Script análisis metilación ADN TFM

Belén Mollá Moliner

21/06/2021

Contents

Resumen de los pasos ejecutados en el pipeline para el análisis de metilación de ADN (Illumina EPIC methylation 850K array).

El código R utilizado para el análisis de la metilación ha sido extraído de:

- "MethylationArrayAnalysis: A cross-package Bioconductor workflow for analysing methylation array data" (Maksimovic J et al., 2017)
- y del TFM "Identification of biomarkers based on dna methylation for diagnosis and prognosis of neonatal sepsis" de Paula Navarrete López (Master Bioinformática Universidad de Valencia) cuyo trabajo ha sido recientemente publicado (Front Immunol 2021;12:622599).

Instalamos los paquetes requeridos para el análisis:

1 Librerias

```
> # Librerías...
> library(minfi)
> library(IlluminaHumanMethylationEPICmanifest)
> library(IlluminaHumanMethylationEPICanno.ilm10b4.hg19)
> library(limma)
> library(ggplot2)
> library(missMethyl)
> library(matrixTests)
```

2 Análisis de control de calidad de los datos crudos, normalización y filtrado de sondas

Para trabajar con los archivos .idat del array de metilación de Illumina utilizamos el paquete **minfi** de R. Definimos el rgSet `RGChannelSet` con los datos de la metilación (valores beta y M) y el phenoData:

```
> setwd(".")
> dir.create("data")
> dir.create("results")
> setPath = "."
> targets <- read.metharray.sheet(paste(setPath, "./data", sep = ""))
> rgSet <- read.metharray.exp(targets = targets)
> phenodata <- pData(rgSet)
```

Pasamos al control de calidad de las sondas, la normalización del `rgSet` y al filtrado de las sondas:

```
> # Nos quedamos con aquellos CpGs que tengan un p valor
+ # medio < 0.05 indicativo de buena calidad de la señal
> detP <- detectionP(rgSet)
> rgSet <- rgSet[, colMeans(detP) < 0.05]
> detP <- detP[, colMeans(detP) < 0.05]
> targets <- targets[colMeans(detP) < 0.05, ]
> # Normalización
> Funnorm <- preprocessFunnorm(rgSet)
> # Nos quedamos con las sondas que en todas las muestras
+ # tengan un p-value inferior a 0.01
> keep <- rowSums(detP.F < 0.01) == ncol(Funnorm)
> GRSetF.filt <- Funnorm[keep, ]
> # Nos quedamos con las sondas anotadas en el chrY o el chrX
> annEPIC <- getAnnotation(IlluminaHumanMethylationEPICanno.ilm10b4.hg19)
> keep <- !(featureNames(GRSetF.filt) %in% annEPIC$Name[(annEPIC$chr ==
+     "chrY") | (annEPIC$chr == "chrX")])
> GRSetF.filt <- GRSetF.filt[keep, ]
> # Eliminamos las sondas que coinciden con locis con CpG o
+ # SBE
> GRSetF.filt <- dropLociWithSnps(GRSetF.filt, snps = c("CpG",
+     "SBE"), maf = 0)
> # Nos quedamos con las sondas reactivas anotadas para array
+ # Illumina
> CrossReactiveProbes <- read.csv(setPath, "./data/13059_2016_1066_MOESM1_ESM.csv",
+     stringsAsFactors = FALSE)
> keep <- !(featureNames(GRSetF.filt) %in% CrossReactiveProbes$X)
> GRSetF.filt <- GRSetF.filt[keep, ]
```

Archivo EPIC/13059_2016_1066_MOESM1_ESM.csv desargado de https://github.com/sirselim/illumina450k_filtering}{GitHub:sirselim/illumina450k_filtering}.

El archivo original forma parte del material suplementario de Pidsley et al., (2016)[1], suggests cross-reactive and variant containing probes to filter at QC.

[1]Pidsley, R., Zotenko, E., Peters, T. J., Lawrence, M. G., Risbridger, G. P., Molloy, P., ... Clark, S. J. (2016). Critical evaluation of the Illumina MethylationEPIC BeadChip microarray for whole-genome DNA methylation profiling. Genome Biology, 17(1), 208. https://doi.org/10.1186/s13059-016-1066-1. The data generated as part of this study are available at the Gene Expression Omnibus (GEO) under accession GSE86833.

3 Análisis de metilación diferencial

A partir de los datos preprocesados (filtrados y normalizados) extraemos los datos de metilación (valores M) con la función **getM()** del paquete **minfi** de R.

```
> # obtención de valores M
> Mval <- getM(GRsetF)
```

3.1 Expresión diferencial de regiones metiladas (DMRs) entre SEPSIS y CONTROL (SC)

Se crea el factor **f** que va a dividir las muestras entre CONTROL y SEPSIS. Se define la **matriz de diseño** con los grupos. Se define la **matriz de contraste** con 1 contraste SEPSIS vs CONTROL. Utilizando los valores de M (Mval) se ajusta el modelo lineal usando la función **lmFit()** del paquete **limma** de R. Depués hacemos los contrastes entre los grupos que hemos definido en la matriz de contraste con la función **contrasts.fit()** y **eBayes()** y extraemos los resultados con **topTable()**:

```
> # factor con los grupos CONTROL (C) y SEPSIS (SV + SN + SVN)
> f <- factor(ifelse(GRsetF$Sample_Group == "C", "C", "S"))
>
> # Matriz de diseño SC
> design_SC <- model.matrix(~0+f)
> colnames(design_SC) <- c("C", "S")
> design_SC
>
> # Matriz de contraste
> cont.matrix <- makeContrasts(1= S - C, levels=design_SC)
>
> # Fit the linear model
> fit=lmFit(Mval, design_SC)
> # Estimación bayesiana (eBayes) de los contrastes
> fit.cont <- contrasts.fit(fit, cont.matrix)
> fit.cont <- eBayes(fit.cont)
> # significación estadística
> summa.fit <- decideTests(fit.cont)
> # Tabla resultados ordenados por p-valor
> limma_SC= topTable(fit.cont,coef=1,sort.by="p",n=Inf)
> # Guardamos resultados
> write.csv(limma_SC,file="limma_SC.csv",row.names=FALSE)
```

3.2 Expresión diferencial de regiones metiladas (DMRs) entre TODOS los grupos

```
> # factor con los grupos CONTROL (C) y SEPSIS (SV + SN + SVN)
> groups <- factor(GRsetF$Sample_Group))
>
> # Matriz de diseño SC
> design <- model.matrix(~0+f)
> colnames(design) <- levels(groups)
>
> # Matriz de contraste
> cont.matrix <- makeContrasts(1= SN - C, 2=SV - C, 3=SN - SV, levels = design_all)
>
> # Fit the linear model
```

```
> fit_all=lmFit(Mval, design)
> # Estimación bayesiana (eBayes) de los contrastes
> fit.cont_all <- contrasts.fit(fit_all, cont.matrix)
> fit.cont_all <- eBayes(fit.cont_all)
> # significación estadística
> summa.fit_all <- decideTests(fit.cont_all)
> summary(summa.fit_all)
> # Tabla resultados ordenados por p-valor
> limma_all_1= topTable(fit.cont_all,coef=1,sort.by="p",n=Inf)
> limma_all_2= topTable(fit.cont_all,coef=2,sort.by="p",n=Inf)
> limma_all_3= topTable(fit.cont_all,coef=3,sort.by="p",n=Inf)
> limma_all_4= topTable(fit.cont_all,coef=4,sort.by="p",n=Inf)
> limma_all_5= topTable(fit.cont_all,coef=5,sort.by="p",n=Inf)
> limma_all_6= topTable(fit.cont_all,coef=6,sort.by="p",n=Inf)
> # Guardamos resultados
> write.csv(limma_all_1,file="limma_all_1.csv",row.names=FALSE)
> write.csv(limma_all_2,file="limma_all_2.csv",row.names=FALSE)
> write.csv(limma_all_3,file="limma_all_3.csv",row.names=FALSE)
> write.csv(limma_all_4,file="limma_all_4.csv",row.names=FALSE)
> write.csv(limma_all_5,file="limma_all_5.csv",row.names=FALSE)
> write.csv(limma_all_6,file="limma_all_6.csv",row.names=FALSE)
```

Sólo son significativas con un p-valor 0.1 las comparaciones S-C y SN-C. Por ello guardamos sólo estos resultados y hacemos el DMRcate sobre estas comparaciones

3.3 Anotación DMP

Seleccionamos las columnas chr, pos, strand, Name, Probe.rs, islands_Name y UCSC_RefGene_name de la base de datos para EPIC según código de methytlArrayAnalisis de Bioconductor

```
> # anotación seleccionamos las columnas chr, pos, strand,
+ # Name, Probe.rs, islands_Name y UCSC_RefGene_nam
> annEPICsub <- annEPIC[match(rownames(getM(GRsetF_integracion_0.1)),
+     annEPIC$Name), c(1:4, 12, 18, 22)]
> DMP_integracion_0.1$SC <- topTable(fit2, num = Inf, coef = 1,
+     genelist = annEPICsub, p.value = 0.1)
> head(DMP_integracion_0.1$SC)
> dim(DMP_integracion_0.1$SC)
> write.csv(DMP_integracion_0.1$SC, file = "limma_SC_anotado.csv",
+     row.names = FALSE)
>
> # anotación seleccionamos las columnas chr, pos, strand,
+ # Name, Probe.rs, islands_Name y UCSC_RefGene_nam
> annEPICsub <- annEPIC[match(rownames(getM(GRsetF_integracion_0.1)),
+     annEPIC$Name), c(1:4, 12, 18, 22)]
> DMP_integracion_0.1$NC <- topTable(fit4, num = Inf, coef = 1,
+     genelist = annEPICsub, p.value = 0.1)
> head(DMP_integracion_0.1$NC)
> dim(DMP_integracion_0.1$NC)
>
> write.csv(DMP_integracion_0.1$NC, file = "limma_all_1_anotado.csv",
+     row.names = FALSE)
```

3.4 CpGs más metilados diferencialmente

Según código de methytlArrayAnalisis de Bioconductor

```
> # plot the top 4 most significantly differentially
+ # methylated CpGs
> bVals <- getBeta(GRsetF_integracion_0.1)
> groups <- GRsetF_integracion_0.1$Sample_Group
> png(paste(setPath, "/resultados_integracion_0.1/most_CpGs_NC_integracion_0.1.png",
+     sep = ""), width = 20, height = 20, res = 300, units = "cm")
> par(mfrow = c(2, 2))
> sapply(rownames(DMP_integracion_0.1$NC)[1:4], function(cpg) {
+     plotCpg(bVals, cpg = cpg, pheno = groups, ylab = "Beta values")
+ })
> dev.off()
>
> # plot the top 4 most significantly differentially
+ # methylated CpGs
> bVals <- getBeta(GRsetF_integracion_0.1)
> groups <- GRsetF_integracion_0.1$Sample_Group
> png(paste(setPath, "/resultados_integracion_0.1/most_CpGs_SC_integracion_0.1.png",
+     sep = ""), width = 20, height = 20, res = 300, units = "cm")
> par(mfrow = c(2, 2))
> sapply(rownames(DMP_integracion_0.1$SC)[1:4], function(cpg) {
+     plotCpg(bVals, cpg = cpg, pheno = groups, ylab = "Beta values")
+ })
> dev.off()
```

3.5 VOLCANO SEPSIS VS CONTROL

Según código Alex Sánchex Pla asignatura ómicas del master

```
> library(limma)
> library(minfi)
>
> # Design and contrast matrix
>
> f <- factor(ifelse(GRsetF_integracion_0.1$Sample_Group == "C",
+     "C", "S"))
> design_SC <- model.matrix(~0 + f)
> colnames(design_SC) <- c("C", "S")
>
> contrast.matrix_SC <- makeContrasts(S - C, levels = design_SC)
>
> # Linear fit
>
> fit <- lmFit(getM(GRsetF_integracion_0.1), design_SC)
> fit2 <- contrasts.fit(fit, contrast.matrix_SC)
> fit2 <- eBayes(fit2)
> results <- decideTests(fit2, adjust.method = "BH", p.value = 0.1)
>
> # Volcano plot c´pdgo pag 66 del modulo 2 de la asignatura
+ # omicas del master
> png(paste("./VOLCANO_integracion_0.1_S_C.png", sep = ""), width = 20,
+     height = 20, res = 300, units = "cm")
```

```
> coefnum = 1
> opt <- par(cex.lab = 0.7)
> volcanoplot(fit2, coef = coefnum, highlight = 20, names = fit2$ID,
+     main = paste("Differentially expressed genes", colnames(contrast.matrix_SC)[coefnum],
+         sep = "\n"))
> abline(v = c(-1, 1))
> par(opt)
> dev.off()
```

3.6 VOLCANO NOSOCOMIAL VS CONTROL

NEW 06/06/2021

```
> # Design and contrast matrix
> f <- factor(GRsetF_integracion_0.1$Sample_Group)
> design_SN <- model.matrix(~0 + f)
> colnames(design_SN) <- c("C", "SN", "SV")
>
> contrast.matrix_SN <- makeContrasts(SN - C, levels = design_SN)
>
> # Linear fit
> fit3 <- lmFit(getM(GRsetF_integracion_0.1), design_SN)
> fit4 <- contrasts.fit(fit3, contrast.matrix_SN)
> fit4 <- eBayes(fit4)
>
> # Volcano plot c´pdgo pag 66 del modulo 2 de la asignatura
+ # omicas del master
> png(paste("./VOLCANO_integracion_0.1_SN_C.png", sep = ""), width = 20,
+     height = 20, res = 300, units = "cm")
> coefnum = 1
> opt <- par(cex.lab = 0.7)
> volcanoplot(fit4, coef = coefnum, highlight = 20, names = fit2$ID,
+     main = paste("Differentially expressed genes", colnames(contrast.matrix_SN)[coefnum],
+         sep = "\n"))
> abline(v = c(-1, 1))
> par(opt)
> dev.off()
```