación del modelo

Una vez hecho esto aplicaremos la función regression, que ajusta un modelo linear con los SNPs que tenemos seleccionados para intentar predecir en cada muestra el estatus del paciente. Se suele añadir como covariables a los componentes principales, pues suelen recoger la estructura de la población y nos permiten ajustar los datos

```
# Volvemos a seleccionar el filtro para seleccionar solo aquellas muestras
# que nos interesan y los SNPs
seqSetFilter(gds, sample.id=sample.sel, variant.id=snp.sel, verbose=FALSE)

# Computamos la regresión logística de los datos intentando
# predecir el estatus

assoc <- regression(seqData, outcome="Status", covar=colnames(scan)[c(4, 16:25)],
                    model.type="logistic", parallel=T)

seqClose(gds)
```

Este paso suele ser computacionalmente bastante costoso, por lo que puede que tarde un rato en ejecutarse.

Para tomar un punto de corte en la significación estadística haremos el ajuste de Bonferroni, pues aunque es un ajuste conservador, es el que suele emplearse en los análisis de GWAS

```
# El valor de p segun Bonferroni
p.cutoff=0.05/nrow(assoc)
```

Variantes significativas

En primer lugar seleccionamos aquellas variantes que tienen una significación estadística deseada

```
# Podemos seleccionar la lista de SNP
lista.snps<-assoc[which(assoc$Wald.Pval<p.cutoff),1]

# Cambiamos los nombres al estandar

transf_nombres<-function(lista){
  lista<-str_remove(lista, "exm-")
  lista<-str_remove(lista, "MS_Replication_Chip_")
  lista<-str_remove_all(lista, paste(paste0("_ver", 1:4), collapse = "|"))
  lista<-str_remove_all(lista,paste(paste0("d", 1:4, "__"), collapse = "|"))
  return(lista)
}

lista.snps<-transf_nombres(lista.snps)
```

A continuacion crearemos una tabla que muestre los resultados encontrados

```
# Calculamos algunos parámetros
df<-assoc[which(assoc$Wald.Pval<p.cutoff),]

# Podemos calcular la ODDs ratio
df$OR<-1/exp(df$Est)/(1+exp(df$Est))

# Su intervalo de confianza
df$lower<-(1+exp(df$Est+2*df$SE))/exp(df$Est+2*df$SE)
df$upper<-(1+exp(df$Est-2*df$SE))/exp(df$Est-2*df$SE)

# El valor de p inverso
df$p<--log10(df$Wald.Pval)

gds <- seqOpen(gds.file)

# Buscamos los alelos
seqSetFilter(gds, sample.id=sample.sel, variant.id=df$variant.id, verbose=F)
df$alelo<-seqGetData(gds, "allele")

# Transformamos los nombres
df$variant.id<-transf_nombres(df$variant.id)

# Ordenamos los datos
df<-df[order(df$p, decreasing = T),]

# Eliminamos los repetidos
df<-df[!duplicated(df$variant.id),]

seqClose(gds)
```

Cuadro 1: Tabla de SNPs significativos en el análisis

SNP ID	N_0	N_1	$Freq_0$	$Freq_1$	Estimacion	Error est	Estadistico Wald	P valor	OR	-Log(p)	Alelo
rs7197	207	210	0.89	0.74	-1.11	0.21	29.13	1e-07	2.28 [3.01-5.58]	7.17	G,A
rs3135388	207	210	0.94	0.81	-1.33	0.25	28.33	1e-07	2.99 [3.29-7.23]	6.99	G,A
rs3129889	207	210	0.94	0.81	-1.33	0.25	28.33	1e-07	2.99 [3.29-7.23]	6.99	A,G
rs3117116	207	210	0.93	0.80	-1.29	0.25	27.74	1e-07	2.86 [3.23-6.96]	6.86	A,G
rs3129860	207	210	0.93	0.80	-1.29	0.25	27.74	1e-07	2.86 [3.23-6.96]	6.86	G,A
rs3129865	207	210	0.93	0.80	-1.29	0.25	27.74	1e-07	2.86 [3.23-6.96]	6.86	G,C
rs3129868	207	210	0.93	0.80	-1.29	0.25	27.74	1e-07	2.86 [3.23-6.96]	6.86	C,A
rs3135350	207	210	0.94	0.81	-1.33	0.25	27.73	1e-07	3 [3.29-7.29]	6.86	A,G
rs3135352	207	210	0.94	0.81	-1.31	0.25	27.63	1e-07	2.93 [3.26-7.14]	6.83	A,C
rs3129971	207	210	0.94	0.81	-1.31	0.25	27.63	1e-07	2.93 [3.26-7.14]	6.83	C,G
rs9268635	207	210	0.94	0.81	-1.31	0.25	27.63	1e-07	2.93 [3.26-7.14]	6.83	G,A

Para poder estudiar estos genes podemos hacer una query a la base de datos del NCBI. Solo ejecutaremos este fragmento si existe conexion a internet

```
# Podemos hacer queries a la base de SNP del NCBI
anotaciones_ncbi<-ncbi_snp_query(lista.snps)


# Otra opcion es realizar las queries a biomart
snpmart = useEnsembl(biomart = "snp", dataset="hsapiens_snp")


datos.genes<-getBM(attributes = c('refsnp_id','allele',
                     'associated_gene', "phenotype_description",
                      "title"),
      filters ="snp_filter",
      values = lista.snps,
      mart = snpmart)
```

Más adelante consultaremos los datos de estas tablas, aunque antes realizaremos una representación gráfica de los datos.

Representaciones gráficas

Gráfico Q-Q plot

En el grafico QQ comparamos los valores de p obtenidos frente a una distribución teórica de los valores de p dado el número de contrastes. Vemos que existen bastantes puntos que se alejan de la recta, si bien no alcanzan la significación estadística

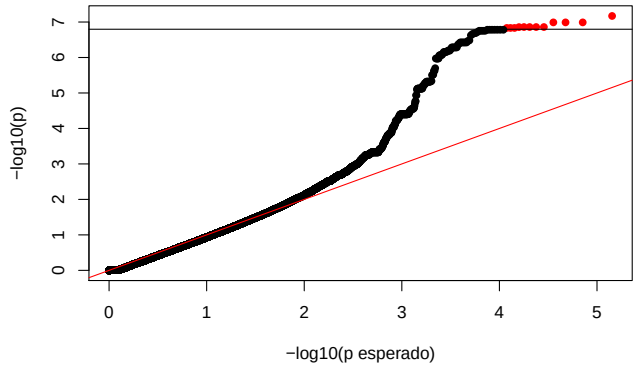

Figura 12: QQ plot

Manhattan plot

El Manhattan plot es un gráfico típico de los estudios GWAS en el que se muestra la secuencia de SNPs en el eje x y en el eje y el valor de p, donde cada punto es un SNP. Este gráfico permite localizar donde se encuentran los SNPs significativos en un estudio GWAS

```
# Cambiamos el nombre de los SNPs
anotacion$Name<-transf_nombres(anotacion$Name)
assoc$variant.id<-transf_nombres(assoc$variant.id)

# Cambiamos el nombre de las columnas
```

```
anotacion<-anotacion[,c("Name", "Chr", "MapInfo")]

# Unimos ambos dataset
assoc<-merge(assoc, anotacion, by.x="variant.id", by.y="Name", all.x=T)
```

Para realizar el Manhattan plot puede simplemente ser necesario una instrucción si por ejemplo usamos la funcion `manhattanPlot()` contenida en el paquete `GWAStools`

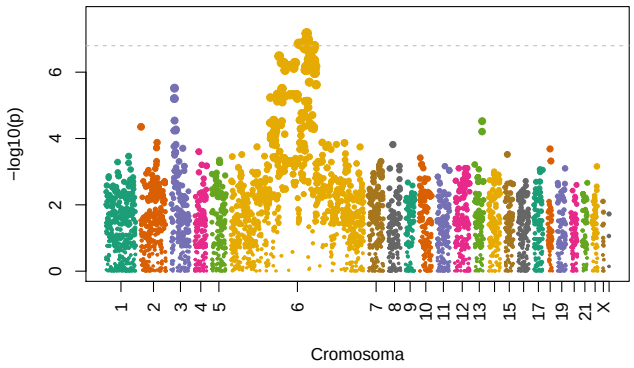

Figura 13: Manhattan plot global

Podemos apreciar que solo son significativos genes localizados en el cromosoma 6. Esto no es de extrañar ya que es donde se encuentra el gen del complejo mayor de histocompatibilidad, uno de los genes más asociados con la EM.

Una opción puede ser realizar un análisis de los datos centrado en esta región cromosómica.

```
# Seleccionamos aquellos SNPs localizados en el cromosoma 6
chrom6<-assoc[assoc$Chr==6,]

# Creamos un nuevo dataset
dp<-chrom6 %>%
  # Agrupamos por cromosoma
    group_by(Chr)%>%
  # Calculamos cual es el máximo de PB en ese cromosoma
    summarise(chr_len=max(MapInfo))%>%
  # Calculamos la suma acumulada y le restamos el maximo
    mutate(tot=cumsum(chr_len)-chr_len) %>%
  # Eliminamos la longitud maxima del cromosoma
    dplyr::select(-chr_len) %>%
  # Hacemos un merge con el dataset con los datos del cromosoma 6
    left_join(chrom6, ., by=c("Chr"="Chr")) %>%
  # Ordenamos los datos por numero de cromosoma y PB
    arrange(Chr, MapInfo) %>%
  # Creamos una columna acumulativa donde sumemos la posicion y la suma
    mutate( BPcum=MapInfo+tot) %>%
  # Creamos una columna para saber qué SNPs vamos a remarcar
    mutate(is_highlight=ifelse(!is.na(Wald.Pval)&
                                 (-log10(Wald.Pval)>-log10(p.cutoff)),
                               TRUE, FALSE))

# Creamos el punto medio
axisdf <- dp %>% group_by(Chr) %>% summarize(center=( max(BPcum) + min(BPcum) ) / 2 )
```

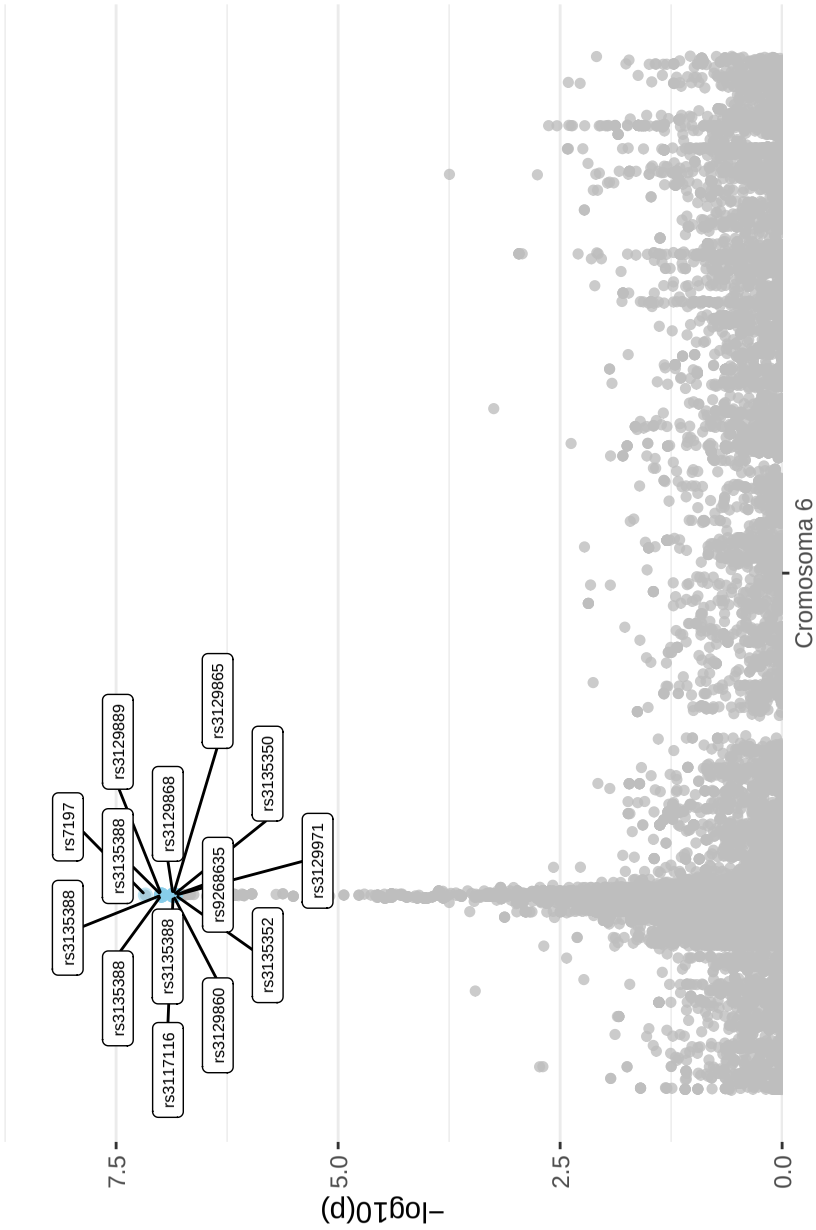

Figura 14: Manhattan plot cromosoma 6

Linkage dysequilibrium matrix

Las matrices de desequilibrio de ligamiento permiten saber si determinados genes estan asociados entre sí por encontrarse cerca en el genoma y además permiten obtener más información sobre los SNPs

```
# Para realizar este análisis tenemos que usar la funcion read.plink
# contenida en el paquete SNPStats
nuevo<-read.plink(bed, bim, fam)

# Seleccionamos el genotipo
genotipo<-nuevo$genotypes

# Cambiamos los nombres de los SNPs al estandar
colnames(genotipo)<-transf_nombres(colnames(genotipo))

# Eliminamos los que tienen nombre duplicado
genotipo<-genotipo[,!duplicated(colnames(genotipo))]
```

En primer lugar podemos representar los genes que son significativos

```
# Podemos seleccionar la lista de genes
gen_sel<-assoc[which(assoc$Wald.Pval<p.cutoff),]

gen_sel<-arrange(gen_sel, MapInfo)

# Calculamos cual es el maximo y el minimo en pb de los genes
minimo<-min(gen_sel$MapInfo)-100
maximo<-max(gen_sel$MapInfo)+100

# Seleccionamos aquellos genes que pertenecen al cromosoma 6 (porque es
# el único cromosoma donde tenemos SNPs significativos) y que se encuentran
# entre nuestras pb seleccionadas
selec<-assoc[assoc$Chr=="6"&between(assoc$MapInfo, minimo, maximo), ]

# Eliminamos los duplicados
selec<-selec[!duplicated(selec$variant.id),]

# Ordenamos los datos seleccionados
selec<-arrange(selec, MapInfo)

# Seleccionamos aquellos SNPs en el orden que hemos realizado
chr6_sel<-genotipo[,order(match(colnames(genotipo),selec$variant.id))]

# Sustituimos los nombre en caso de que posteriormente queramos representarlo
nombres<-ifelse(selec$variant.id%in%lista.snps, selec$variant.id, "")
```

Una vez hecho esto ya podemos representar los datos que se encuentran entre los SNPs

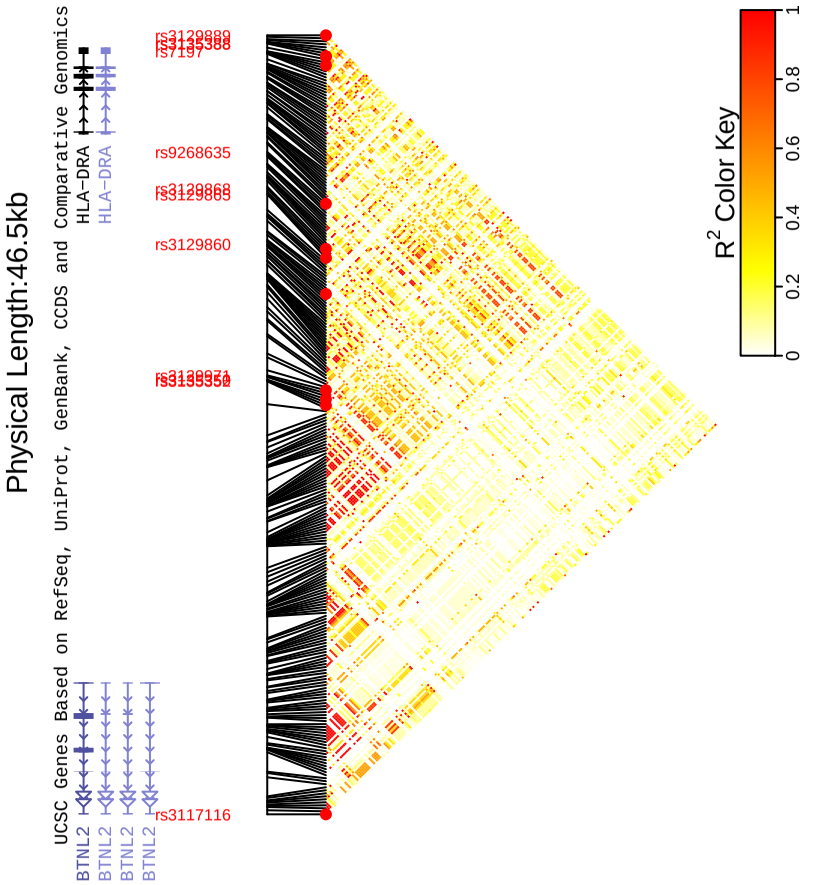

Figura 15: Desequilibrio de ligamiento del cromosoma 6

Otra opción es representar solo los datos del SNP que han salido significativos

```
# Ordenamos por pares bases
# esto lo hacemos porque todos estan en el cromosoma 6
gen_sel<-arrange(gen_sel, MapInfo)

# Seleccionamos aquellos genoticos que concuerdan con nuestras anotaciones
geno_sel<-genotipo[,order(match(colnames(genotipo),gen_sel$variant.id))]
```

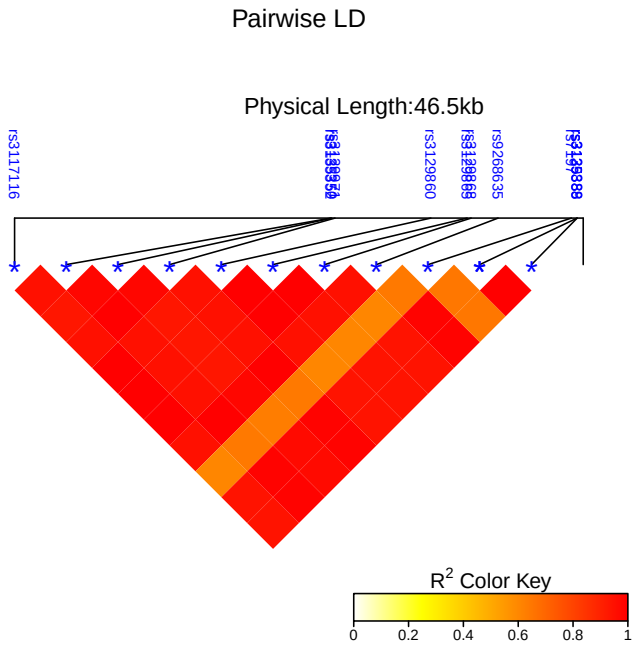

Figura 16: Desequilibrio de SNPs significativos

Análisis de ontología y enriquecimiento

Análisis de listas de SNPs y de genes

Para realizar el análisis de ontología y enriquecimiento usaremos el paquete XGR que presenta una interfaz web para la realizacion de los diferentes cálculos

```
# Seleccionamos los nombres y los valores de p
assoc$variant.id<-transf_nombres(assoc$variant.id)

# Ordenamos por el valor de p
data<-assoc[order(assoc$Wald.Pval, decreasing=F),]

# Eliminamos los duplicados
data<-data[!duplicated(data$variant.id),]

# Seleccionamos los significantes
data.sig.sel<-data[which(data$Wald.Pval<p.cutoff),c("variant.id", "Wald.Pval")]
data.sig.all<-data[which(data$Wald.Pval<p.cutoff),]

# Seleccionamos como matriz
data.mat<-as.matrix(data)
```

```
# Generamos un vector de SNPs
vector<-data.mat[,1]
vector.sig<-data.sig.sel[,1]
```

```
# Podemos dar un score a cada SNPs
resultados<-xSNP2GeneScores(data=data.sig.sel,
                            # Podemos seleccionar la poblacion
                            # con poblacion IBS no funciona bien
                            include.LD='EUR', LD.r2=0.8,
                            # Seleccionamos el límite de significacion deseado
                            # Si somos poco estrictos
                            significance.threshold=p.cutoff,
                            # Si somos muy estrictos
                            #significance.threshold=p.cutoff,
                            # La distancia máxima
                            distance.max=50000,
                            decay.kernel="rapid", decay.exponent=2,
                            GR.SNP="dbSNP_GWAS", GR.Gene="UCSC_knownCanonical",
                            scoring.scheme="max", verbose=F)
```

Cuadro 2: Genes encontrados

Gen	Score	P valor
HLA-DRA	0.37	0.42
BTNL2	0.08	0.83
HCG23	0.05	0.89
C6orf10	0.01	0.97
HLA-DRB6	0.00	0.99
HLA-DRB5	0.00	0.99
HLA-DQA1	0.00	1.00

```
# A continuacion con los genes obtenidos podemos hacer un analisis de
# enriquecimiento
ex<-xSubneterGenes(resultados$Gene, network = "STRING_low", verbose=FALSE)
```

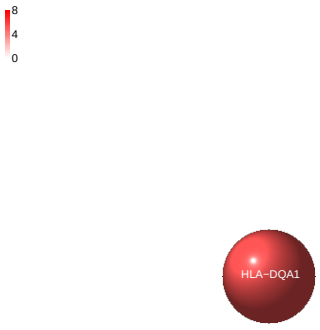

Figura 17: Representación gráfica del análisis de enriquecimiento de los genes

```
# Analisis del enriquecimiento de todos los genes
enriquec<-xEnricherSNPs(data = vector.sig, ontology ="EF",
                        include.LD = "EUR",LD.r2 = 0.8)
# No se obtuvo ningun resultado
```

Podemos probar a realizar un enriquecimiento de los diferentes genes con diferentes ontologías

```
ontologia<-"MP"

# Otras ontologías
#  -Reactome: "MsigdbC2REACTOME"
#  -MsigdbC2CPall
#  -Funcion molecular: "GOMF"
#  -Procesos biologicos: "GOBP"
#  -Anormalidades fenomicas: "HPPA"
#  -Fenotipos de mamiferos:  "MP"

eTerm <- xEnricherGenes(data=resultados$Gene$Gene, ontology=ontologia)

# Ninguna de las ontologías estudiadas dio resultados
```

Sin embargo en nuestro análisis, no dio ningun resultado

Otra opcion es hacer el análisis a partir de las diferentes anotaciones

```
# Transformamos a objeto GRanges todos los SNPs significativos
seleccion <- GRanges(
    seqnames = Rle(paste0("chr", data.sig.all$Chr)),
    ranges = IRanges(data.sig.all$MapInfo),
    Variant = data.sig.all$variant.id,
    Pvalue = data.sig.all$Wald.Pval)

# Preparamos los datos de todos los SNPs para transformarlo a un objeto GRanges

# Eliminamos los datos perdidos
data.na<-data[which(!is.na(data$MapInfo)),]
data.na<-data.na[which(!is.na(data.na$Wald.Pval)),]
data.na<-data.na[which(!is.na(data.na$Chr)),]

# Transformamos el cromosoma a string
data.na$Chr<-paste0("chr", data.na$Chr)

# Seleccionamos solo los datos que queremos
data.na<-data.na[,c("Chr", "MapInfo", "variant.id", "Wald.Pval")]

# Renombramos las variables
names(data.na)<-c("seqnames", "ranges", "Variant", "Pvalue")

# Creamos un objeto GRanges
todos = makeGRangesFromDataFrame(data.na, keep.extra.columns=TRUE,
                                 seqnames.field="seqnames", start.field="ranges",
                                 end.field="ranges")
```

```
ontology <- "GOMF"
eTerm_gomf_im <- xGRviaGeneAnno(data.file=seleccion,
                                background.file=todos, format.file="GRanges",
                                ontology=ontology, ontology.algorithm="lea")

# Con este método tampoco somos capaces de obtener ningun resultado
```

Análisis gráfico

Podemos realizar un análisis de las distancias a las que se encuentran los diferentes SNPs de acuerdo con la ontología "Experimental factor ontology".

```
# Calculamos los nodos (Tarda en ejecutarse)
social<-xSocialiserSNPs(data=vector)
```

También se puede calcular las distancias para los genes que hemos obtenido de análisis anteriores

```
# Podemos hacer lo mismo con la lista de genes significativos
social.genes<-xSocialiserGenes(data=resultados$Gene$Gene)
```

Otra opción es hacerlo solo con la lista de SNPs significativos, aunque en este caso por lo escaso de la lista no retorna ningun valor

```
# Calculamos las distancias para los SNPs significativos (Tarda en ejecutarse)
social.SNP<-xSocialiserSNPs(data=vector.sig,
                            # Aquí podemos especificar otros parámetros
                            # como el desequilibrio de ligamiento
                            include.LD = "EUR",LD.r2 = 0.8)
```

Una vez hecho esto podemos representar gráficamente los resultados

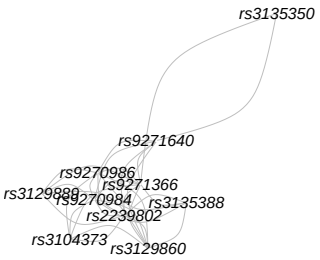

Figura 18: Distancias de los SNPs significativos

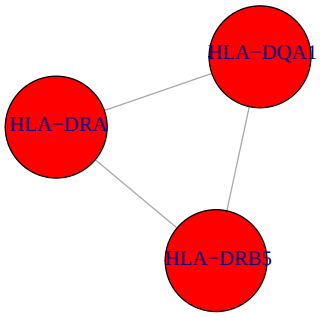

Figura 19: Distancias de los genes significativos

Podemos representar en un gráfico las relaciones de los SNPs encontrados significativos con el resto de SNPs incluidos en el array

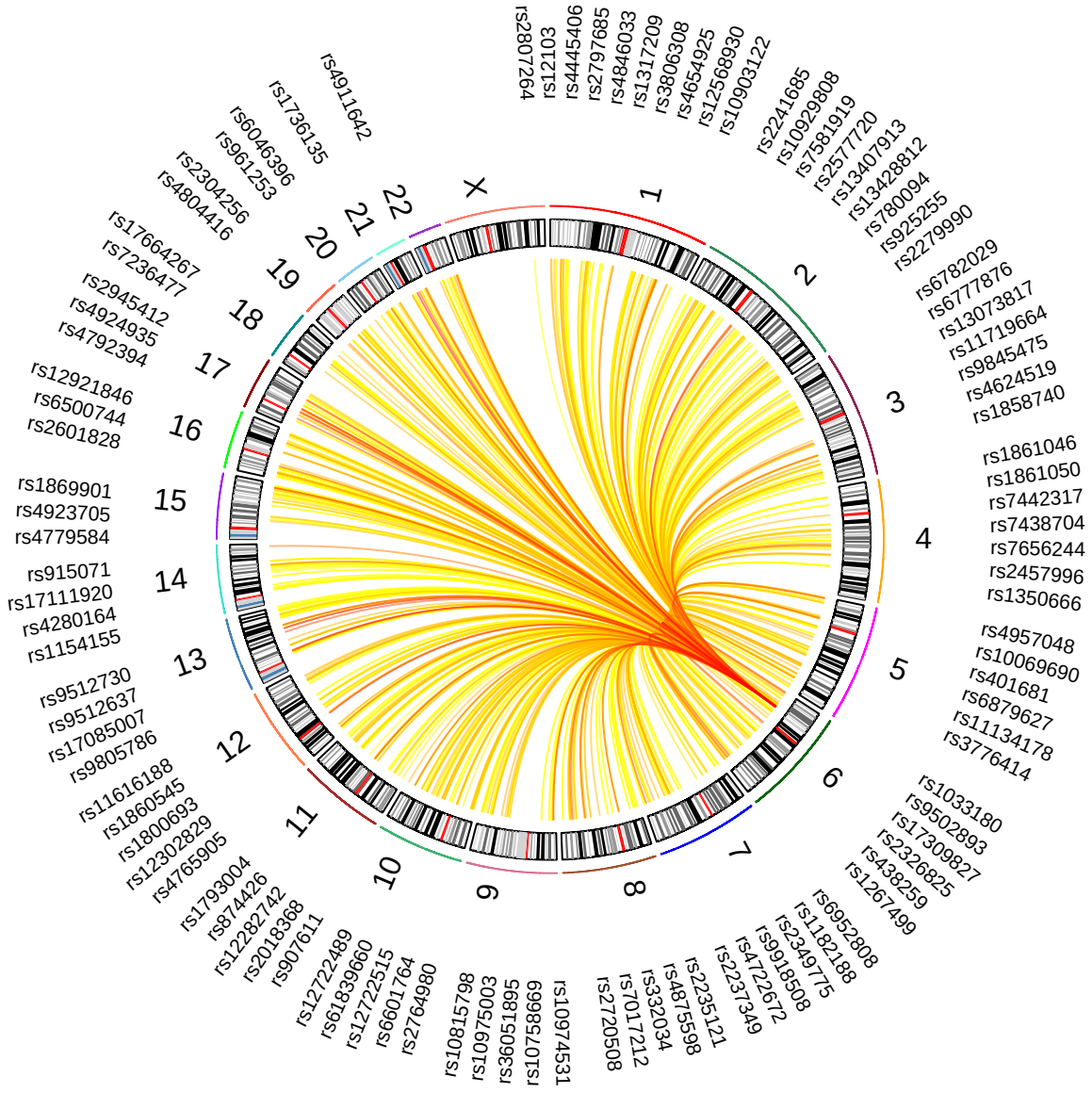

Figura 20: Gráfico circular

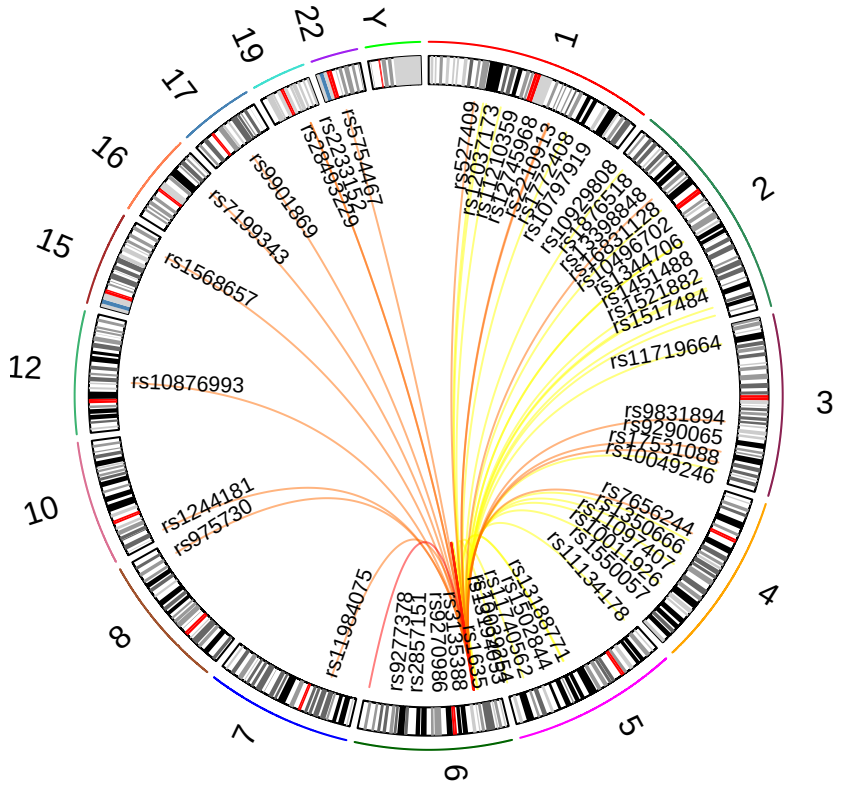

Figura 21: Gráfico circular del SNP rs3135388

Bibliografía

Reed, Eric, Sara Nunez, David Kulp, Jing Qian, Muredach P Reilly, and Andrea S Foulkes. 2015.“” *Statistics in Medicine* 34 (28): 3769–92. https://doi.org/10.1002/sim.6605.