16S analysis pipeline
QC and ASV picking using the dada2 pipeline
Module 5: 16S rRNA Analysis Pipeline

• **Session 1:**
  QC and ASV picking using the dada2 pipeline

• **Session 2:**
  Taxonomic classification and alignment using the dada2 pipeline
Learning Objectives

• To give an overview on quality control process

• To give a background on dada2

• To give a general understanding of the 16S rRNA analysis pipeline using dada2

• To complete the dada2 workflow
Learning Outcomes

• To understand each step of 16S rRNA analysis pipeline using dada2
• To run 16S rRNA analysis pipeline from raw reads
• To know how to edit files and use command line
• To choose the best approach and tools to analyze your data
16S analysis pipeline
QC and ASV picking using the dada2 pipeline
Outline

• Quality Control

• DADA2 background

• DADA2 workflow
Quality Control

• Before analyzing generated sequence to draw biological conclusions, a quality control check should be performed to make sure there is no biases in the data.

• QC gives a quick impression of whether your data has any problems of which you should be aware before doing any analysis.
Quality Control

Potential problems:

• Low confidence bases (Ns)
• Sequence specific bias
• Sequence contamination
• Adapters
• …
Quality Control

Software packages for QC:

• FastQC
• MultiQC
• FastX-Toolkit
• PRINSEQ
• TagCleaner
• NGS QC Tool-Kit
• …
Quality Control

FASTQ format

What is a FastQ file?

FASTQ = FASTA + Quality

FastQ format is a text-based format for storing both a biological sequence and its corresponding quality scores.
Quality Control

FASTQ format

• Each FastQ file contains hundreds of millions of rows.
• Each block of 4 lines, starting with " @" represents a read.

**Line 1** begins with a '@' character and is followed by a sequence identifier and an optional description (like a FASTA title line).

**Line 2** is the raw sequence letters (ATCG).

**Line 3** begins with a '+' character and is optionally followed by the same sequence identifier (and any description).

**Line 4** encodes the quality values for the sequence in Line 2, and must contain the same number of symbols as letters in the sequence.
A FastQ file containing a single sequence might look like this:

```
@read_ID
GATTTGGGGTTCAAAGCAGTATCGATCAAATAGTAAATCCATT
TGTTCAACTCACAGTTT
+
! */ ' ' * ( ( ( ( * * * + ) ) % % % + + ) ( % % % % ) . 1 * * * - 
*''))**55CCFtitititititiCCCCCCC65
```

The character '!' represents the lowest quality while '~' is the highest.
Quality Control

Quality measurements

Base-calling error probabilities are reported by sequencers. Usually in Phred (quality) score. Usually coded by ASCII characters

**Phred score**

$$Q = -10\log_{10}P$$

If the quality of a base is 20, the probability that it is wrong is 0.01

<table>
<thead>
<tr>
<th>T</th>
<th>C</th>
<th>A</th>
<th>G</th>
<th>T</th>
<th>A</th>
<th>C</th>
<th>T</th>
<th>C</th>
<th>G</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>37</td>
<td>35</td>
</tr>
</tbody>
</table>
Quality Control

Quality measurements

<table>
<thead>
<tr>
<th>Character</th>
<th>Sanger</th>
<th>Solexa</th>
<th>Illumina 1.3+</th>
<th>Illumina 1.5+</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phred score</td>
<td>33, 64</td>
<td>64, 73</td>
<td>64, 73</td>
<td>64, 73</td>
</tr>
<tr>
<td>Raw Reads Typically</td>
<td>(0, 40)</td>
<td>(0, 40)</td>
<td>(0, 40)</td>
<td>(0, 40)</td>
</tr>
</tbody>
</table>

(Note: Imeasurements are for interpreting sequencing quality and not for assessment of the coding region of DNA.)
Quality Control

What is FastQC?

FastQC aims to provide a QC report which can spot problems which originate either in the sequencer or in the library material.
Quality Control

FastQC reports

- Normal
- Slightly abnormal
- Unexpected
# Quality Control

## FastQC reports

<table>
<thead>
<tr>
<th>Measure</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filename</td>
<td>good_sequence_short.txt</td>
</tr>
<tr>
<td>File type</td>
<td>Conventional base calls</td>
</tr>
<tr>
<td>Encoding</td>
<td>Illumina 1.5</td>
</tr>
<tr>
<td>Total Sequences</td>
<td>250000</td>
</tr>
<tr>
<td>Sequences flagged as poor quality</td>
<td>0</td>
</tr>
<tr>
<td>Sequence length</td>
<td>40</td>
</tr>
<tr>
<td>%GC</td>
<td>45</td>
</tr>
</tbody>
</table>
Quality Control

FastQC reports
Per Base Sequence Quality

Quality scores across all bases (Sanger / Illumina 1.9 encoding)
Quality Control

FastQC reports

Per Base Sequence Quality

Good quality FastQC report:

Bad quality FastQC report
Quality Control

FastQC reports

Per Sequence Quality Scores

Good quality FastQC report: 

Bad quality FastQC report
Quality Control

FastQC reports

Per Base Sequence Content

DNA library

RNA library
16S analysis pipeline
QC and ASV picking using the dada2 pipeline
Outline

• Quality Control

• DADA2 background

• DADA2 workflow
Divisive Amplicon Denoising Algorithm 2 (DADA2) is an open source algorithm implemented in R, which uses a statistical inference to correct amplicon errors.

It intends to simplify the study of microbial communities by allowing to reconstruct amplicon-sequenced communities at the highest resolution.

Callahan et al., 2016
DADA2 background

DADA2 implements a complete workflow that takes raw amplicon sequencing data in fastq files as input.

It produces an error-corrected table of the abundances of amplicon sequence variants in each sample (ASV table).

Callahan et al., 2016
DADA2 background

Benefits to DADA2:

• Compatible with all amplicon types
  16S, 18S, ITS,…

• Works on different next generation sequencing platforms
  Illumina, Ion Torrent, 454 pyrosequencing

• Provides single-nucleotide resolution

• Lower false-positive rate
DADA2 background

- R or RStudio.
- QIIME2
  - Simplified and condensed the dada2 workflow
DADA2 background

Benjamin Callahan

16SrRNA Intermediate Bioinformatics Online Course:
Int_BT_2019
Imane Allali
DADA2 background

Sample sequences → amplicon reads → OTUs

Errors → DADA2 → Make OTUs

Benjamin Callahan
16S rRNA Intermediate Bioinformatics Online Course: Int_BT_2019
Imane Allali
16S analysis pipeline
QC and ASV picking using the dada2 pipeline
Outline

• Quality Control

• DADA2 background

• DADA2 workflow
DADA2 workflow

1. Raw FASTQ files
2. Sort F and R reads
3. Quality inspection
4. Trim and filter reads
5. Chimera checking and removal
6. Merge reads
7. Sample inference
8. Error rate estimation
9. Assignment taxonomy & phylogeny
10. Diversity analysis and visualization
DADA2 workflow

- Raw FASTQ files
- Sort F and R reads
- Quality inspection
- Trim and filter reads
- Chimera checking and removal
- Merge reads
- Sample inference
- Error rate estimation
- Assignment taxonomy & phylogeny
- Diversity analysis and visualization

16S rRNA Intermediate Bioinformatics Online Course:
Int_BT_2019
Imane Allali
DADA2 workflow

Before running the pipeline:

• Barcodes, adapters should be removed
  – Cutadapt, Trimmomatic, …

• Samples should be demultiplexed
  – FASTX-Toolkit, idemp, …

• For paired-end data, forward and reverse reads must be in the same order.
DADA2 workflow

The data:

• The data can be accessed [here](#).
• Stool samples.
• Paired-end 300 bp reads.
• Barcodes/Adapters have been removed.
DADA2 workflow

Getting Ready

Load the dada2 package in your R/RStudio

```r
library(dada2); packageVersion("dada2")
```

If you do not already have it, see the [dada2 installation instructions](#)
Getting Ready

Set the path, it points to the **dog samples** directory:

```r
MY_HOME <- Sys.getenv("HOME")
data <- paste(MY_HOME, "/dada2_tutorial_dog/dog_samples", sep='') # change the path
list.files(data)
```

```r
## [1] "Dog1_R1.fastq"  "Dog1_R2.fastq"  "Dog10_R1.fastq"  "Dog10_R2.fastq"
## [5] "Dog15_R1.fastq"  "Dog15_R2.fastq"  "Dog16_R1.fastq"  "Dog16_R2.fastq"
## [9] "Dog17_R1.fastq"  "Dog17_R2.fastq"  "Dog2_R1.fastq"  "Dog2_R2.fastq"
## [13] "Dog22_R1.fastq"  "Dog22_R2.fastq"  "Dog23_R1.fastq"  "Dog23_R2.fastq"
## [17] "Dog24_R1.fastq"  "Dog24_R2.fastq"  "Dog29_R1.fastq"  "Dog29_R2.fastq"
## [21] "Dog3_R1.fastq"   "Dog3_R2.fastq"   "Dog30_R1.fastq"  "Dog30_R2.fastq"
## [25] "Dog31_R1.fastq"  "Dog31_R2.fastq"  "Dog8_R1.fastq"   "Dog8_R2.fastq"
## [29] "Dog9_R1.fastq"   "Dog9_R2.fastq"
```
DADA2 workflow

Getting Ready

Sort the forward and reverse reads

```r
# Forward and reverse fastq filenames have format: SAMPLENAME_R1.fastq and SAMPLENAME_R2.fastq
dataF <- sort(list.files(data, pattern="_R1.fastq", full.names = TRUE))
dataR <- sort(list.files(data, pattern="_R2.fastq", full.names = TRUE))
```

Extract sample names

```r
# Extract sample names, assuming filenames have format: SAMPLENAME_XXX.fastq
list.sample.names <- sapply(strsplit(basename(dataF), "_"), \[\[`, 1)
list.sample.names
```

```r
## [1] "Dog1" "Dog10" "Dog15" "Dog16" "Dog17" "Dog2" "Dog22" "Dog23" "Dog24" "Dog29" "Dog3" "Dog30" "Dog31" "Dog8" "Dog9"
```
DADA2 workflow

Quality Control

- The quality plot of three forward samples.

- Scores never really go below 30.
DADA2 workflow

Quality Control

- The reverse reads are slightly different.
- The scores are good but they drop off right around 275 bp.
DADA2 workflow

Filter and Trim

Set filtered subdirectory and rename files

```r
# Place filtered files in filtered/ subdirectory
filt.dataF <- file.path(data, "filtered", paste0(list.sample.names, "_F_filt.fastq.gz"))
filt.dataR <- file.path(data, "filtered", paste0(list.sample.names, "_R_filt.fastq.gz"))

names(filt.dataF) <- list.sample.names
names(filt.dataR) <- list.sample.names
```
DADA2 workflow

Filter and Trim

```r
out <- filterAndTrim(dataF, filt.dataF, dataR, filt.dataR, truncLen=c(290,275),
                     maxN=0, maxEE=c(2,2), truncQ=2, rm.phix=TRUE,
                     compress=TRUE, multithread=TRUE) # On Windows set multithread=FALSE
```

**truncLen** truncates your reads at specific base.

```
truncLen=c(290,275)
```

The amplicon length.
The length of your overlap, by default is 20 for DADA2.
DADA2 workflow

Filter and Trim

\[
\text{out} \leftarrow \text{filterAndTrim}(\text{dataF, filt.dataF, dataR, filt.dataR, truncLen=c(290,275), maxN=0, maxEE=c(2,2), truncQ=2, rm.phix=TRUE, compress=TRUE, multithread=TRUE})
\]

\[
\text{maxN} \quad \text{maximum number of ambiguous nucleotides.}
\]

\[
\text{maxN}=0
\]

DADA2 requires no Ns.
DADA2 workflow

Filter and Trim

```r
out <- filterAndTrim(dataF, filt.dataF, dataR, filt.dataR, truncLen=c(290,275),
                      maxN=0, maxEE=c(2,2), truncQ=2, rm.phix=TRUE,
                      compress=TRUE, multithread=TRUE) # On Windows set multithread=FALSE
```

**maxEE** maximum number of estimated errors allowed in your reads.

`maxEE=c(2,2)`

The quality of your sequences.
DADA2 workflow

Filter and Trim

```r
out <- filterAndTrim(dataF, filt.dataF, dataR, filt.dataR, truncLen=c(290,275),
                      maxN=0, maxEE=c(2,2), truncQ=2, rm.phix=TRUE,
                      compress=TRUE, multithread=TRUE)  # On Windows set multithread=FALSE
```

**truncQ** truncates the read at the first nucleotide with a specific quality score.

\[
\text{truncQ}=2
\]

Score of 2 means that the probability of the base being incorrect is 63%.
DADA2 workflow

Filter and Trim

```r
out <- filterAndTrim(dataF, filt.dataF, dataR, filt.dataR, truncLen=c(290,275), maxN=0, maxEE=c(2,2), truncQ=2, rm.phix=TRUE, compress=TRUE, multithread=TRUE) # On Windows set multithread=FALSE
```

`rm.phix` removes reads that match against the phiX genome.

`rm.phix=TRUE`
DADA2 workflow

Filter and Trim

```r
out <- filterAndTrim(dataF, filt.dataF, dataR, filt.dataR, truncLen=c(290,275),
                      maxN=0, maxEE=c(2,2), truncQ=2, rm.phix=TRUE,
                      compress=TRUE, multithread=TRUE)  # On Windows set multithread=FALSE
```

**Compress** if you want to fastq files to be gzipped.

**Multithread** if you want your files to run in parallel.
DADA2 workflow

Learn the Error Rates

It will create an error model that will be used by the DADA2 algorithm.

\[
\text{errF} \leftarrow \text{learnErrors}(\text{filt.dataF, multithread=} \text{TRUE})
\]

\[
\text{errR} \leftarrow \text{learnErrors}(\text{filt.dataR, multithread=} \text{TRUE})
\]
Learn the Error Rates

- The error rates for each possible transition (A -> C).
- As quality score increases, the expected error rate decreases.
DADA2 workflow

Sample Inference

Set filtered subdirectory and rename files

```r
# Place filtered files in filtered/ subdirectory
filt.dataF <- file.path(data, "filtered", paste0(list.sample.names, "_F_filt.fastq.gz"))
filt.dataR <- file.path(data, "filtered", paste0(list.sample.names, "_R_filt.fastq.gz"))
names(filt.dataF) <- list.sample.names
names(filt.dataR) <- list.sample.names
```

It uses the error model that was created earlier.

p-value high -> sequence likely caused by errors.
p-value low -> sequence is real.
16SrRNA Intermediate Bioinformatics Online Course

16S analysis pipeline
QC and ASV picking using the dada2 pipeline
Practical
Practical

The practical is available here:

https://iallali.github.io/DADA2_pipeline/16SrRNA_DADA2_pipeline.html
Practical

Getting Ready

First, we load the `dada2` package on your RStudio. If you do not already have it, see the `dada2` installation instructions.

```r
library(dada2); packageVersion("dada2")
```

```r
##[1] '1.13.1'
```

We set the path so that it points to the extracted directory of the dataset named "dog_samples" on your computer or cluster:

```r
MY_HOME <- Sys.getenv("HOME")
data <- paste(MY_HOME, '/dada2_tutorial_dog/dog_samples', sep='')  # change the path
list.files(data)
```

```r
##[1] 'Dog1_R1.fastq' 'Dog1_R2.fastq' 'Dog10_R1.fastq' 'Dog10_R2.fastq'
##[5] 'Dog15_R1.fastq' 'Dog15_R2.fastq' 'Dog16_R1.fastq' 'Dog16_R2.fastq'
##[9] 'Dog17_R1.fastq' 'Dog17_R2.fastq' 'Dog2_R1.fastq' 'Dog2_R2.fastq'
##[13] 'Dog22_R1.fastq' 'Dog22_R2.fastq' 'Dog23_R1.fastq' 'Dog23_R2.fastq'
##[17] 'Dog24_R1.fastq' 'Dog24_R2.fastq' 'Dog29_R1.fastq' 'Dog29_R2.fastq'
##[21] 'Dog3_R1.fastq' 'Dog3_R2.fastq' 'Dog30_R1.fastq' 'Dog30_R2.fastq'
##[25] 'Dog31_R1.fastq' 'Dog31_R2.fastq' 'Dog9_R1.fastq' 'Dog9_R2.fastq'
##[29] 'Dog9_R1.fastq' 'Dog9_R2.fastq' 'filtered'
```

If your listed files match those here, you can start running the DADA2 pipeline.

Now, we read in the names of the fastq files and we sort them by forward and reverse. Then, we perform some string manipulation to extract a list of the sample names.

```r
# Forward and reverse fastq filenames have format: SAMPLENAME_R1.fastq and SAMPLENAME_R2.fastq
dataF <- sort(list.files(data, patterns="_R1.fastq", full.names = TRUE))
dataR <- sort(list.files(data, patterns="_R2.fastq", full.names = TRUE))

# Extract sample names, assuming filenames have format: SAMPLENAME_RXXX.fastq
list.sample.names <- apply(strsplit(basename(dataF), ""), 1, list)
```

```r
##[1] "Dog1" "Dog10" "Dog15" "Dog16" "Dog17" "Dog2" "Dog22" "Dog23"
##[5] "Dog24" "Dog29" "Dog3" "Dog30" "Dog31" "Dog8" "Dog9"
```
16S rRNA analysis pipeline
Taxonomic classification and alignment using the dada2 pipeline
Module 5: 16S rRNA Analysis Pipeline

• **Session 1:**
  QC and ASV picking using the dada2 pipeline

• **Session 2:**
  Taxonomic classification and alignment using the dada2 pipeline
Outline

• Quality Control

• DADA2 background

• DADA2 workflow
DADA2 workflow

1. Raw FASTQ files
2. Sort F and R reads
3. Quality inspection
4. Trim and filter reads
5. Sample inference
6. Error rate estimation
7. Merge reads
8. Construct Sequence Table
9. Chimera checking and removal
10. Assign taxonomy & phylogeny
11. Diversity analysis and visualization
DADA2 workflow

1. Raw FASTQ files
2. Sort F and R reads
3. Quality inspection
4. Trim and filter reads
5. Construct Sequence Table
6. Merge reads
7. Sample inference
8. Error rate estimation
9. Chimera checking and removal
10. Assign taxonomy & phylogeny
11. Diversity analysis and visualization
DADA2 workflow

Merge Reads

```r
merge.reads <- mergePairs(dadaF, filt.dataF, dadaR, filt.dataR, verbose=TRUE)
```

mergePairs merges reads only if they exactly overlap.

The length of your overlap, by default is 20 nt for DADA2, you can lower it by using this parameter `minOverlap`.
DADA2 workflow

Merge Reads

```
head(merge.reads[[1]])
```

<table>
<thead>
<tr>
<th></th>
<th>sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>TGTGTCACAGCGCAGCGGCTATACCGGAGGTACGCTAAATTGTCAGCTAGCTGAGTGACCCAGCTTTTTTAAGTCAGATGTTAAATCCCGGGCC</td>
</tr>
<tr>
<td>2</td>
<td>TGTGTCAGCGCAGCGGCTATACCGGAGGTACGCTAAATTGTCAGCTAGCTGAGTGACCCAGCTTTTTTAAGTCAGATGTTAAATCCCGGGCC</td>
</tr>
<tr>
<td>3</td>
<td>TGTGTCACAGCGCAGCGGCTATACCGGAGGTACGCTAAATTGTCAGCTAGCTGAGTGACCCAGCTTTTTTAAGTCAGATGTTAAATCCCGGGCC</td>
</tr>
<tr>
<td>4</td>
<td>TGTGTCAGCGCAGCGGCTATACCGGAGGTACGCTAAATTGTCAGCTAGCTGAGTGACCCAGCTTTTTTAAGTCAGATGTTAAATCCCGGGCC</td>
</tr>
<tr>
<td>5</td>
<td>TGTGTCACAGCGCAGCGGCTATACCGGAGGTACGCTAAATTGTCAGCTAGCTGAGTGACCCAGCTTTTTTAAGTCAGATGTTAAATCCCGGGCC</td>
</tr>
<tr>
<td>6</td>
<td>TGTGTCACAGCGCAGCGGCTATACCGGAGGTACGCTAAATTGTCAGCTAGCTGAGTGACCCAGCTTTTTTAAGTCAGATGTTAAATCCCGGGCC</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>abundance forward reverse mismatch nmatch nindel prefer accept</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>460 2 1 253 0 0 2 TRUE</td>
</tr>
<tr>
<td>2</td>
<td>456 1 1 253 0 0 2 TRUE</td>
</tr>
<tr>
<td>3</td>
<td>421 5 1 253 0 0 2 TRUE</td>
</tr>
<tr>
<td>4</td>
<td>414 7 2 252 0 0 2 TRUE</td>
</tr>
<tr>
<td>5</td>
<td>401 6 1 253 0 0 2 TRUE</td>
</tr>
<tr>
<td>6</td>
<td>400 4 1 253 0 0 2 TRUE</td>
</tr>
</tbody>
</table>
DADA2 workflow

Construct Amplicon Sequence Variant (ASV) Table

```r
seqtab <- makeSequenceTable(merge.reads)
dim(seqtab)

## [1] 15 13527

table(nchar(getSequences(seqtab)))

##
## 311 312 313 315
## 107 9136 4283 1
```
# DADA2 workflow

## Construct Amplicon Sequence Variant (ASV) Table

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
<th>J</th>
<th>K</th>
<th>L</th>
<th>M</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Dog1</td>
<td>0</td>
<td>0</td>
<td>242</td>
<td>205</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>223</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>Dog10</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>Dog15</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>Dog16</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>Dog17</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>Dog2</td>
<td>373</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>276</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>283</td>
<td>277</td>
<td>0</td>
<td>322</td>
</tr>
<tr>
<td>8</td>
<td>Dog22</td>
<td>1926</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1516</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1453</td>
<td>1459</td>
<td>0</td>
<td>1366</td>
</tr>
<tr>
<td>9</td>
<td>Dog23</td>
<td>0</td>
<td>955</td>
<td>0</td>
<td>0</td>
<td>805</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>Dog24</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>11</td>
<td>Dog29</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>Dog3</td>
<td>0</td>
<td>921</td>
<td>0</td>
<td>0</td>
<td>944</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>Dog30</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>14</td>
<td>Dog31</td>
<td>0</td>
<td>0</td>
<td>1596</td>
<td>1625</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1523</td>
<td>0</td>
<td>0</td>
<td>1536</td>
</tr>
<tr>
<td>15</td>
<td>Dog8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>16</td>
<td>Dog9</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>17</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
DADA2 workflow

Chimera Sequence

• Chimeras are sequences formed from two or more biological sequences joined together.

• Amplicons with chimeric sequences can be formed during PCR.

• Chimeras are rare with shotgun sequencing but are common in amplicon sequencing when closely related sequences are amplified.
DADA2 workflow

Chimera Sequence

https://help.ezbiocloud.net/
DADA2 workflow

Chimera Checking and Removal

```r
seqtab.nochim <- removeBimeraDenovo(seqtab, method = "consensus", multithread = TRUE, verbose = TRUE)
```

- It uses *de novo* to check for two parent chimeras.

- Chimeric sequences are identified if they can be exactly reconstructed by combining a left-segment and a right-segment from two more abundant “parent” sequences.

```r
dim(seqtab.nochim)
#> [1] 15 4415

sum(seqtab.nochim)/sum(seqtab)
#> [1] 0.5094968
```
DADA2 workflow

Chimera Checking and Removal

Strain 1  Strain 2

They are not aligned

They are partially aligned
DADA2 workflow

Chimera Checking and Removal

Strain 1

They are partially aligned

They are partially aligned

Chimeric Sequence

Strain 2
Chimera Checking and Removal

It uses *de novo* to check for two parent chimeras.

Chimeric sequences are identified if they can be exactly reconstructed by combining a left-segment and a right-segment from two more abundant “parent” sequences.
16SrRNA Intermediate Bioinformatics Online Course

16S rRNA analysis pipeline
Taxonomic classification and alignment using the dada2 pipeline
DADA2 workflow

Track Reads through DADA2 pipeline

```r
getN <- function(x) sum(getUniques(x))
track.nbr.reads <- cbind(out, sapply(dadaF, getN), sapply(dadaR, getN), sapply(merge.reads, getN), rowSums(seqtab.nochimp))

colnames(track.nbr.reads) <- c("input", "filtered", "denoisedF", "denoisedR", "merged", "nonchim")
rownames(track.nbr.reads) <- list.sample.names
head(track.nbr.reads)
```

<table>
<thead>
<tr>
<th></th>
<th>input</th>
<th>filtered</th>
<th>denoisedF</th>
<th>denoisedR</th>
<th>merged</th>
<th>nonchim</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dog1</td>
<td>118343</td>
<td>84815</td>
<td>82888</td>
<td>83710</td>
<td>76306</td>
<td>35440</td>
</tr>
<tr>
<td>Dog10</td>
<td>79342</td>
<td>61952</td>
<td>60537</td>
<td>61130</td>
<td>54948</td>
<td>24103</td>
</tr>
<tr>
<td>Dog15</td>
<td>131483</td>
<td>91235</td>
<td>89781</td>
<td>90372</td>
<td>83013</td>
<td>48592</td>
</tr>
<tr>
<td>Dog16</td>
<td>114424</td>
<td>83564</td>
<td>81753</td>
<td>82442</td>
<td>74533</td>
<td>46006</td>
</tr>
<tr>
<td>Dog17</td>
<td>99610</td>
<td>72919</td>
<td>71427</td>
<td>72229</td>
<td>67364</td>
<td>42960</td>
</tr>
<tr>
<td>Dog2</td>
<td>108679</td>
<td>79668</td>
<td>77366</td>
<td>78301</td>
<td>69262</td>
<td>30129</td>
</tr>
</tbody>
</table>
DADA2 workflow

Assign Taxonomy

taxa <- assignTaxonomy(seqtab.nochim, paste(MY_HOME, "/dada2_tutorial_dog/RefSeq-RDP16S_v3_May2018.fa.gz", sep=' '), multithread=TRUE) # change the path

RefSeq-RDP16S_v3_May2018.fa.gz

Three most common 16S databases: Silva, RDP and GreenGenes
DADA2 workflow

Assign Taxonomy

Maintained:

- Silva version 132, Silva version 128, Silva version 123 (Silva dual-license)
- RDP trainset 16, RDP trainset 14
- GreenGenes version 13.8
- UNITE (use the General Fasta releases)

Contributed:

- RefSeq + RDP (NCBI RefSeq 16S rRNA database supplemented by RDP)
  - Reference files formatted for assignTaxonomy
  - Reference files formatted for assignSpecies
- GTDB: Genome Taxonomy Database (More info: http://gtdb.ecogenomic.org/)
  - Reference files formatted for assignTaxonomy
  - Reference files formatted for assignSpecies
- HitDB version 1 (Human InTestinal 16S rRNA)
- RDP fungi LSU trainset 11
- Silva Eukaryotic 18S, v132 & v128
- PR2 version 4.7.2+. SEE NOTE BELOW.
DADA2 workflow

Assign Taxonomy

taxa.print <- taxa # Removing sequence rownames for display only
rownames(taxa.print) <- NULL
head(taxa.print)

<table>
<thead>
<tr>
<th>#</th>
<th>Kingdom</th>
<th>Phylum</th>
<th>Class</th>
<th>Order</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Bacteria</td>
<td>Firmicutes</td>
<td>Clostridia</td>
<td>Clostridiales</td>
</tr>
<tr>
<td>2</td>
<td>Bacteria</td>
<td>Firmicutes</td>
<td>Clostridia</td>
<td>Clostridiales</td>
</tr>
<tr>
<td>3</td>
<td>Bacteria</td>
<td>Bacteroidetes</td>
<td>Bacteroidia</td>
<td>Bacteroidiales</td>
</tr>
<tr>
<td>4</td>
<td>Bacteria</td>
<td>Bacteroidetes</td>
<td>Bacteroidia</td>
<td>Bacteroidiales</td>
</tr>
<tr>
<td>5</td>
<td>Bacteria</td>
<td>Firmicutes</td>
<td>Clostridia</td>
<td>Clostridiales</td>
</tr>
<tr>
<td>6</td>
<td>Bacteria</td>
<td>Firmicutes</td>
<td>Clostridia</td>
<td>Clostridiales</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>#</th>
<th>Family</th>
<th>Genus</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Peptostreptococcaceae</td>
<td>Clostridium_XI</td>
</tr>
<tr>
<td>2</td>
<td>Peptostreptococcaceae</td>
<td>Clostridium_XI</td>
</tr>
<tr>
<td>3</td>
<td>Prevotellaceae</td>
<td>Alloprevotella</td>
</tr>
<tr>
<td>4</td>
<td>Prevotellaceae</td>
<td>Alloprevotella</td>
</tr>
<tr>
<td>5</td>
<td>Peptostreptococcaceae</td>
<td>Clostridium_XI</td>
</tr>
<tr>
<td>6</td>
<td>Peptostreptococcaceae</td>
<td>Clostridium_XI</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>#</th>
<th>Species</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Clostridium_hiranonis(AB023970)</td>
</tr>
<tr>
<td>2</td>
<td>Clostridium_hiranonis(AB023970)</td>
</tr>
<tr>
<td>3</td>
<td>Prevotellamassilia_timonensis(NR_144750.1)</td>
</tr>
<tr>
<td>4</td>
<td>Prevotellamassilia_timonensis(NR_144750.1)</td>
</tr>
<tr>
<td>5</td>
<td>Clostridium_hiranonis(AB023970)</td>
</tr>
<tr>
<td>6</td>
<td>Clostridium_hiranonis(AB023970)</td>
</tr>
</tbody>
</table>
DADA2 workflow

Assign Taxonomy

```python
write.csv(taxa, file="ASVs_taxonomy.csv")
saveRDS(taxa, "ASVs_taxonomy.rds")
```

<table>
<thead>
<tr>
<th>A</th>
<th>Kingdom</th>
<th>Phylum</th>
<th>Class</th>
<th>Order</th>
<th>Family</th>
<th>Genus</th>
<th>Species</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ASV_1</td>
<td>Bacteria</td>
<td>Firmicutes</td>
<td>Clostridia</td>
<td>Clostridales</td>
<td>Peptostreptococcaceae</td>
<td>Clostridium_hiranonis(AB023970)</td>
</tr>
<tr>
<td>2</td>
<td>ASV_2</td>
<td>Bacteria</td>
<td>Firmicutes</td>
<td>Clostridia</td>
<td>Clostridales</td>
<td>Peptostreptococcaceae</td>
<td>Clostridium_hiranonis(AB023970)</td>
</tr>
<tr>
<td>3</td>
<td>ASV_3</td>
<td>Bacteria</td>
<td>Bacteroidetes</td>
<td>Bacteroidia</td>
<td>Bacteroidales</td>
<td>Prevotellaceae</td>
<td>Alloprevotella</td>
</tr>
<tr>
<td>4</td>
<td>ASV_4</td>
<td>Bacteria</td>
<td>Bacteroidetes</td>
<td>Bacteroidia</td>
<td>Bacteroidales</td>
<td>Prevotellaceae</td>
<td>Alloprevotella</td>
</tr>
<tr>
<td>5</td>
<td>ASV_5</td>
<td>Bacteria</td>
<td>Firmicutes</td>
<td>Clostridia</td>
<td>Clostridales</td>
<td>Peptostreptococcaceae</td>
<td>Clostridium_hiranonis(AB023970)</td>
</tr>
<tr>
<td>6</td>
<td>ASV_6</td>
<td>Bacteria</td>
<td>Firmicutes</td>
<td>Clostridia</td>
<td>Clostridales</td>
<td>Peptostreptococcaceae</td>
<td>Clostridium_hiranonis(AB023970)</td>
</tr>
<tr>
<td>7</td>
<td>ASV_7</td>
<td>Bacteria</td>
<td>Bacteroidetes</td>
<td>Bacteroidia</td>
<td>Bacteroidales</td>
<td>Prevotellaceae</td>
<td>Alloprevotella</td>
</tr>
<tr>
<td>8</td>
<td>ASV_8</td>
<td>Bacteria</td>
<td>Bacteroidetes</td>
<td>Bacteroidia</td>
<td>Bacteroidales</td>
<td>Prevotellaceae</td>
<td>Alloprevotella</td>
</tr>
<tr>
<td>9</td>
<td>ASV_9</td>
<td>Bacteria</td>
<td>Firmicutes</td>
<td>Clostridia</td>
<td>Clostridales</td>
<td>Peptostreptococcaceae</td>
<td>Clostridium_hiranonis(AB023970)</td>
</tr>
<tr>
<td>10</td>
<td>ASV_10</td>
<td>Bacteria</td>
<td>Firmicutes</td>
<td>Clostridia</td>
<td>Clostridales</td>
<td>Peptostreptococcaceae</td>
<td>Clostridium_hiranonis(AB023970)</td>
</tr>
<tr>
<td>11</td>
<td>ASV_11</td>
<td>Bacteria</td>
<td>Bacteroidetes</td>
<td>Bacteroidia</td>
<td>Bacteroidales</td>
<td>Prevotellaceae</td>
<td>Alloprevotella</td>
</tr>
<tr>
<td>12</td>
<td>ASV_12</td>
<td>Bacteria</td>
<td>Bacteroidetes</td>
<td>Bacteroidia</td>
<td>Bacteroidales</td>
<td>Prevotellaceae</td>
<td>Alloprevotella</td>
</tr>
<tr>
<td>13</td>
<td>ASV_13</td>
<td>Bacteria</td>
<td>Firmicutes</td>
<td>Clostridia</td>
<td>Clostridales</td>
<td>Peptostreptococcaceae</td>
<td>Clostridium_hiranonis(AB023970)</td>
</tr>
<tr>
<td>14</td>
<td>ASV_14</td>
<td>Bacteria</td>
<td>Firmicutes</td>
<td>Clostridia</td>
<td>Clostridales</td>
<td>Peptostreptococcaceae</td>
<td>Clostridium_hiranonis(AB023970)</td>
</tr>
<tr>
<td>15</td>
<td>ASV_15</td>
<td>Bacteria</td>
<td>Bacteroidetes</td>
<td>Bacteroidia</td>
<td>Bacteroidales</td>
<td>Prevotellaceae</td>
<td>Alloprevotella</td>
</tr>
<tr>
<td>16</td>
<td>ASV_16</td>
<td>Bacteria</td>
<td>Bacteroidetes</td>
<td>Bacteroidia</td>
<td>Bacteroidales</td>
<td>Prevotellaceae</td>
<td>Alloprevotella</td>
</tr>
<tr>
<td>17</td>
<td>ASV_17</td>
<td>Bacteria</td>
<td>Firmicutes</td>
<td>Clostridia</td>
<td>Clostridales</td>
<td>Peptostreptococcaceae</td>
<td>Clostridium_hiranonis(AB023970)</td>
</tr>
<tr>
<td>18</td>
<td>ASV_18</td>
<td>Bacteria</td>
<td>Firmicutes</td>
<td>Clostridia</td>
<td>Clostridales</td>
<td>Peptostreptococcaceae</td>
<td>Clostridium_hiranonis(AB023970)</td>
</tr>
<tr>
<td>19</td>
<td>ASV_19</td>
<td>Bacteria</td>
<td>Bacteroidetes</td>
<td>Bacteroidia</td>
<td>Bacteroidales</td>
<td>Prevotellaceae</td>
<td>Alloprevotella</td>
</tr>
<tr>
<td>20</td>
<td>ASV_20</td>
<td>Bacteria</td>
<td>Bacteroidetes</td>
<td>Bacteroidia</td>
<td>Bacteroidales</td>
<td>Prevotellaceae</td>
<td>Alloprevotella</td>
</tr>
<tr>
<td>21</td>
<td>ASV_21</td>
<td>Bacteria</td>
<td>Bacteroidetes</td>
<td>Bacteroidia</td>
<td>Bacteroidales</td>
<td>Prevotellaceae</td>
<td>Alloprevotella</td>
</tr>
<tr>
<td>22</td>
<td>ASV_22</td>
<td>Bacteria</td>
<td>Firmicutes</td>
<td>Clostridia</td>
<td>Clostridales</td>
<td>Peptostreptococcaceae</td>
<td>Clostridium_hiranonis(AB023970)</td>
</tr>
<tr>
<td>23</td>
<td>ASV_23</td>
<td>Bacteria</td>
<td>Firmicutes</td>
<td>Clostridia</td>
<td>Clostridales</td>
<td>Peptostreptococcaceae</td>
<td>Clostridium_hiranonis(AB023970)</td>
</tr>
<tr>
<td>24</td>
<td>ASV_24</td>
<td>Bacteria</td>
<td>Bacteroidetes</td>
<td>Bacteroidia</td>
<td>Bacteroidales</td>
<td>Prevotellaceae</td>
<td>Alloprevotella</td>
</tr>
<tr>
<td>25</td>
<td>ASV_25</td>
<td>Bacteria</td>
<td>Bacteroidetes</td>
<td>Bacteroidia</td>
<td>Bacteroidales</td>
<td>Prevotellaceae</td>
<td>Alloprevotella</td>
</tr>
</tbody>
</table>

16S rRNA Intermediat Bioinformatics Online Course: Int_BT_2019
Imane Allali
DADA2 workflow

Assign Taxonomy

```r
asv_headers <- vector(dim(seqtab.no.chim)[2], mode="character")
count.asv.tab <- t(seqtab.no.chim)
row.names(count.asv.tab) <- sub(">", "", asv_headers)
write.csv(count.asv.tab, file="ASVs_counts.csv")
saveRDS(count.asv.tab, file="ASVs_counts.rds")
```
DADA2 workflow

Alignment

library(DECIPHER)

Tools for curating, analyzing, and manipulating biological sequences.

seqs <- getSequences(seqtab.nochim)
names(seqs) <- seqs # This propagates to the tip labels of the tree
alignment <- AlignSeqs(DNAStringSet(seqs), anchor=NA)
Construct Phylogenetic Tree

It constructs a neighbor-joining tree.
1. Change sequence alignment output into `phyDat` structure.
2. Create distance matrix using `dist.ml`.
3. Perform neighbor joining.
4. Perform internal maximum likelihood.
Construct Phylogenetic Tree

It fits a GTR+G+I (Generalized time-reversible with Gamma rate variation) maximum likelihood tree using the neighbor-joining tree as a starting point.

```r
fitGTR <- update(fit, k=4, inv=0.2)
fitGTR <- optim.pml(fitGTR, model="GTR", optInv=TRUE, optGamma=TRUE,
rearrangement = "stochastic", control = pml.control(trace = 0))

saveRDS(fitGTR, "phangorn.tree.RDS")
```
16SrRNA Intermediate Bioinformatics Online Course

16S analysis pipeline
Taxonomic classification and alignment using the dada2 pipeline
Practical
Practical

The practical is available here:

https://iallali.github.io/DADA2_pipeline/16SrRNA_DADA2_pipeline.html
5. Merge the Paired Reads

In this step, we merge the forward and reverse reads to obtain the full sequences.

```r
merge.reads <- mergePairs(dadaF, filt.dataF, dadaR, filt.dataR, verbose=TRUE)
```

```r
## 76306 paired-reads (in 1153 unique pairings) successfully merged out of 82378 (in 2526 pairings) input.
## 54948 paired-reads (in 720 unique pairings) successfully merged out of 60139 (in 1811 pairings) input.
## 83016 paired-reads (in 843 unique pairings) successfully merged out of 89310 (in 2175 pairings) input.
## 74534 paired-reads (in 1093 unique pairings) successfully merged out of 81135 (in 2674 pairings) input.
## 67264 paired-reads (in 855 unique pairings) successfully merged out of 71053 (in 1743 pairings) input.
## 69262 paired-reads (in 1194 unique pairings) successfully merged out of 76778 (in 2891 pairings) input.
## 97401 paired-reads (in 954 unique pairings) successfully merged out of 106466 (in 2847 pairings) input.
## 129237 paired-reads (in 1632 unique pairings) successfully merged out of 146610 (in 5263 pairings) input.
## 111558 paired-reads (in 1143 unique pairings) successfully merged out of 120597 (in 3110 pairings) input.
## 85319 paired-reads (in 839 unique pairings) successfully merged out of 92957 (in 2348 pairings) input.
## 68550 paired-reads (in 915 unique pairings) successfully merged out of 74010 (in 2198 pairings) input.
## 92870 paired-reads (in 1266 unique pairings) successfully merged out of 107043 (in 4252 pairings) input.
## 101935 paired-reads (in 981 unique pairings) successfully merged out of 108579 (in 2495 pairings) input.
```