Datos normalizados

Camila Silva

2023-04-13

```
library("phyloseq")
library("ggplot2")
library("vegan")
```

```
## Loading required package: permute
```

```
## Loading required package: lattice
```

```
## This is vegan 2.6-4
```

```
#library("BiodiversityR")
library("RColorBrewer")
library("stringi")
library("dplyr")
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library("plyr")
```

```
## ------------------------------------------------------------------------------
```

```
## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)
```

```
## ------------------------------------------------------------------------------
```

```
##
## Attaching package: 'plyr'
```

```
## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

```
library("edgeR")
```

```
## Loading required package: limma
```

```
setwd("/home/camila/GIT/Tesis_Maestria/Data/fresa_solena/Data1")
outpath = "/home/camila/GIT/Tesis_Maestria/Analisis_Comparativo/Fresa_Solena/Results_img"
### Cargado de datos
fresa_kraken <- import_biom("fresa_kraken.biom")
colnames(fresa_kraken@tax_table@.Data) <- c("Kingdom", "Phylum", "Class", "Order", "Family", "Genus", "S
fresa_kraken@tax_table@.Data <- substr(fresa_kraken@tax_table@.Data,4,100)
colnames(fresa_kraken@otu_table@.Data) <- substr(colnames(fresa_kraken@otu_table@.Data),1,6)
metadata_fresa <- read.csv2("/home/camila/GIT/Tesis_Maestria/Data/fresa_solena/Data1/metadata.csv",head
fresa_kraken@sam_data <- sample_data(metadata_fresa)
fresa_kraken@sam_data$Sample<-row.names(fresa_kraken@sam_data)
colnames(fresa_kraken@sam_data)<-c('Treatment','Samples')
samples_to_remove <- c("MP2079","MP2080","MP2088","MP2109","MP2137")
fresa_kraken_fil <- prune_samples(!(sample_names(fresa_kraken) %in% samples_to_remove), fresa_kraken)
percentages_fil <- transform_sample_counts(fresa_kraken_fil, function(x) x*100 / sum(x) )
percentages_df <- psmelt(percentages_fil)
```

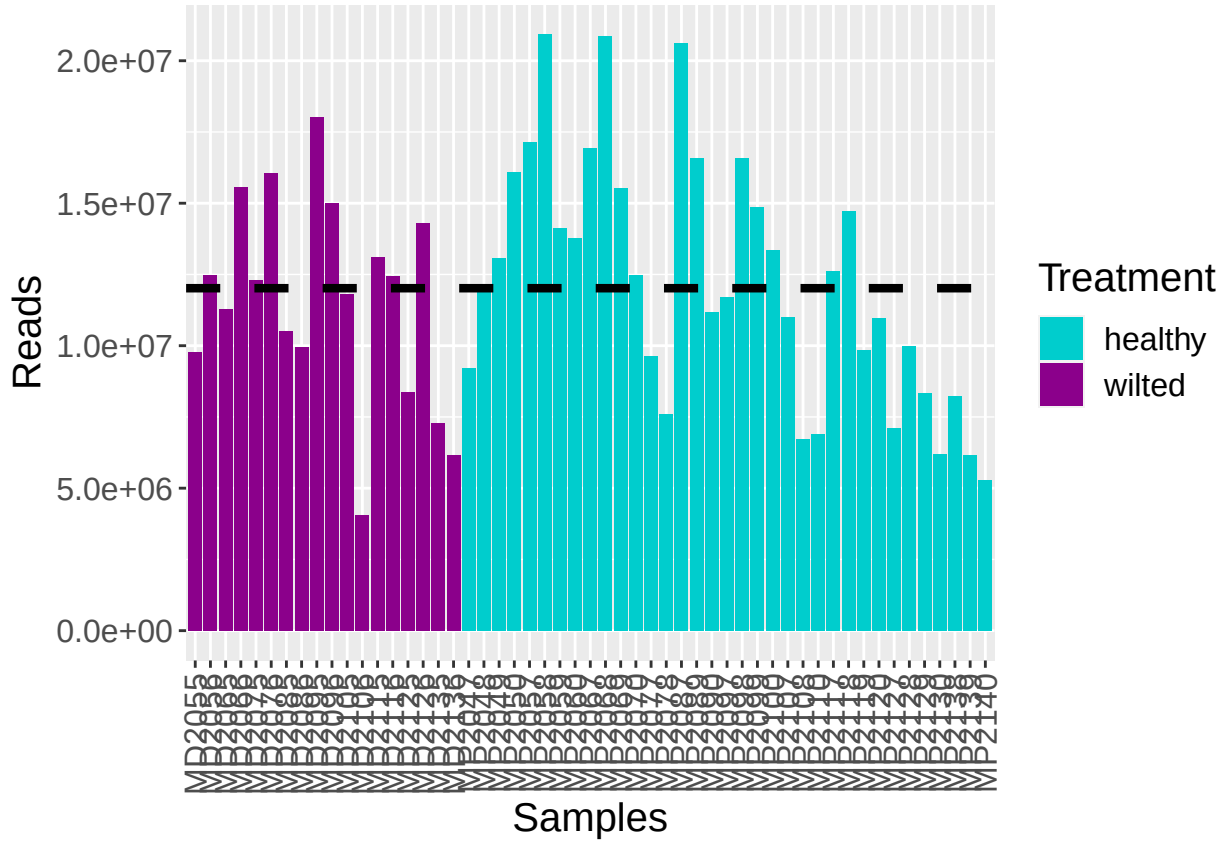
Queremos ver la profundidad de las muestras,

```
dprof <- data.frame(Samples = colnames(fresa_kraken_fil@otu_table@.Data),
                    Reads = sample_sums(fresa_kraken_fil),
                    Treatment = fresa_kraken_fil@sam_data@.Data[[1]])

mu_Samples <- mean(dprof$Reads)

ggplot(data = dprof, mapping = aes(x = Samples, y = Reads))+
  geom_bar(stat = "identity", aes( fill = Treatment)) +
  geom_hline(yintercept=mu_Samples, color = "black", linetype="dashed", size=1.5) +
  scale_fill_manual(values = c("cyan3","darkmagenta")) +
  theme(text = element_text(size = 15),
        axis.text.x = element_text(angle = 90, hjust = 1, vjust = 0.5))
```

```
## Warning: Using 'size' aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use 'linewidth' instead.
```



Podemos ver que nuestros datos tienen diferentes profundidades entre muestras. La línea marca la media de la profundidad de las muestras.

Esto deja en claro que los datos necesitan normalización. McMurdi et al. encontró una gran metodología para normalizar los datos. En su artículo Waste Not, Want Not: Why Rarefying Microbiome Data Is Inadmisible, hacen una interesante discusión sobre este tema constante en este tipo de análisis. Usaremos su metodología:

Metodo de Normalización

```
edgeRnorm = function(phy, ...){
  require("edgeR")
  require("phyloseq")
  if (!taxa_are_rows(phy)) {
    phy <- t(phy) #transpone el objeto phyoseq
  }
  x = as(otu_table(phy), "matrix") #convierte la tabla de abundancias en una matriz
  x = x + 1 #suma uno para evitar los ceros , y evitar errores
  ## SI DESDE ANTES SE HACE FILTRO DE CALIDAD, EN TEORIA YA NO HABRIAN CONTEOS EN CEROS...
  y = edgeR::DGEList(counts = x, remove.zeros = TRUE) #lo convierte en una Lista DGE
  #DGEList -> objeto a partir de una tabla de conteos, elimina los ceros
  z = edgeR::calcNormFactors(y, ...) #Realiza la normalización codificada por EdgeR, utilizando el méto
  #calcNormFactors -> Calcule los factores de normalización para escalar los tamaños de biblioteca sin
  #calcNormFactors: calcular factores de normalización para alinear columnas de una matriz de conteo
  #checamos que no dividimos por cero dentro de `calcNormFactors`.
  if (!all(is.finite(z$samples$norm.factors))) {
```

```
    stop("Something wrong with edgeR::calcNormFactors on this data, non-finite $norm.factors")
  }
  return(z)
}

# Cada una de las funciones de normalización toma un objeto phloseq y devuelve
# devuelve un objeto physeq cuya tabla otu se transforma.
# 1. norma de borde
#edgeRnorm -> Esta función escala datos NGS normalizados utilizando
#la función de normalización provista en edgeR.
z <- edgeRnorm(fresa_kraken_fil, method = "TMM")
# unimos z con el resto del objeto phyloseq
nor_fresa_kraken_fil <- merge_phyloseq(otu_table(z@.Data[[1]], taxa_are_rows = TRUE),
                         tax_table(fresa_kraken_fil@tax_table@.Data),
                         fresa_kraken_fil@sam_data)
rm(z)
```

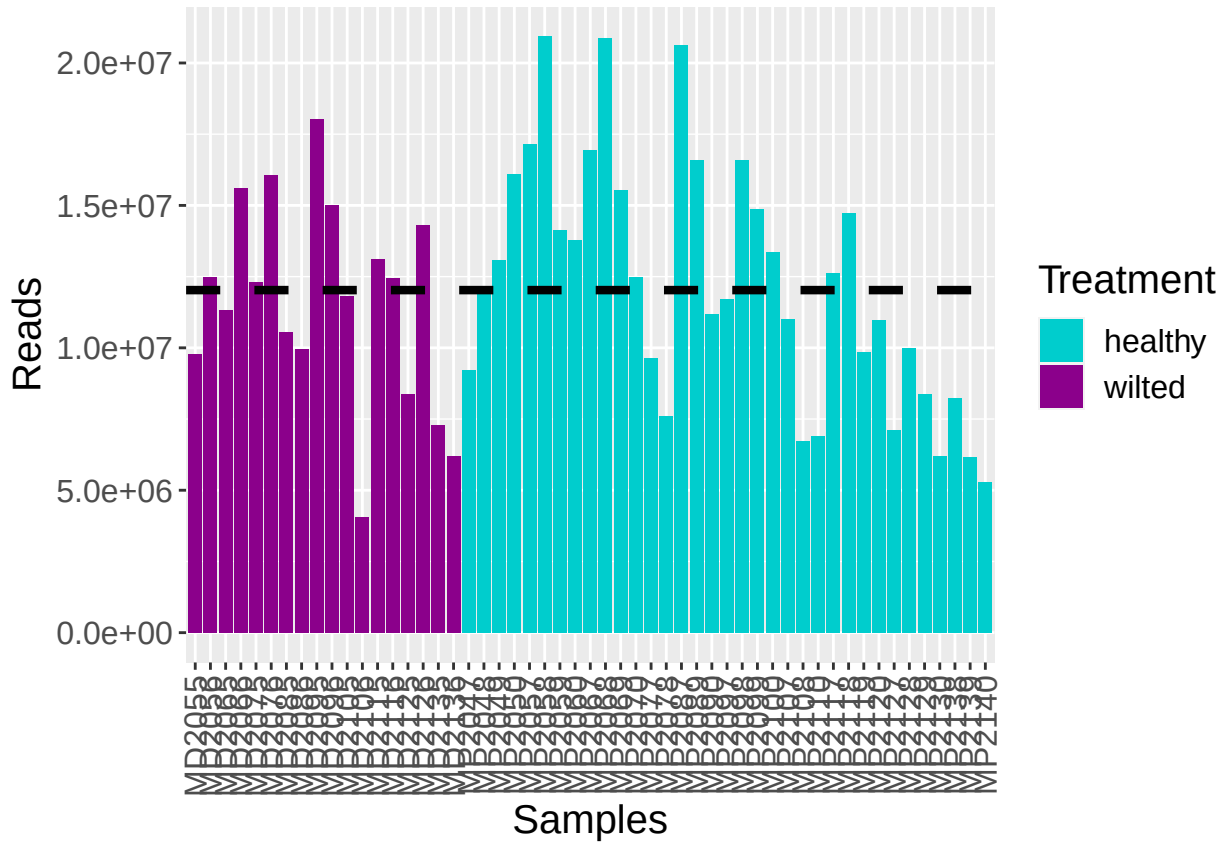
method=“TMM”es la media recortada ponderada de los valores M (a la referencia) propuesta por Robinson y Oshlack (2010), donde los pesos son del método delta en datos binomiales. Si refColumnno se especifica, se utiliza la biblioteca cuyo cuartil superior está más cerca del cuartil superior medio.

El método “TMM” es una media recortada ponderada de los valores M, que compara los niveles de expresión génica entre dos grupos (un grupo de referencia y un grupo de prueba). El método fue propuesto por Robinson y Oshlack en 2010 y utiliza pesos delta para datos binomiales. Si no se especifica la “refColumn” (la columna de referencia), se utiliza la biblioteca cuyo cuartil superior está más cerca del cuartil superior medio. En términos más simples, el método TMM es una técnica estadística utilizada para comparar los niveles de expresión génica y utiliza pesos y cálculos específicos para hacerlo. La “refColumn” se refiere a la columna de datos que sirve como referencia para la comparación

```
nor_dprof <- data.frame(Samples = colnames(nor_fresa_kraken_fil@otu_table@.Data),
                        Reads = sample_sums(nor_fresa_kraken_fil),
                        Treatment = nor_fresa_kraken_fil@sam_data@.Data[[1]])

mu_norSamples <- mean(nor_dprof$Reads)

ggplot(data = nor_dprof, mapping = aes(x = Samples, y = Reads))+
  geom_bar(stat = "identity", aes( fill = Treatment)) +
  geom_hline(yintercept=mu_norSamples, color = "black", linetype="dashed", size=1.5) +
  scale_fill_manual(values = c("cyan3","darkmagenta")) +
  theme(text = element_text(size = 15),
        axis.text.x = element_text(angle = 90, hjust = 1, vjust = 0.5))
```


2.0e+07
1.5e+07
1.0e+07
5.0e+06
0.0e+00
Reads
Samples
Treatment
healthy
wilted