

Mothur Illumina Tutorial

Daniel Vaultot

17 janvier 2018

Contents

1	Aim of tutorial	1
2	Prerequisites	1
3	Data used	2
4	Directory structure	2
5	Pre visualization of the fastq files with R	3
6	Analysis with mothur	6
7	What is next ?	10
8	Alternative strategies	10

1 Aim of tutorial

This tutorial explain how to process Illumina sequences.

- The first part of the tutorial makes use of R to obtain information on the number and quality of sequences.
- The second part uses mothur to process the sequences and compute the final abundance table.

2 Prerequisites

Install the following software :

- mothur : <https://github.com/mothur/mothur/releases/tag/v1.39.5>
- PR2 database : https://github.com/vaultot/pr2_database/releases/download/4.7.2/pr2_version_4.7.2_mothur.zip . This file must be decompressed in the /databases directory
- R : <https://pbil.univ-lyon1.fr/CRAN/>
- R studio : <https://www.rstudio.com/products/rstudio/download/#download>
- Download and install the following libraries by running under R studio the following lines

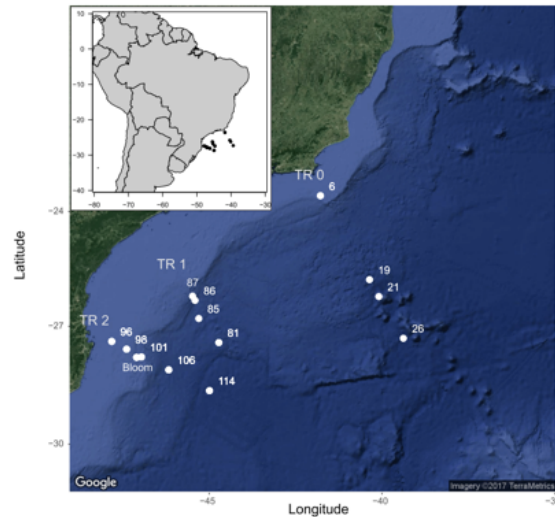
```
install.packages("dplyr")      # To manipulate dataframes
install.packages("stringr")    # To strings

install.packages("ggplot2")    # for high quality graphics

source("https://bioconductor.org/biocLite.R")
```

```
biocLite("Biostrings")      # manipulate sequences
biocLite('dada2')           # metabarcode data analysis
```

3 Data used



The samples originate from the CARBOM cruise (2013) off Brazil.

Samples have been sorted by flow cytometry and 3 genes have been PCR amplified :

- 18S rRNA - V4 region
- 16S rRNA with plastid
- nifH

The PCR products have been sequenced by 1 run of Illumina 2*250 bp. The data consist of the picoplankton samples from one transect and fastq files have been subsampled with 1000 sequences per sample.

3.1 References

- G rikas Ribeiro C, Marie D, Lopes dos Santos A, Pereira Brandini F, Vaultot D. (2016). Estimating microbial populations by flow cytometry: Comparison between instruments. *Limnol Oceanogr Methods* 14:750–758.
- G rikas Ribeiro C, Lopes dos Santos A, Marie D, Brandini P, Vaultot D. (2018). Relationships between photosynthetic eukaryotes and nitrogen-fixing cyanobacteria off Brazil. *ISME J* in press.
- G rikas Ribeiro C, Lopes dos Santos A, Marie D, Helena Pellizari V, Pereira Brandini F, Vaultot D. (2016). Pico and nanoplankton abundance and carbon stocks along the Brazilian Bight. *PeerJ* 4:e2587.

4 Directory structure

- /fastq_carbom : fastq files from the carbom cruise
- / databases : Silva alignment and PR2 database files (see Prerequisite above)
- /mothur/illumina : Tutorial for Illumina files (carbom cruise)
- /mothur/454 : Tutorial with 454 files

5 Pre visualization of the fastq files with R

Load the necessary libraries

```
library("dada2")
library("Biostrings") # To manipulate DNA sequences

library("ggplot2")
library("stringr")
library("dplyr")
```

(1) Function to plot quality of Illumina files (dada2 library)

This function takes as input the path where the fastq files are stored. The plots are saved as pdf files and the subdirectory /qual/

```
fastq_qual_plot <- function(fastq_path) {

  # List all the files in the directory fastq_path that have the extension .fastq
  fns <- sort(list.files(fastq_path, full.names = TRUE))
  fns <- fns[str_detect( basename(fns), ".fastq")]

  for(i in 1:length(fns)) {
    p1 <- plotQualityProfile(fns[i]) + ggtitle(basename(fns[i]))
    p1_file <- paste0(basename(fns[i]), ".pdf")
    ggsave( plot=p1, filename= paste0(fastq_path, "/qual/", p1_file),
            device = "pdf", width = 15, height = 15, scale=1, units="cm")
  }
}
```

(2) Function to compute number of sequences in fastq files

This function takes as input the path where the fastq files are stored. It saves a file with the number of sequences in each fastq file.

```
fastq_size <- function(fastq_path) {

  df <- data.frame()

  # List all the files in the directory fastq_path
  fns <- sort(list.files(fastq_path, full.names = TRUE))

  # Only consider files containing the R1 extension (the R2 file should have the same number of sequences)
  fns <- fns[str_detect( basename(fns), "R1")]

  for(i in 1:length(fns)) {
    geom <- fastq.geometry(fns[i])
    df_one_row <- data.frame( n_seq=geom[1], file_name=basename(fns[i]) )
    df <- bind_rows(df, df_one_row)
  }
  df
}
```

(3) Set up the directories for the analysis

```
# Change the following line to the path where you unzipped the tutorials
tutorial_dir <- "C:/Users/vaulot/Google Drive/Scripts/"
```

```

working_dir <- paste0( tutorial_dir, "metabarcodes_tutorials/mothur/illumina")
ngs_dir <- paste0( tutorial_dir, "metabarcodes_tutorials/fastq_carbom")

setwd(working_dir)

```

(4) Compute number of paired reads in each fastq file

Note that the data have been sub-sampled at 1000 reads per file.

```

# Call function that returns fastq file size
df <- fastq_size(ngs_dir)

```

```

# Only keep the R1 files since R1 and R2 files have same length
df <- df %>% filter(str_detect(file_name, "R1") == TRUE)
df

```

```

##      n_seq      file_name
## 1    1000 120p_S39_R1.subsample.fastq
## 2    1000 121p_S57_R1.subsample.fastq
## 3    1000 122p_S4_R1.subsample.fastq
## 4    1000 125p_S22_R1.subsample.fastq
## 5    1000 126p_S40_R1.subsample.fastq
## 6    1000 140p_S5_R1.subsample.fastq
## 7    1000 141p_S23_R1.subsample.fastq
## 8    1000 142p_S41_R1.subsample.fastq
## 9    1000 155p_S59_R1.subsample.fastq
## 10   1000 156p_S6_R1.subsample.fastq
## 11   1000 157p_S24_R1.subsample.fastq
## 12   1000 165p_S42_R1.subsample.fastq
## 13   1000 166p_S60_R1.subsample.fastq
## 14   1000 167p_S7_R1.subsample.fastq

```

```

# The following line writes a file with the data

```

```

# write.table(df, file = paste0(working_dir, "/n_seq.txt"), sep="\t", row.names = FALSE, na="", quote=F)

```

```

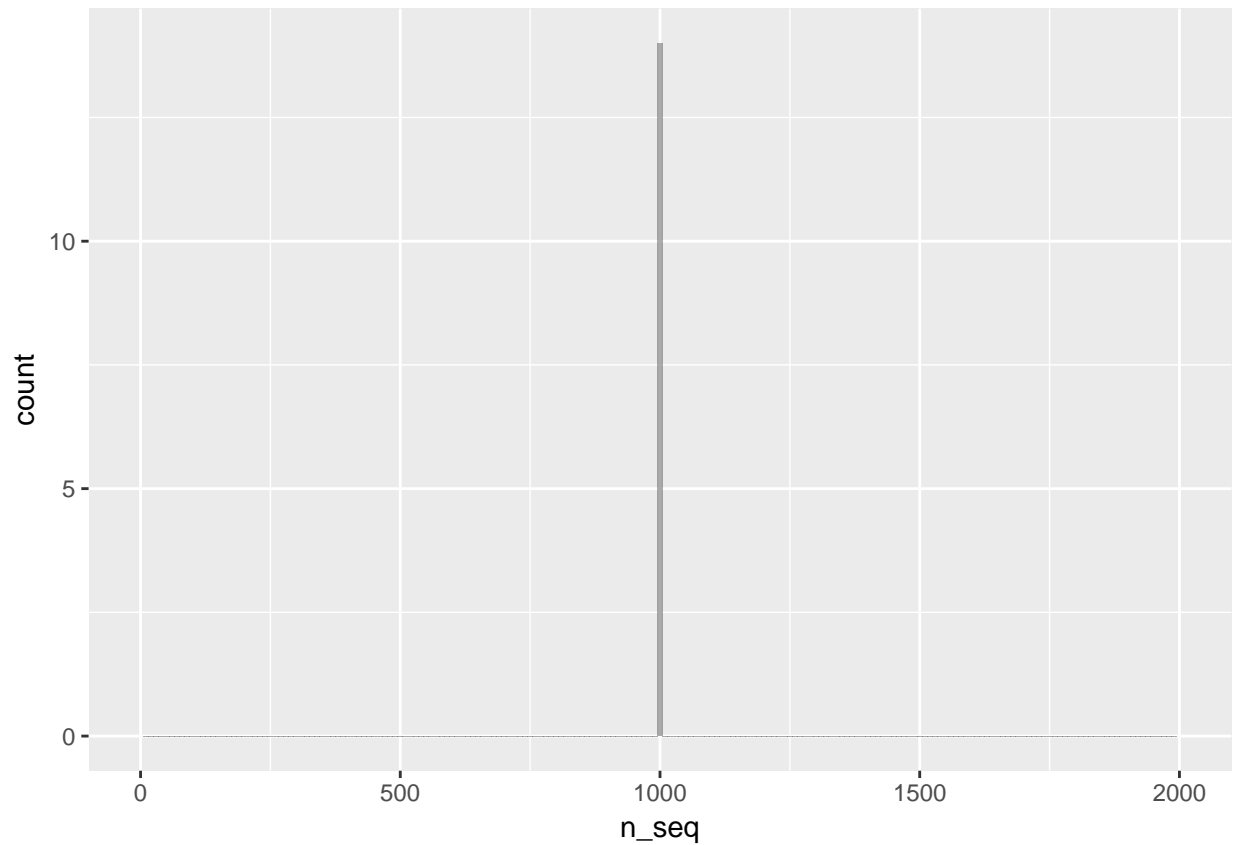
# Plot the histogram of the number of sequences per file

```

```

ggplot(df, aes(x=n_seq)) +
  geom_histogram( alpha = 0.5, position="identity", binwidth = 10) +
  xlim(0, 2000)

```



(5) **Plot the quality fo each fastq file**

```
# The script execute the following line  
# fastq_qual_plot(ngs_dir)  
  
# Here we just show one example of plot  
fastq_file <- paste0(ngs_dir, "/120p_S39_R1.subsample.fastq")  
plotQualityProfile(fastq_file)
```



(6) Clean up memory

It is necessary to clean up the memory because the fastq files are quite big and occupy a lot of memory during processing

```
rm(list=ls())
```

6 Analysis with mothur

Two files containing all the commands are provided

* mothur_carbom_linux.txt : use on a server (not tested on Mac) * mothur_carbom_windows.txt : use on windows

Note that some of the steps have been removed for simplicity.

The major steps of the processing are :

- Build the contigs from the R1 and R2 reads
- Extract the sequences that contain the 2 primers
- Remove sequences in low abundance (singletons in particular)
- Align sequences to a reference alignment
- Remove chimeras
- Assign taxonomy based on PR2
- Compute sequence distance
- Cluster sequences at a given threshold (make OTUs)
- Create a final file with all the information

- (1) First define a few constants to make the script independant of the files

```
# Change the DIR_DATA below to the path where you have downloaded the different files
DIR_DATA="... /metabarcodes_tutorials/fastq_carbom"

FILE_PR2_TAX="../databases/pr2_version_4.72_mothur.tax"
FILE_PR2_FASTA="../databases/pr2_version_4.72_mothur.fasta"
FILE_SILVA="../databases/silva.seed_v123.euk.fasta"
MOTHUR="mothur"
PROJECT="carbom"
```

- (2) Change directory to where the fastq files are located

```
cd $DIR_DATA
```

- (3) Make the contigs using the file \$PROJECT.txt (= carbom.txt).

This file has the following structure :

Sample	R1 file	R2 file
120p	120p_S39_R1.subsample.fastq	120p_S39_R2.subsample.fastq
121p	121p_S57_R1.subsample.fastq	121p_S57_R2.subsample.fastq
122p	122p_S4_R1.subsample.fastq	122p_S4_R2.subsample.fastq

```
$MOTHUR "#make.contigs(file=$PROJECT.txt, processors=32)"
```

- (4) Remove sequences that do not satisfy the following conditions:

- Number of ambiguities = 0
- Minlength=350
- Maxlength=450

```
$MOTHUR "#screen.seqs(fasta=$PROJECT.trim.contigs.fasta,group=$PROJECT.contigs.groups,
                      maxambig=0,minlength=350, maxlength=450, processors=32)"
```

- (5) Extract the sequences based on the presence of forward and reverse primers

- Mismatches allowed on the forward primer - pdiffs=2,
- Mismatches allowed on the reverse primer - rdiffs=2
- Oligo file : oligos18s_V4_Zingone.oligos

Keyword	Primer forward	Primer reverse	Name of primer
primer	CCAGCASCYGC GGTAATTCC	ACTTTCGTTCTTGATYRATGA	18S_V4_Zingone

```
$MOTHUR "#pcr.seqs(fasta=$PROJECT.trim.contigs.good.fasta,
                  group=$PROJECT.contigs.good.groups,
                  oligos=oligos18s_V4_Zingone.oligos,
                  pdiffs=2, rdiffs=2,
                  processors=32)"
```

- (6) Shorten file names and indicate gene name

```
cp $PROJECT.trim.contigs.good.pcr.fasta $PROJECT_18S.fasta
cp $PROJECT.contigs.good.pcr.groups $PROJECT_18S.groups
```

(7) Dereplicate unique sequences

```
$MOTHUR "#unique.seqs(fasta=$PROJECT_18S.fasta)"
```

(8) Create a count file

This file create a table which as the following structure. For each unique sequence, it provides the total number of sequences and the number of sequences in each sample.

Representative_Sequence	total	120p	121p	122p	125p	126p
M02439_22_000000000-AD0LA_1_1101_14247_1437	277	46	35	0	12	20
M02439_22_000000000-AD0LA_1_1101_12787_1647	2	2	0	0	0	0
M02439_22_000000000-AD0LA_1_1101_17899_1772	2	2	0	0	0	0
M02439_22_000000000-AD0LA_1_1101_13893_1778	1	1	0	0	0	0

This step saves disk space and speed up analysis

```
$MOTHUR "#count.seqs(name=$PROJECT_18S.names,  
                      group=$PROJECT_18S.groups, processors=32)"
```

(9) Remove singletons

One can change the settings with the cutoff parameter.

```
$MOTHUR "#split.abund(count=$PROJECT_18S.count_table,  
                      fasta=$PROJECT_18S.unique.fasta,  
                      cutoff=1, accnos=true)"
```

(10) Align sequences to reference alignment

The file to be used can be downloaded from the mothur web site : https://www.mothur.org/w/images/a/a4/Silva.seed_v128.tgz. It is best to :

- extract only the eukaryotes using mothur command: `get.lineage(taxonomy=$SILVA.tax, taxon=Eukaryota, fasta=$SILVA.align)`
- remove all the gaps that are common to all sequences with mothur command `filter.seqs` (see next line)

```
$MOTHUR "#align.seqs(fasta=$PROJECT_18S.unique.abund.fasta,  
                    reference=$FILE_SILVA,  
                    flip=T, processors=32)"
```

(11) Remove all the gaps that are common to all sequences

```
$MOTHUR "#filter.seqs(fasta=$PROJECT_18S.unique.abund.align, processors=32)"
```

(12) Precluster the sequences

The number of differences taken into account can be changed. In general use `diffs=2`. However if one does not want to make OTUS for example to look at fine genetic variation, it is necessary to remove this step.

```
$MOTHUR "#pre.cluster(fasta=$PROJECT_18S.unique.abund.filter.fasta,  
                     count=$PROJECT_18S.abund.count_table,  
                     diffs=1, processors=32)"
```

(13) Remove chimeras

```
$MOTHUR "#chimera.uchime(fasta=$PROJECT_18S.unique.abund.filter.precluster.fasta,  
                        count=$PROJECT_18S.unique.abund.filter.precluster.count_table,  
                        processors=32)"
```



```
$MOTHUR "#remove.seqs(fasta=$PROJECT_18S.unique.abund.filter.precluster.fasta,
                      accnos=$PROJECT_18S.unique.abund.filter.precluster.denovo.uchime.accnos,
                      count=$PROJECT_18S.unique.abund.filter.precluster.count_table)"
```

(14) **Remove sequences in low abundance (here cutoff=2)**

It is critical to remove the sequences in low abundance to speed up processing. In general use `cutoff = 10`.

```
$MOTHUR "#split.abund(count=$PROJECT_18S.unique.abund.filter.precluster.pick.count_table,
                      fasta=$PROJECT_18S.unique.abund.filter.precluster.pick.fasta,
                      cutoff=2, accnos=true)"
```

(15) **Remove sequences that are too short or too long (here minlength=200)**

```
$MOTHUR "#screen.seqs(fasta=$PROJECT_18S.unique.abund.filter.precluster.pick.abund.fasta,
                      count=$PROJECT_18S.unique.abund.filter.precluster.pick.abund.count_table,
                      minlength=200, processors=32)"
```

(16) **Rename files to remember that sequences in low abundance where removed**

```
cp $PROJECT_18S.unique.abund.filter.precluster.pick.abund.good.fasta
   $PROJECT_18S.uniq.preclust.no_chim.more_than_2.fasta
cp $PROJECT_18S.unique.abund.filter.precluster.pick.abund.good.count_table
   $PROJECT_18S.uniq.preclust.no_chim.more_than_2.count_table
```

(17) **Classify the sequences using the PR2 database**

Two files are required

- pr2.fasta
- pr2.taxo

```
$MOTHUR "#classify.seqs(fasta=$PROJECT_18S.uniq.preclust.no_chim.more_than_2.fasta,
                        count=$PROJECT_18S.uniq.preclust.no_chim.more_than_2.count_table,
                        reference=$FILE_PR2.fasta, taxonomy=$FILE_PR2.tax,
                        processors=32,
                        probs=T)"
```

(18) **Compute distance matrix**

It is critical to have as few sequences as possible at this step because the computation time is proportionnal to the **square** of the number of sequences.

```
$MOTHUR "#dist.seqs(fasta=$PROJECT_18S.uniq.preclust.no_chim.more_than_2.fasta, processors=32)"
```

(19) **Cluster the sequences to create the OTUs**

Here we use a 0.02 cutoff corresponding to 98% similarity.

```
$MOTHUR "#cluster(column=$PROJECT_18S.uniq.preclust.no_chim.more_than_2.dist,
                  count=$PROJECT_18S.uniq.preclust.no_chim.more_than_2.count_table,
                  cutoff=0.02, processors=32)"
```

(20) **Classify the OTUs based on the classification of the sequences (see above)**

```
$MOTHUR "#classify.otu(taxonomy=$PROJECT_18S.uniq.preclust.no_chim.more_than_2.$FILE_PR2_END.wang.taxonomy,
                       count=$PROJECT_18S.uniq.preclust.no_chim.more_than_2.count_table,
                       list=$PROJECT_18S.uniq.preclust.no_chim.more_than_2.opti_mcc.list,
                       label=0.02, probs=F, basis=sequence)"
```

(21) Get sequences represnetative of each OTU

```
$MOTHUR "#get.oturep(fasta=$PROJECT_18S.uniq.preclust.no_chim.more_than_2.fasta,
                    column=$PROJECT_18S.uniq.preclust.no_chim.more_than_2.dist,
                    count=$PROJECT_18S.uniq.preclust.no_chim.more_than_2.count_table,
                    list=$PROJECT_18S.uniq.preclust.no_chim.more_than_2.opti_mcc.list,
                    method=abundance,
                    cutoff=0.02) "
```

(22) **Format the final result in a single synthetic file**

- otu id
- abundance in each sample
- representative sequence
- taxonomy

```
$MOTHUR "#create.database(list=$PROJECT_18S.uniq.preclust.no_chim.more_than_2.opti_mcc.list,
count=$PROJECT_18S.uniq.preclust.no_chim.more_than_2.opti_mcc.0.02.rep.count_table,
label=0.02,
repfasta=$PROJECT_18S.uniq.preclust.no_chim.more_than_2.opti_mcc.0.02.rep.fasta ,
constaxonomy=$PROJECT_18S.uniq.preclust.no_chim.more_than_2.opti_mcc.0.02.cons.taxonomy)"
```

[illegible]

7 What is next ?

The database format can be easily used by the phyloseq package. A short tutorial can be found here : https://github.com/vaulot/R_tutorials

8 Alternative strategies

- Use the R dada2 package : <https://f1000research.com/articles/5-1492/v2>
- Use vsearch : <https://github.com/torognes/vsearch/wiki/VSEARCH-pipeline>