Proyecto 2 DADA

Code

Hide

```
library(dada2)
```

```
Loading required package: Rcpp
```

Secuenciacion de la muestra de vid S85

Hide

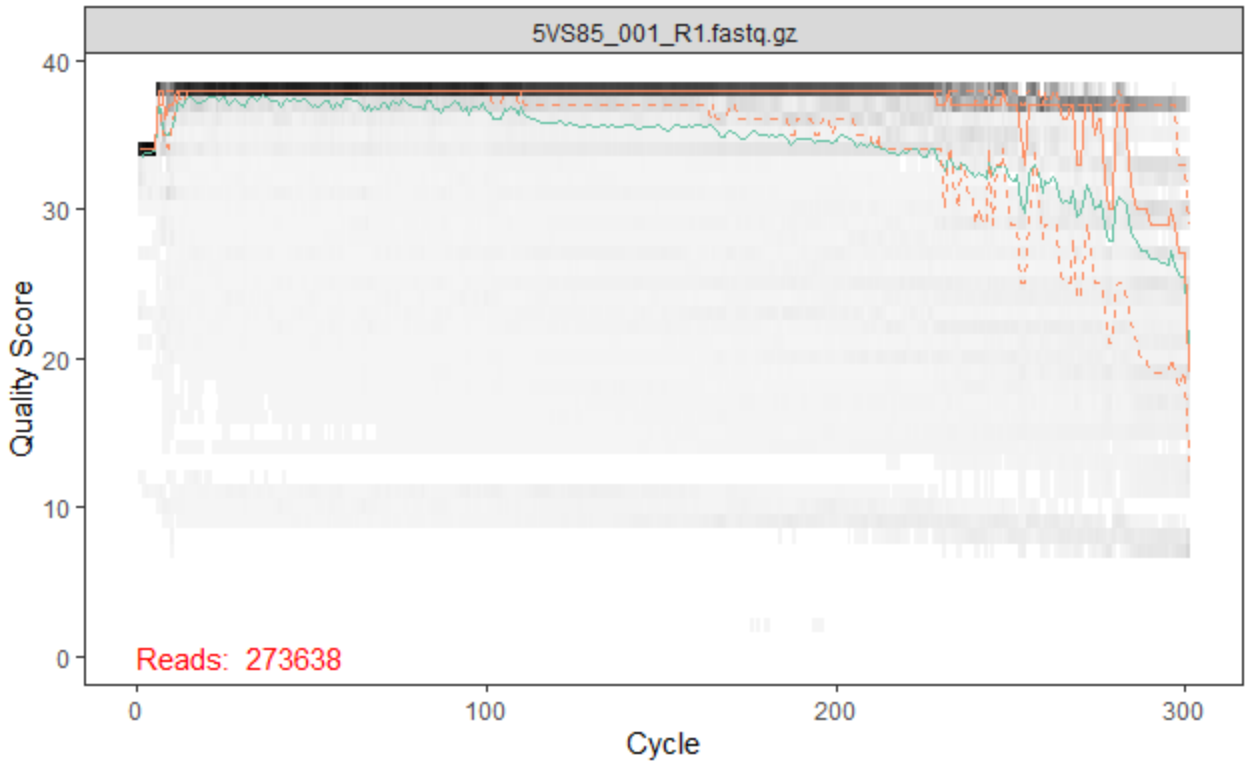
```
# Determinar el camino al directorio donde estan las muestras:

path <- "~/CursoR/CursoRgit/Secuenciacion_proyecto"

# Ahora se separan las muestras en objetos entre forward y reverse reads:

# Forward
fnFs <- sort(list.files(path,pattern="_R1.fastq.gz", full.names = TRUE))

# Reverse
fnRs <- sort(list.files(path,pattern="_R2.fastq.gz", full.names = TRUE))

sample.names <- sapply(strsplit(basename(fnFs), "_"), `[`,1)
```

Revision de los perfiles de calidad

Hide

```
plotQualityProfile(fnFs[1])
```

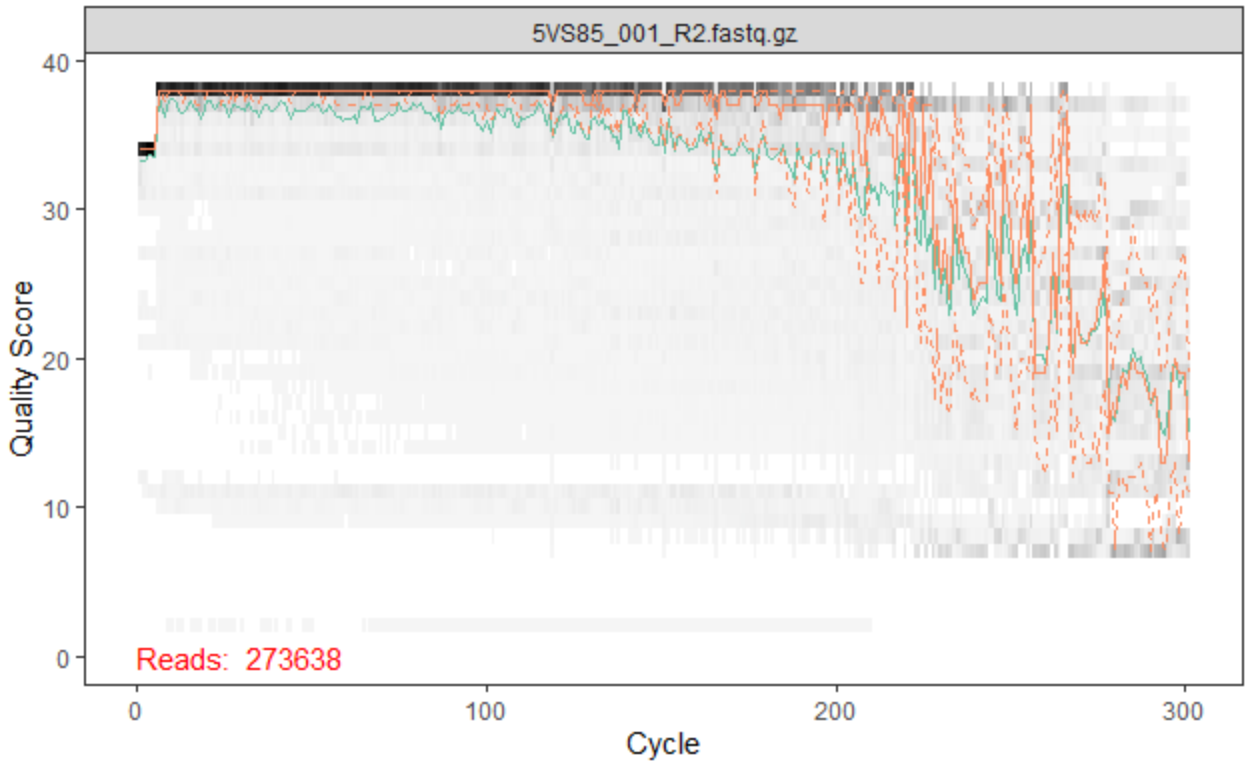
```
Warning: 'package:stats' puede no estar disponible en el proceso de carga
```



Hide

```
# reverse
plotQualityProfile(fnRs[1])
```

```
Warning: 'package:stats' puede no estar disponible en el proceso de carga
```



Filtrar y cortar

Primero crearemos una nueva carpeta para nuestras secuencias filtradas, asi como un nombre para los archivos .fastq que obtengamos.

Hide

```
# Guardando el camino a nuestras muestras filtradas en un objeto nuevo

filtFs <- file.path(path, "filtered", paste0(sample.names, "_F_filt.fastq.gz"))

filtRs <- file.path(path, "filtered", paste0(sample.names, "_R_filt.fastq.gz"))

# Asignando los nombres de las muestras a nuestros nuevos objetos
names(filtFs) <- sample.names
names(filtRs) <- sample.names
```

Explicacion de los parametros de filtrado y corte

Se revisaron los perfiles de calidad y se decidio cortar para la secuencia forward en la base 240 y para el reverse en 210, con el objetivo de ser lo mas estricto posible en cuanto al perfil de calidad, ya que lo ideal es que sea arriba del 40%. Sin embargo, se perdían muchas muestras por lo que se decidio que para la secuencia forward se cortara en la base 270 y para el reverse en 210. Para ambas secuencias se eligio un puntaje de calidad justo en el 30%.

En cuanto al error esperado maxEE se opto por elegir (5,5), con el objetivo de ser menos estrictos en cuanto a los errores esperados y asi evitar perder muchas muestras.

Hide

```
out <- filterAndTrim(fnFs, filtFs, # forward reads
                     fnRs, filtRs, # reverse reads
                     truncLen=c(270,210), # truncado o corte
                     maxN=0, # remover Ns NUNCA SE MODIFICA
                     maxEE=c(5,5), # error esperado
                     truncQ=2, # quality score
                     rm.phix=TRUE, compress=TRUE, # defaults
                     multithread=FALSE) # En windows multithread=FALSE
```

Hide

```
# Se guarda el progreso:

write.csv(out, "~/CursoR/CursoRgit/Materiales/Conteo_reads5_proyecto.csv")

#### Por si queremos retomar despues de filtrar ####

## Nuevo Camino
path2 <- "~/CursoR/CursoRgit/Secuenciacion_proyecto/filtered/"

# Forward
filtFs <- sort(list.files(path2,pattern="_F_filt.fastq.gz", full.names = TRUE))

# Reverse
filtRs <- sort(list.files(path2,pattern="_R_filt.fastq.gz", full.names = TRUE))
```

Tasas de error

Hide

```
# Forward
errF <- learnErrors(filtFs, multithread=TRUE)
save(errF,file = "errFproyecto2.RData")

# Reverse
errR <- learnErrors(filtRs, multithread=TRUE)
save(errR,file = "errRproyecto2.RData")

# Para volver a subir los archivos nuevamente se utiliza:
load("errFproyecto2.RData")
load("errRproyecto2.RData")

# Plot error rates
plotErrors(errF, nominalQ = TRUE)
plotErrors(errR, nominalQ = TRUE)
```

Inferencia de las muestras

Hide

```
# Forward
dadaFs_nopool <- dada(filtFs, err=errF, multithread=TRUE,
                      pool = FALSE)
save(dadaFs_nopool, file = "dadaFs_nopool_proyecto2.RData")

load("dadaFs_nopool_proyecto2.RData")

# Reverse
dadaRs_nopool <- dada(filtRs, err=errR, multithread=TRUE,
                      pool = FALSE)
save(dadaRs_nopool, file = "dadaRs_nopool_proyecto2.RData")

load("dadaRs_nopool_proyecto2.RData")
```

Uniendo las lecturas forward y reverse

Hide

```
mergers <- mergePairs(dadaFs_nopool, filtFs, dadaRs_nopool, filtRs, verbose = TRUE)
```

```
94419 paired-reads (in 18387 unique pairings) successfully merged out of 184113 (in 79911 pairin
gs) input.
```

Hide

```
save(mergers, file = "mergers_proyecto2.RData")
```

Hacer tablas de secuencias

Hide

```
seqtab <- makeSequenceTable(mergers)
dim(seqtab) # numero de muestras x numero de ASVs
```

```
[1]     1 18387
```

Hide

```
# Checar la longitud de todas las secuencias
table(nchar(getSequences(seqtab)))
```

```
 270  274  281  295  296  297  298  327  345  356  371  375  379  381  397
   1    1    1    5    2    2    5    4    1    1    1    1    1    2    1
 401  407  411  412  413  414  415  418  421  422  423  425  432  433  434
   1    1    1    1    1    1    1    4    1    1    2    1    2    3   10
 435  436  437  438  439  440  441  442  443  444  445  446  447  448  449
   1    2    1   45  686 6064  602  548  180  231 2226   85   40   33   32
 450  451  452  453  454  455  456  457  458  459  460  461  462  463  464
   2   19    6   56  310   15   35  128  201  271  334  125  230  446 1970
 465  466  467
2724  653   26
```

Quitar quimeras

Hide

```
# Se identificaron 5249 bimeras de 7392 secuencias.

# Basados en esto el 71% de mis secuencias son quimeras

## Comparar esta tabla con la original que incluye quimeras
dim(seqtab.nochim)
```

```
[1]    1 5249
```

Hide

```
sum(seqtab.nochim)/sum(seqtab) # se mantuvieron un 53% de secuencias no quimericas
```

```
[1] 0.530264
```

Seguimiento del proceso

Hide

```
out <- read.csv("~/CursoR/CursoRgit/Materiales/Conteo_reads5_proyecto.csv")

# Primero crearemos una funcion
getN <- function(x) sum(getUniques(x))

# Creamos una nueva tabla llamada track
track <- cbind(out, # Paso 1: filtrado y corte
               getN(dadaFs_nopool),
               getN(dadaRs_nopool), # Paso 3: denoising
               getN(mergers), # Paso 4: unir muestras
               rowSums(seqtab.nochim)) # Paso 5: quitar quimeras

# Nombramos nuestras filas y columnas
colnames(track) <- c("Sample_names","input", "filtered", "denoisedF", "denoisedR", "merged", "no
nchim")

#Guardamos esta tabla
write.csv(track, "~/CursoR/CursoRgit/Materiales/Abundancias_proyecto_Mariana.csv")
```

Asignar taxonomia

Hide

```
# Se crea la tabla de taxonomia
taxa <- assignTaxonomy(seqtab.nochim, "~/CursoR/CursoRgit/Secuenciacion/SILVA/silva_nr99_v138.1_
train_set.fa.gz", multithread = TRUE)

# Se añaden especies a la tabla
taxa <- addSpecies(taxa, "~/CursoR/CursoRgit/Secuenciacion/SILVA/silva_species_assignment_v138.
1.fa.gz")

save(taxa, file = "taxa_proyecto.RData")

write.csv(taxa, "~/CursoR/CursoRgit/Materiales/taxa_proyecto_Mariana.csv")
```