https://github.com/AdamOss88

Running the pipeline

AMPLICON SEQUENCING DATA ANALYSIS

SEQUENCING DATA ANALYSIS

Adam Ossowicki

AdamOss88

Type / to search

Overview | Repositories 7 | Projects | Packages | Stars 1

Popular repositories

Customize your pins

fasta_primer_TOOLS — Public

Raw-to-Rawr-amplicon-workshop — Public

Materials used for a amplicon data analysis in University Leiden 2023

R

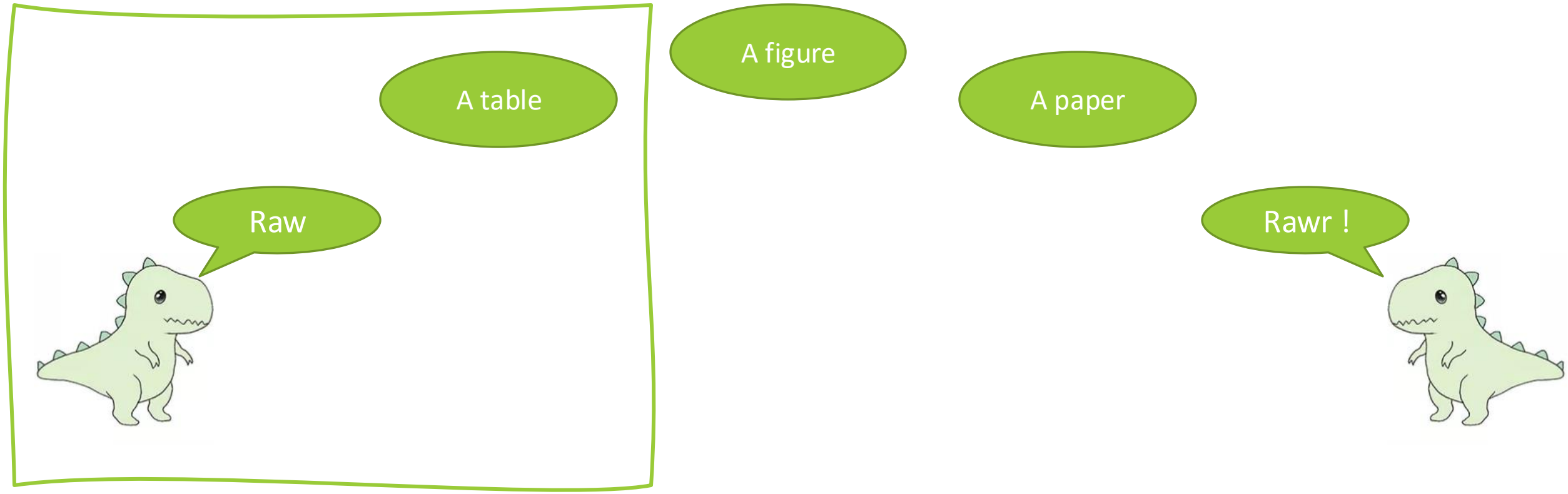
Raw-to-Rawr-server — Public

R

Raw-to-Rawr-SLURM — Public

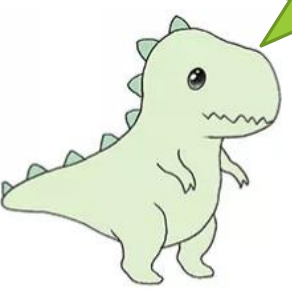
R

Stages of data analysis



Pre-processing

Stages of amplicon data analysis



Raw

Dereplication – check but probably already done, how to check

Quality check

Primer removal marcelm/**cutadapt**

Quality trimming/ filtering

OUT/ASV inferring DADA2 Amplicon Sequencing. Exactly.

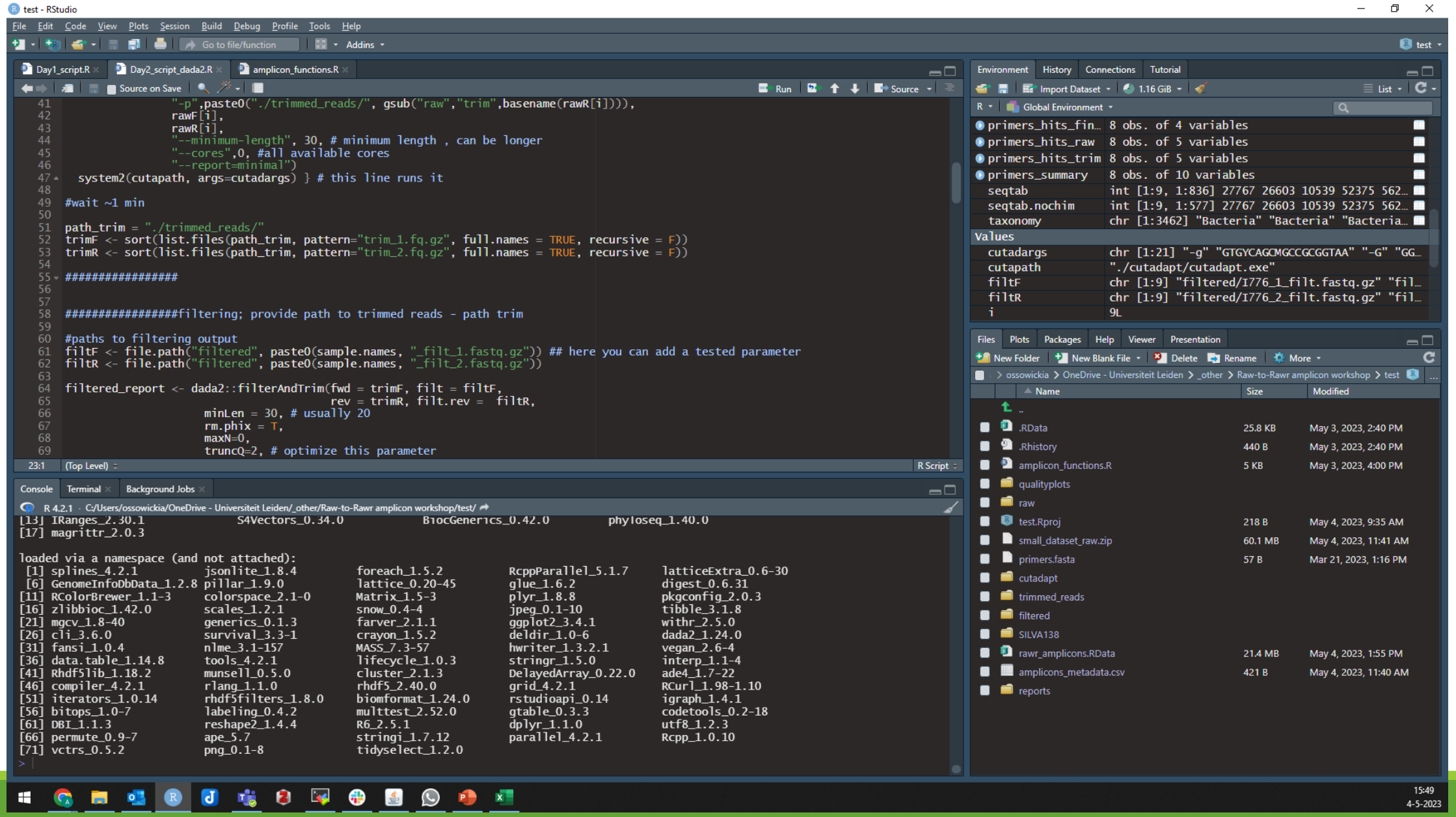
Taxonomy assignment

Normalization

A table

	ASV1	ASV2	ASV3	ASV4	ASV5	ASV6								
I706	2133	1779	489	1398	1346	60								
I710	7575	8203	5166	677	1832	6	581							
I711	9686	5699	3022	535	1351	13	400	4						
I714	46	41	32	0	8	3904	0	1877	1					
I716	8578	3594	1768	8923	2734	50	1714	0	562					
I718	8138	8770	6161	2465	1996	6	3059	0	1803	1706				
I719	6282	8921	5905	2358	2644	13	3804	0	1747	1697	17			
I720	7625	11278	5158	2153	1228	11	3435	0	1828	1871	5	4		
I721	42	44	23	0	6	2992	0	8580	12	23	1631	0		
I722	4737	5664	2230	947	1235	296	1972	0	979	832	9	366	13	
I723	60	36	39	0	6	2297	0	875	9	17	1183	0	137	101
I725	8422	9706	6980	1645	1681	8	1980	0	1795	1646	19	510	105	220
I726	8680	13953	3077	1316	2319	7	1666	0	1094	1004	12	698	173	290
I727	47	33	0	0	7	3645	0	2760	0	31	950	0	1474	118
I728	14993	1949	1416	6434	3032	8	890	0	480	400	11	588	162	85
I731	8488	2756	1986	1342	1886	29	578	0	776	693	5	652	3284	103
I732	3735	911	990	2737	1142	66	555	0	285	300	0	377	6320	110
I733	16844	1525	1316	9030	1258	6	693	0	549	402	0	438	1029	0

Rstudio


test - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
test
Day1_script.R
Day2_script_dada2.R
amplicon_functions.R
Source on Save
Run
Source
"-p",paste0("./trimmed_reads/", gsub("raw","trim",basename(rawR[i]))),
rawF[i],
rawR[i],
"--minimum-length", 30, # minimum length , can be longer
"--cores",0, #all available cores
"--report=minimal")
system2(cutapath, args=cutadargs) } # this line runs it
#wait ~1 min
path_trim = "./trimmed_reads/"
trimF <- sort(list.files(path_trim, pattern="trim_1.fq.gz", full.names = TRUE, recursive = F))
trimR <- sort(list.files(path_trim, pattern="trim_2.fq.gz", full.names = TRUE, recursive = F))
################
################filtering; provide path to trimmed reads - path trim
#paths to filtering output
filtF <- file.path("filtered", paste0(sample.names, "_filt_1.fastq.gz")) ## here you can add a tested parameter
filtR <- file.path("filtered", paste0(sample.names, "_filt_2.fastq.gz"))
filtered_report <- dada2::filterAndTrim(fwd = trimF, filt = filtF,
rev = trimR, filt.rev = filtR,
minLen = 30, # usually 20
rm.phix = T,
maxN=0,
truncQ=2, # optimize this parameter
23:1
(Top Level)
R Script
Console
Terminal
Background Jobs
R 4.2.1 · C:/Users/ossowickia/OneDrive - Universiteit Leiden/_other/Raw-to-Rawr amplicon workshop/test/
[13] IRanges_2.30.1 S4Vectors_0.34.0 BiocGenerics_0.42.0 phyloseq_1.40.0
[17] magrittr_2.0.3
loaded via a namespace (and not attached):
[1] splines_4.2.1 jsonlite_1.8.4 foreach_1.5.2 RcppParallel_5.1.7 latticeExtra_0.6-30
[6] GenomeInfoDbData_1.2.8 pillar_1.9.0 lattice_0.20-45 glue_1.6.2 digest_0.6.31
[11] RColorBrewer_1.1-3 colorspace_2.1-0 Matrix_1.5-3 plyr_1.8.8 pkgconfig_2.0.3
[16] zlibbioc_1.42.0 scales_1.2.1 snow_0.4-4 jpeg_0.1-10 tibble_3.1.8
[21] mgcv_1.8-40 generics_0.1.3 farver_2.1.1 ggplot2_3.4.1 withr_2.5.0
[26] cli_3.6.0 survival_3.3-1 crayon_1.5.2 deldir_1.0-6 dada2_1.24.0
[31] fansi_1.0.4 nlme_3.1-157 MASS_7.3-57 hwriter_1.3.2.1 vegan_2.6-4
[36] data.table_1.14.8 tools_4.2.1 lifecycle_1.0.3 stringr_1.5.0 interp_1.1-4
[41] Rhdf5lib_1.18.2 munsell_0.5.0 cluster_2.1.3 DelayedArray_0.22.0 ade4_1.7-22
[46] compiler_4.2.1 rlang_1.1.0 rhdf5_2.40.0 grid_4.2.1 RCurl_1.98-1.10
[51] iterators_1.0.14 rhdf5filters_1.8.0 biomformat_1.24.0 rstudioapi_0.14 igraph_1.4.1
[56] bitops_1.0-7 labeling_0.4.2 multtest_2.52.0 gtable_0.3.3 codetools_0.2-18
[61] DBI_1.1.3 reshape2_1.4.4 R6_2.5.1 dplyr_1.1.0 utf8_1.2.3
[66] permute_0.9-7 ape_5.7 stringi_1.7.12 parallel_4.2.1 Rcpp_1.0.10
[71] vctrs_0.5.2 png_0.1-8 tidyselect_1.2.0
Environment
History
Connections
Tutorial
Import Dataset
1.16 GiB
List
R
Global Environment
primers_hits_fin… 8 obs. of 4 variables
primers_hits_raw 8 obs. of 5 variables
primers_hits_trim 8 obs. of 5 variables
primers_summary 8 obs. of 10 variables
seqtab int [1:9, 1:836] 27767 26603 10539 52375 562…
seqtab.nochim int [1:9, 1:577] 27767 26603 10539 52375 562…
taxonomy chr [1:3462] "Bacteria" "Bacteria" "Bacteria…
Values
cutadargs chr [1:21] "-g" "GTGYCAGCMGCCGCGGTAA" "-G" "GG…
cutapath "./cutadapt/cutadapt.exe"
filtF chr [1:9] "filtered/I776_1_filt.fastq.gz" "fil…
filtR chr [1:9] "filtered/I776_2_filt.fastq.gz" "fil…
i 9L
Files
Plots
Packages
Help
Viewer
Presentation
New Folder
New Blank File
Delete
Rename
More
ossowickia › OneDrive - Universiteit Leiden › _other › Raw-to-Rawr amplicon workshop › test
Name
Size
Modified
.RData 25.8 KB May 3, 2023, 2:40 PM
.Rhistory 440 B May 3, 2023, 2:40 PM
amplicon_functions.R 5 KB May 3, 2023, 4:00 PM
qualityplots
raw
test.Rproj 218 B May 4, 2023, 9:35 AM
small_dataset_raw.zip 60.1 MB May 4, 2023, 11:41 AM
primers.fasta 57 B Mar 21, 2023, 1:16 PM
cutadapt
trimmed_reads
filtered
SILVA138
rawr_amplicons.RData 21.4 MB May 4, 2023, 1:55 PM
amplicons_metadata.csv 421 B May 4, 2023, 11:40 AM
reports
15:49
4-5-2023

Rmarkdown

```
---
title: "MicroWorkshop25"
author: "Adam Oss"
date: "2025-01-25"
output: html_document
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```

quality checks pre-processed
```{r Raw-to-Rawr#1}

## load libraries
library("dada2")

## link the raw data
path = "./raw/" #set path to raw data

rawF <- sort(list.files(path, pattern="_1.fq.gz", full.names = TRUE, recursive = F))
rawR <- sort(list.files(path, pattern="_2.fq.gz", full.names = TRUE, recursive = F))

## get sample names
sample.names <- sapply(strsplit(basename(rawF), "_raw_"), `[`, 1)

```
```

Packages for R

Using them :

#1 Load the whole package :

library(“mypackage”)

#Use a function

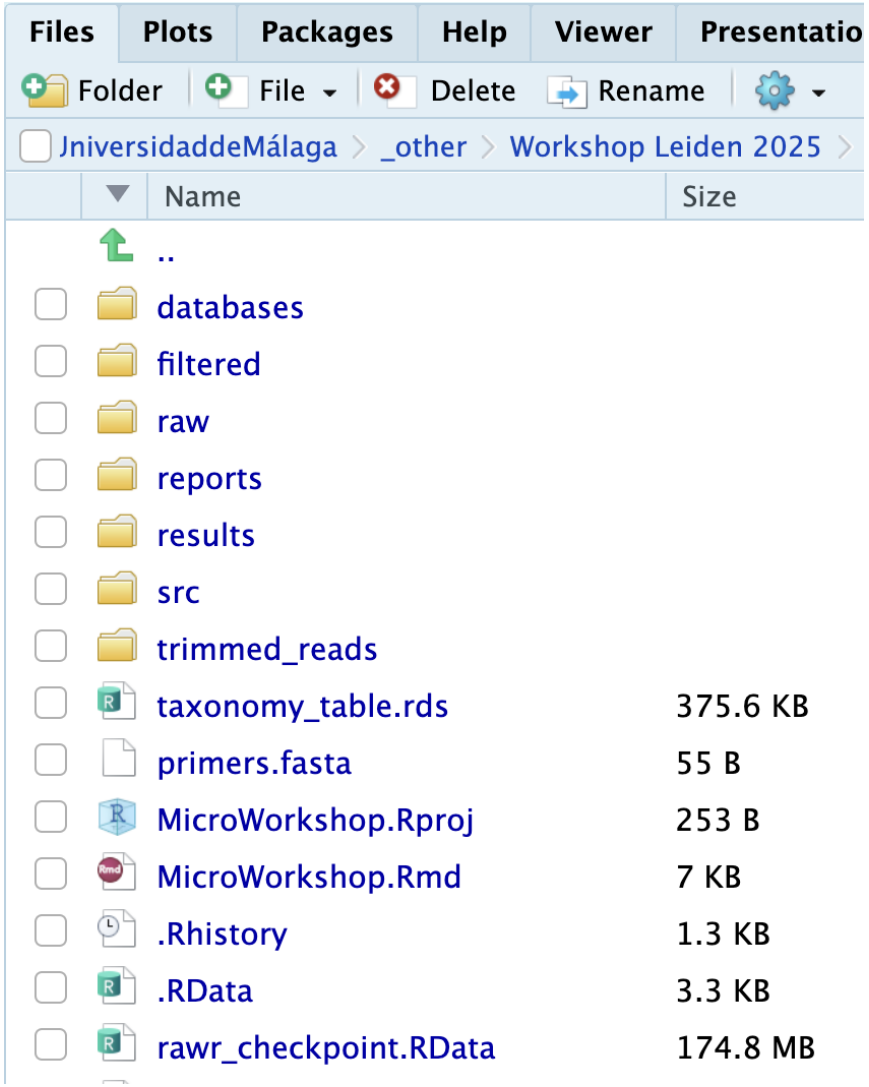
myfunction()

#2 load selected function from a package:

mypackage::myfunction()

doeasn’t always work

Before you start working on anything **create a new project**



Why is it important ?

Make meaningful names

~~Project1/allmyfiles.R~~

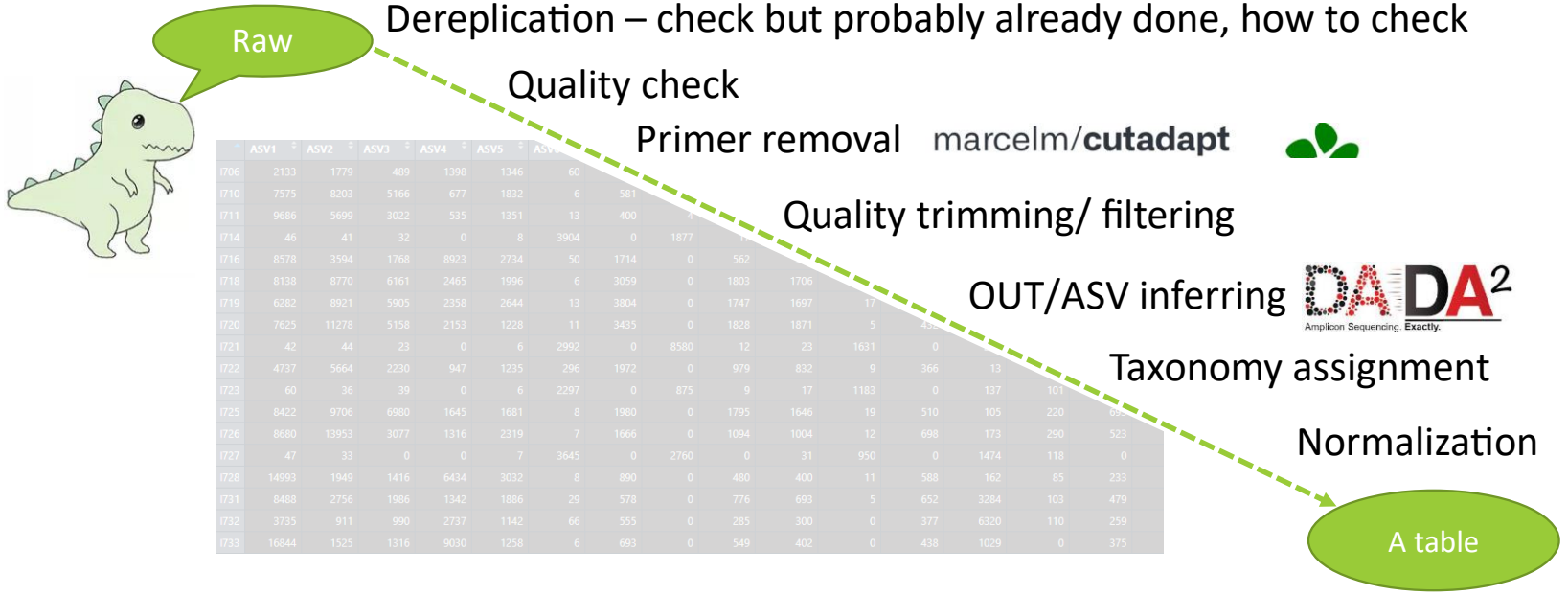
Relative paths !

./raw/

./src/amplicon_functions.R

Source: https://stackoverflow.com/questions/67357402/how-to-i-see-visual-of-my-folder-structure-in-rstudio

- Load data
- Primers trimming (cutadapt)
- Quality filter and trim
  (filterAndTrim)
- Errors model building
  (learnErrors)
- Dereplication
  (derepFastq)
- Sample interference
  (dada)
- Merging reads
  (mergePairs)
- Table
  (makeSequenceTable)
- Remove chimeras/bimeras
  (removeBimeraDenovo)
- Taxonomic assignment up to genus level
  (assignTaxonomy)
- Taxonomic assignment up to species level
  (addSpecies)
- Save results



- Load data
- Primers trimming (cutadapt)
- Quality filter and trim
  (filterAndTrim)
- Errors model building
  (learnErrors)
- Dereplication
  (derepFastq)
- Sample interference
  (dada)
- Merging reads
  (mergePairs)
- Table
  (makeSequenceTable)
- Remove chimeras/bimeras
  (removeBimeraDenovo)
- Taxonomic assignment up to genus level
  (assignTaxonomy)
- Taxonomic assignment up to species level
  (addSpecies)
- Save results


```
#trimming arguments and trimming
for (i in 1:length(rawF)) {
  cutadargs = c("-g",toString(primers_all[1]),
                "-G",toString(primers_all[2]),
                "-a",toString(primers_all[7]),
                "-A",toString(primers_all[8]),
                "-n",1, #for ITS change to 2
                "-o",paste0("./trimmed_reads/", gsub("raw","trim",basename(rawF[i]))),
                "-p",paste0("./trimmed_reads/", gsub("raw","trim",basename(rawR[i]))),
                rawF[i],
                rawR[i],
                "--minimum-length", 30, # minimum length , can be longer
                "--cores",0, #all available cores
                "--report=minimal")
  system2(cutapath, args=cutadargs) } # this line runs it

trimF <- sort(list.files(path_trim, pattern="trim_1.fq.gz", full.names = TRUE, recursive = F))
trimR <- sort(list.files(path_trim, pattern="trim_2.fq.gz", full.names = TRUE, recursive = F))
```

ITS case

Main differences (in processing):

- Variable length
- The primer sequence can occur more than once

- Load data
- Primers trimming (cutadapt)
- Quality filter and trim (filterAndTrim)
- Errors model building (learnErrors)
- Dereplication (derepFastq)
- Sample interference (dada)
- Merging reads (mergePairs)
- Table (makeSequenceTable)
- Remove chimeras/bimeras (removeBimeraDenovo)
- Taxonomic assignment up to genus level (assignTaxonomy)
- Taxonomic assignment up to species level (addSpecies)
- Save results

- Filter out low quality bases
- Generate report with info how many reads survived and saves filtered files (paths: filtF/R)
- In ok dataset 5-10% is still filtered out

```
filtered_report <- dada2::filterAndTrim(fwd = trimF, filt = filtF,
                                        rev = trimR, filt.rev =  filtR,
                    minLen = 30, # usually 20 dada2 cannot assign taxonomy to <30
                    rm.phix = T, # phiX genome – common contamination
                    maxN=0, # later steps do not allow N
                    truncQ=2, # cut when the quality goes below
                    maxEE=c(2,2), # max number of expected errors
                    compress=T,  # save as .gz (saves A LOT of space)
                    multithread=T, # doesn't work on windows anyways
                    verbose=T) # it talks to us while working
```

- Load data
- Primers trimming (cutadapt)
- Quality filter and trim (filterAndTrim)
- Errors model building (learnErrors)
- Dereplication (derepFastq)
- Sample interference (dada)
- Merging reads (mergePairs)
- Table (makeSequenceTable)
- Remove chimeras/bimeras (removeBimeraDenovo)
- Taxonomic assignment up to genus level (assignTaxonomy)
- Taxonomic assignment up to species level (addSpecies)
- Save results

Dada2 is not the only one !!!

- MOTHUR - DGC
- MOTHUR - Opticlust
- QIIME – Uclust
- QIIME – Deblur
- UNOISE3
- UPARSE

"DADA2 offered the best sensitivity, at the expense of decreased specificity compared to USEARCH-UNOISE3 and Qiime2-Deblur."

doi: 10.1371/journal.pone.0227434

What is dada2 and what it does ?

The dada2 package infers exact amplicon sequence variants (ASVs) from high-throughput amplicon sequencing data, replacing the coarser and less accurate OTU clustering approach. The dada2 pipeline takes as input demultiplexed fastq files, and outputs the sequence variants and their sample-wise abundances after removing substitution and chimera errors. Taxonomic classification is available via a native implementation of the RDP naive Bayesian classifier, and species-level assignment to 16S rRNA gene fragments by exact matching.

Schematic of OTU and DADA2 approaches towards amplicon sequencing errors.

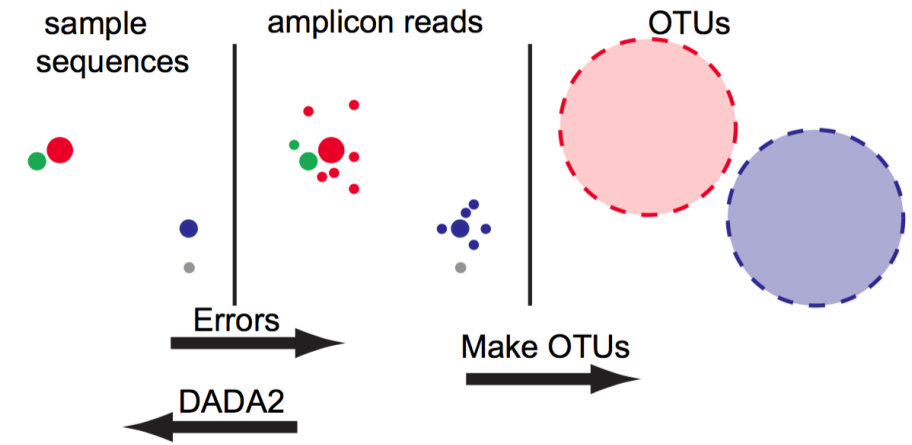


Figure 1. Circles represent identical sets of sequencing reads with size scaled by abundance and color corresponding to the true error-free sequence (there are four distinct sequences in the sample: red, green, blue and grey). Errors are introduced by amplicon sequencing from the left to the middle part of the diagram. OTU methods guard against false positive inferences by lumping similar sequences together. DADA2 uses a statistical model of amplicon errors to infer the underlying sample sequences directly, and thus tries to denoise the data from the middle to the left.

Illumina Miseq error rates as a function of quality.

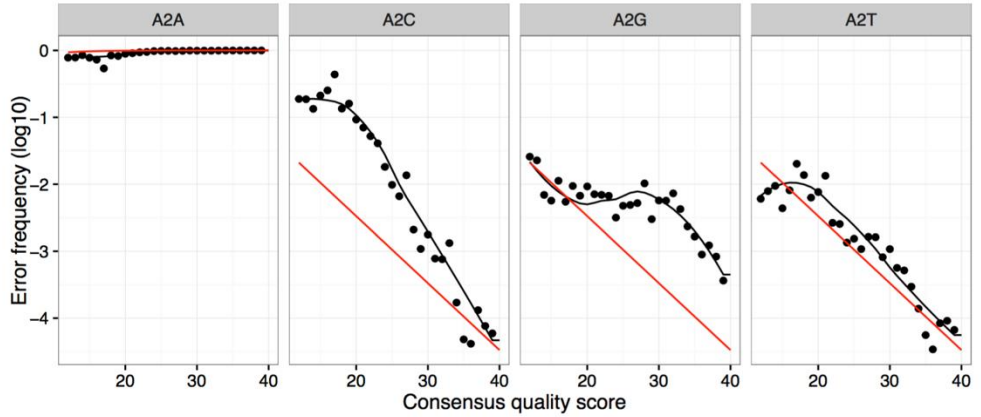


Figure 8. The forward-read error rates observed in the 142 pooled samples from MacIntyre 2015 are shown for the case where the correct base is an A. The x-axis shows the quality score; the y-axis the frequency of the specified transition. Dots show the observed frequencies, the black line the error model inferred by DADA2 using its default loess fitting, and the red line the expected rates given the nominal definition of the quality score: $Q = -10log_{10}(p_{err})$. Illumina quality scores are quite informative about substitution error rates, but systematic deviations from the expected rates are observed. This plot was generated by the `plotErrors` function in the DADA2 R package.

Source: Callahan et al. 2016

- Load data
- Primers trimming (cutadapt)
- Quality filter and trim
  (filterAndTrim)
- Errors model building
  (learnErrors)
- Dereplication
  (derepFastq)
- Sample interference
  (dada)
- Merging reads
  (mergePairs)
- Table
  (makeSequenceTable)
- Remove chimeras/bimeras
  (removeBimeraDenovo)
- Taxonomic assignment up to genus level
  (assignTaxonomy)
- Taxonomic assignment up to species level
  (addSpecies)
- Save results

- Building parametric error models for “expected errors” the essence of dada2

```
errF <- dada2::learnErrors(filtF,
                           nbases = 1e8, # how much data is used to build the model
                           errorEstimationFunction = loessErrfun_mod4,# skip for NOT novaseq
                           randomize = T,
                           MAX_CONSIST = 15,
                           multithread =T,
                           verbose = TRUE)
```

Illumina Miseq error rates as a function of quality.

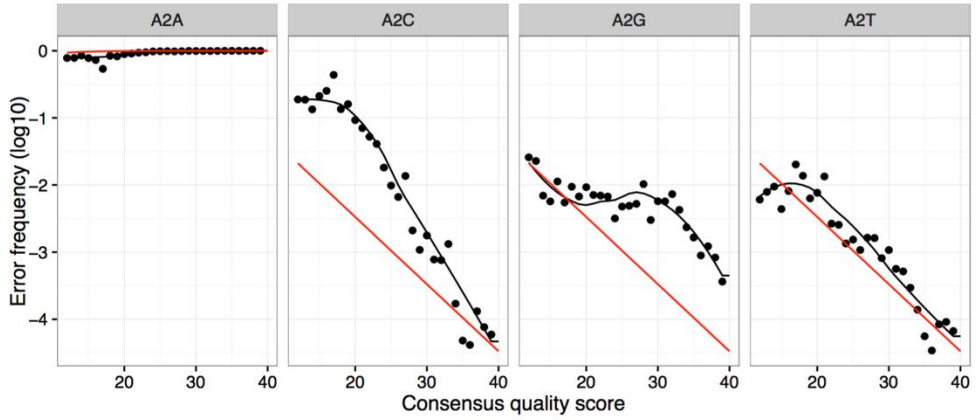
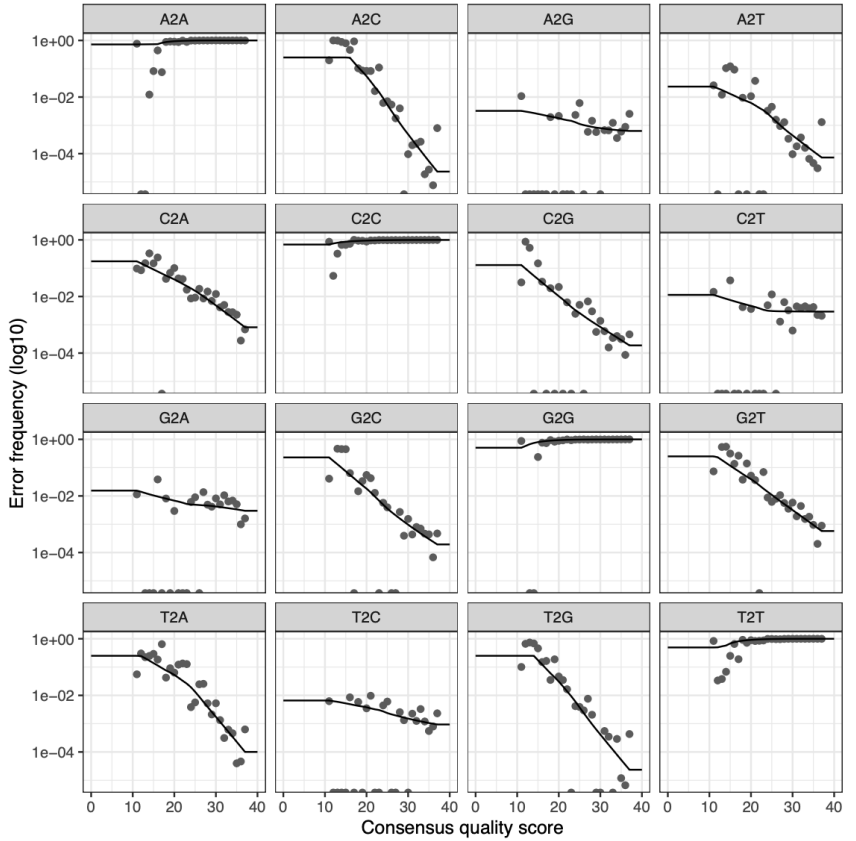


Figure 8. The forward-read error rates observed in the 142 pooled samples from MacIntyre 2015 are shown for the case where the correct base is an A. The x-axis shows the quality score; the y-axis the frequency of the specified transition. Dots show the observed frequencies, the black line the error model inferred by DADA2 using its default loess fitting, and the red line the expected rates given the nominal definition of the quality score: $Q = -10log_{10}(p_{err})$. Illumina quality scores are quite informative about substitution error rates, but systematic deviations from the expected rates are observed. This plot was generated by the `plotErrors` function in the DADA2 R package.

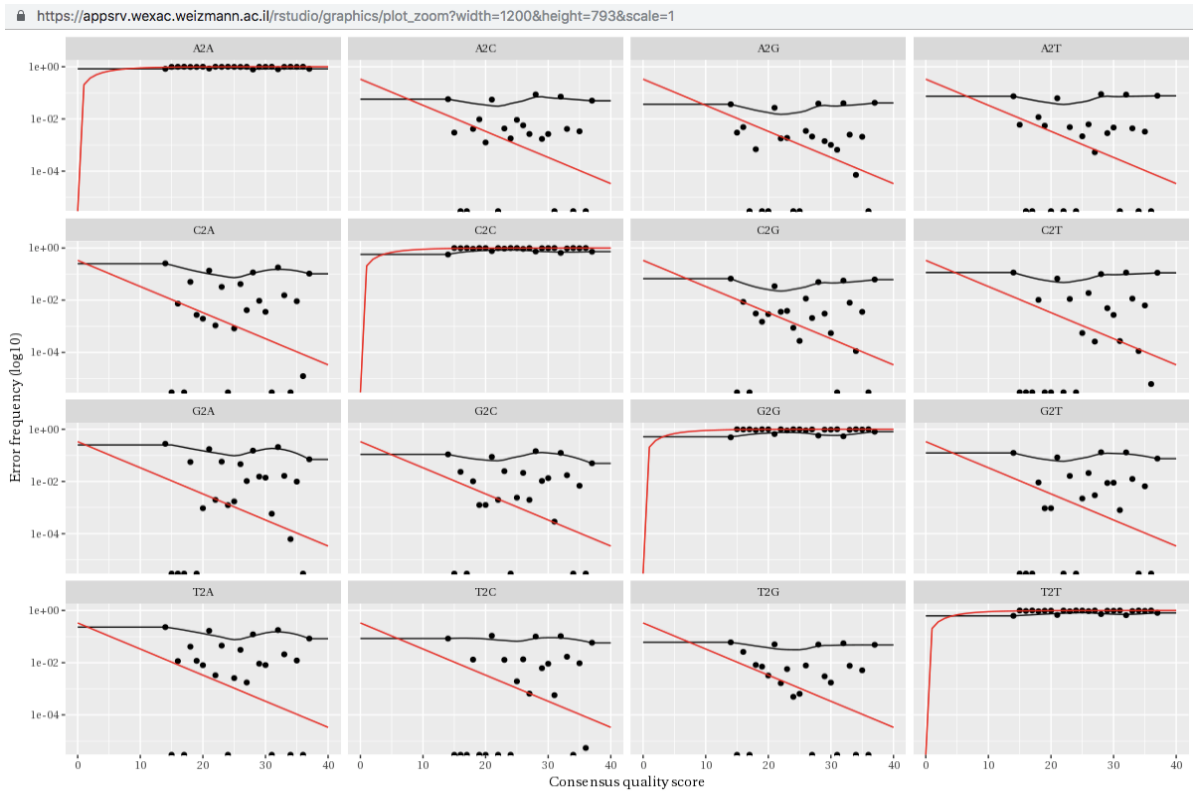
```r
#libs
library("dada2")
library("magrittr")

#functions
source("./src/novaseq.R")

#get sample names
sample_names = gsub("_raw_1.fq.gz","",list.files("./raw_data", pattern="_1.fq.gz"))

################filtering; provide path to trimmed reads - path trim

#paths to filtered reads
filtF <- sort(list.files("./processed/2.filtered/", pattern="1.fq.gz", full.names = TRUE))
filtR <- sort(list.files("./processed/2.filtered/", pattern="2.fq.gz", full.names = TRUE))

################ learning error rates for novaseq sequencing
##make sure "magrittr" is loaded
set.seed(35) # makes the error learning consistent

errF <- dada2::learnErrors(filtF,
                    nbases = 1e8,
                    errorEstimationFunction = loessErrfun_mod4,# skip for NOT novaseq
                    randomize = T,
                    MAX_CONSIST = 12,
                    multithread =8,
                    verbose = T)

errR <- dada2::learnErrors(filtR,
                    nbases = 1e8,
                    errorEstimationFunction = loessErrfun_mod4, # skip for NOT novaseq
                    randomize = T,
                    MAX_CONSIST = 12,
                    multithread =8,
                    verbose = T)

####saving error graphs
pdf(file = "./reports/dada2_error_plots.pdf")
plotErrors(errF)
plotErrors(errR)
dev.off()
####
```

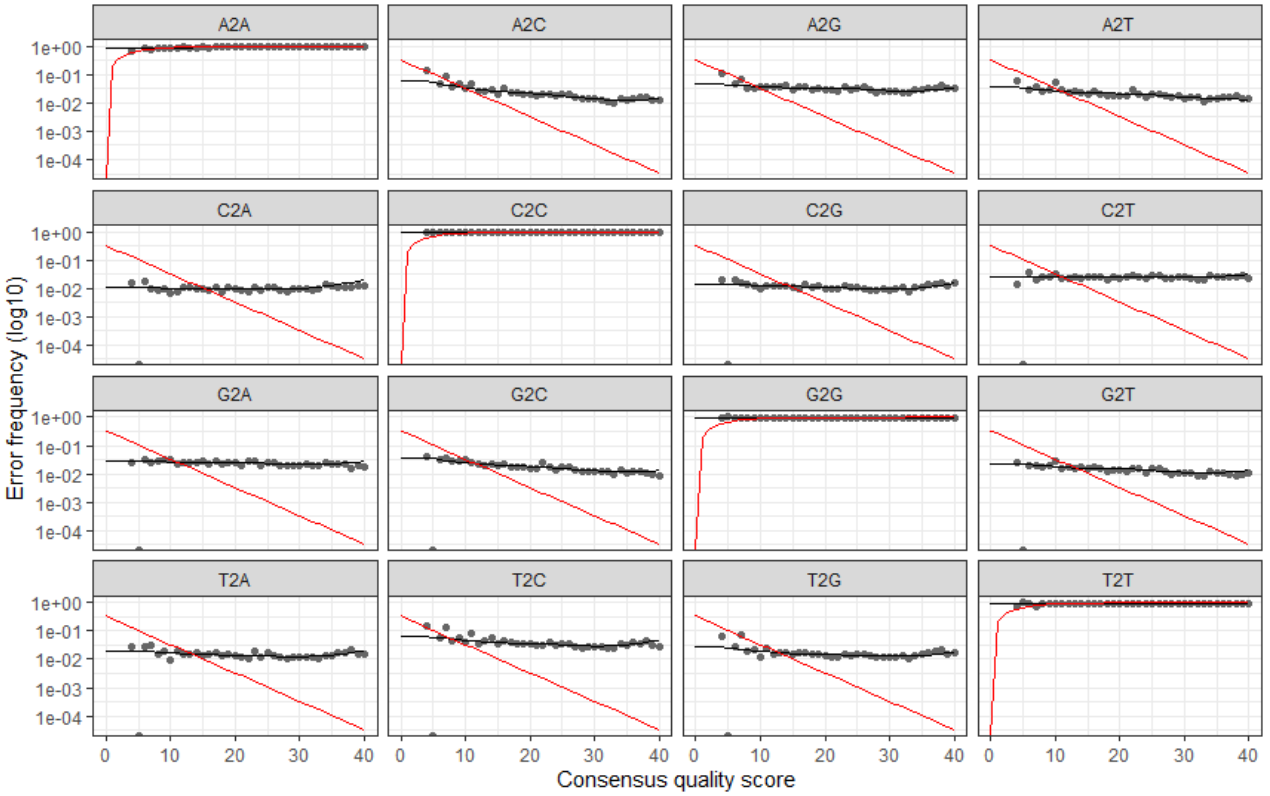
Do you know how to check it ?

```r
errF <- dada2::learnErrors(filtF,
                    nbases = 1e8,
                    errorEstimationFunction = loessErrfun_mod4,
                    randomize = T,
                    MAX_CONSIST = 12,
                    multithread =8,
                    verbose = T)
```



What if…. ?





IN

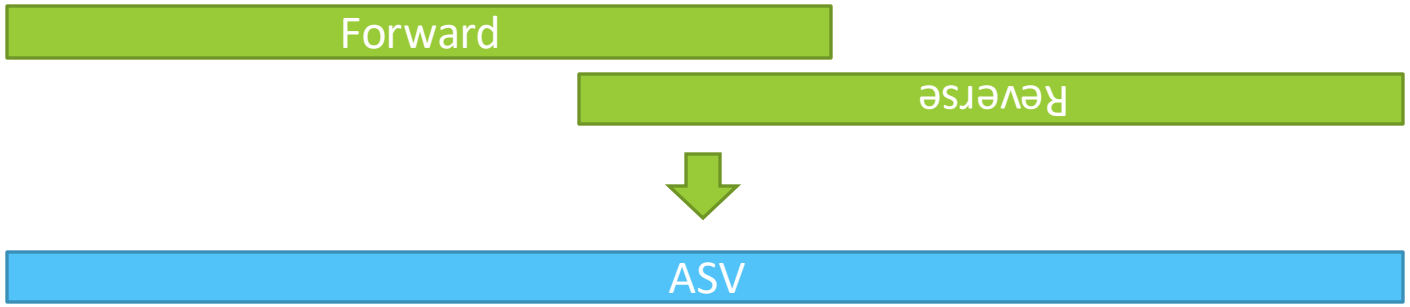
OUT

- Load data
- Primers trimming (cutadapt)
- Quality filter and trim (filterAndTrim)
- Errors model building (learnErrors)
- Dereplication (derepFastq)
- Sample interference (dada)
- Merging reads (mergePairs)
- Table (makeSequenceTable)
- Remove chimeras/bimeras (removeBimeraDenovo)
- Taxonomic assignment up to genus level (assignTaxonomy)
- Taxonomic assignment up to species level (addSpecies)
- Save results

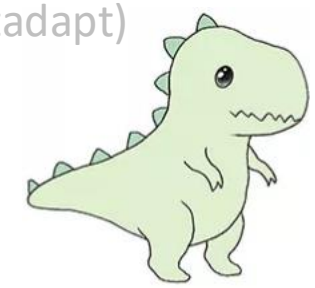
- Dereplication – combines identical sequences, saves abundance # optional but recommended
- Interference – ASV's are generated using error models

```
dada2::dada(derepF, err=errF, multithread=T, pool=F)
# pool=F increased sensitivity (default)
# pool=T saves time, pool from all the samples
# pool="pseudo" saves some time
```

- F and R reads come together, overlap 12 bp (default)



- Load data
- Primers trimming (cutadapt)
- Quality filter and trim
  (filterAndTrim)
- Errors model building
  (learnErrors)
- Dereplication
  (derepFastq)
- Sample interference
  (dada)
- Merging reads
  (mergePairs)
- Table
  (makeSequenceTable)
- Remove chimeras/bimeras
  (removeBimeraDenovo)
- Taxonomic assignment up to genus level
  (assignTaxonomy)
- Taxonomic assignment up to species level
  (addSpecies)
- Save results

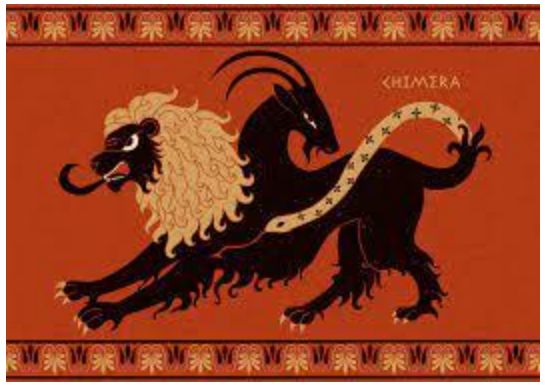


Raw

A table

Still not the final one …
It has ASV sequences (!) as column names

removeBimeraDenovo(seqtab, method="consensus", multithread=TRUE, verbose=TRUE)



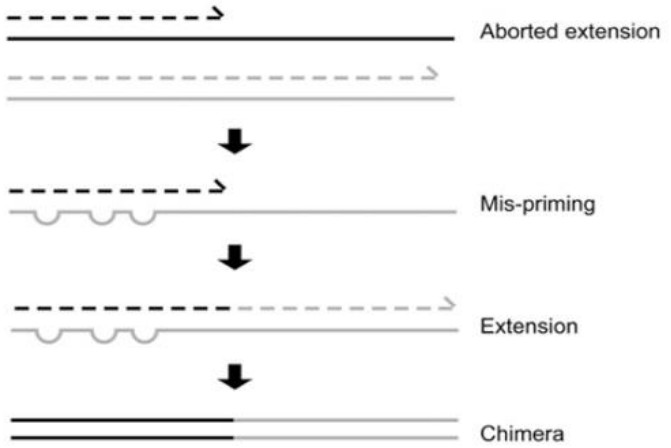


Figure 1. Formation of chimeric sequences during PCR. An aborted extension product from an earlier cycle of PCR can function as a primer in a subsequent PCR cycle. If this aborted extension product anneals to and primes DNA synthesis from an improper template, a chimeric molecule is formed.

From Haas *et al.* (2011) Chimeric 16S rRNA sequence formation and detection in Sanger and 454-pyrosequenced PCR amplicons, *Genome Research*.

- Load data
- Primers trimming (cutadapt)
- Quality filter and trim (filterAndTrim)
- Errors model building (learnErrors)
- Dereplication (derepFastq)
- Sample interference (dada)
- Merging reads (mergePairs)
- Table (makeSequenceTable)
- Remove chimeras/bimeras (removeBimeraDenovo)
- Taxonomic assignment up to genus level (assignTaxonomy)
- Taxonomic assignment up to species level (addSpecies)
- Save results

- Dada2 uses naive Bayesian classifier method to assign taxonomy

assignTaxonomy(seqtab.nochim, "./SILVA138/silva_nr99_v138.1_train_set.fa.gz", multithread=TRUE)

You need to use **pre-formatted database**, or format it yourself

Info about both is here: https://benjjneb.github.io/dada2/training.html

Maintained:

- **Silva version 138.1 - UPDATED Mar 10, 2021**, version 132, version 128, version 123
  - NOTE: As of Silva version 138, the official DADA2-formatted reference fastas are optimized for classification of Bacteria a Archaea, and are not suitable for classifying Eukaryotes.
- **RDP trainset 18**, RDP trainset 16, RDP trainset 14
- **UNITE (use the General Fasta releases, "All eukaryotes")**
- *Deprecated: GreenGenes version 13.8 (the source GreenGenes database is no longer being maintained)*

Contributed:

- **GTDB Version 202: Genome Taxonomy Database** (More info on GTDB)
  - Version 86 for `assignTaxonomy` and `assignSpecies`
- RefSeq + RDP (NCBI RefSeq 16S rRNA database supplemented by RDP)
  - Reference files formatted for `assignTaxonomy`
  - Reference files formatted for `assignSpecies`
- HitDB version 1 (Human InTestinal 16S rRNA)
- Human Oral Microbiome Database: HOMD
- MiDAS: Field Guide to the Microbes of Activated Sludge and Anaerobic Digesters
- MIDORI Reference 2 (for taxonomic assignments of Eukaryota mitochondrial DNA sequences)
- RDP fungi LSU trainset 11
- Silva Eukaryotic 18S, v132 & v128
- nifH ARB, version 1
- PR2 version 4.7.2+. SEE NOTE BELOW.

Note: PR2 has different `taxLevels` than the dada2 default. When assigning taxonomy against PR2, use the following:

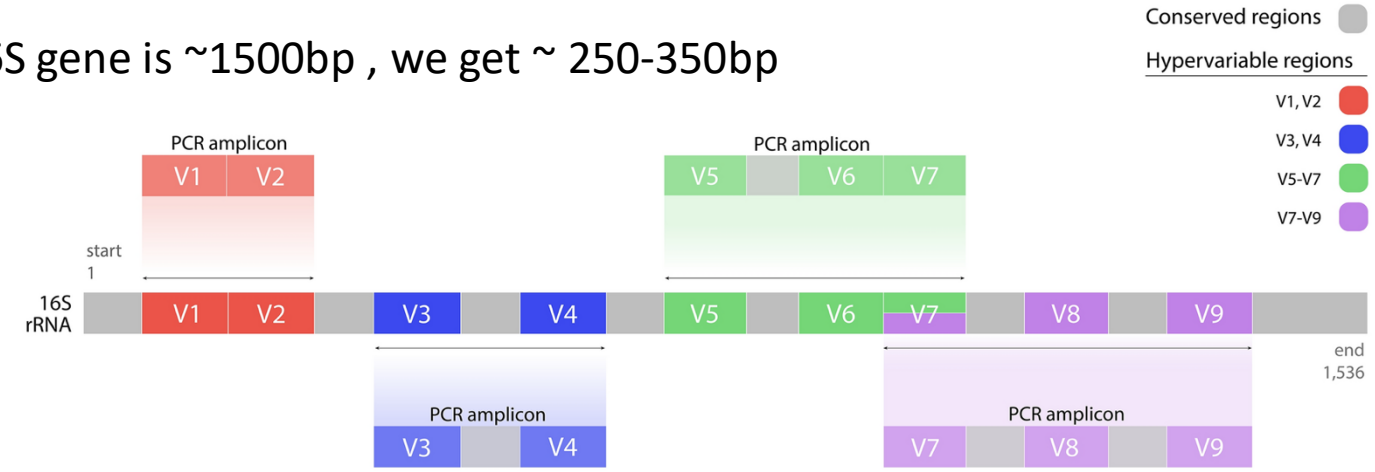
```
assignTaxonomy(..., taxLevels = c("Kingdom","Supergroup","Division","Class","Order","Family","Genus","Species"))
```

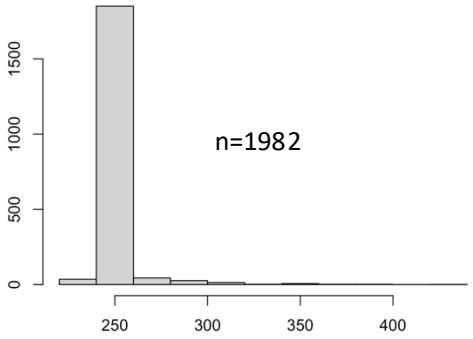
https://doi.org/10.1186/s12864-017-3501-4
SILVA, RDP, Greengenes, NCBI and OTT — how do these taxonomies compare?

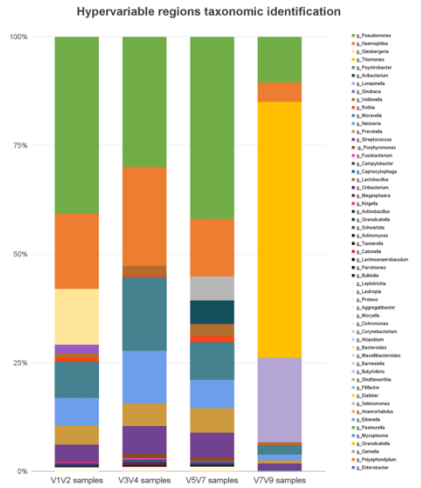
Species assignment

- fundamental question: What is a bacteria species? And why it matters…

- The full 16S gene is ~1500bp , we get ~ 250-350bp







López-Aladid at al. 2023

- Load data
- Primers trimming (cutadapt)
- Quality filter and trim (filterAndTrim)
- Errors model building (learnErrors)
- Dereplication (derepFastq)
- Sample interference (dada)
- Merging reads (mergePairs)
- Table (makeSequenceTable)
- Remove chimeras/bimeras (removeBimeraDenovo)
- Taxonomic assignment up to genus level (assignTaxonomy)
- Taxonomic assignment up to species level (addSpecies)
- Save results

- Checkpoints

```
##checkpoint1
save.image(file = "rawr_amplicons.RData")
##
```

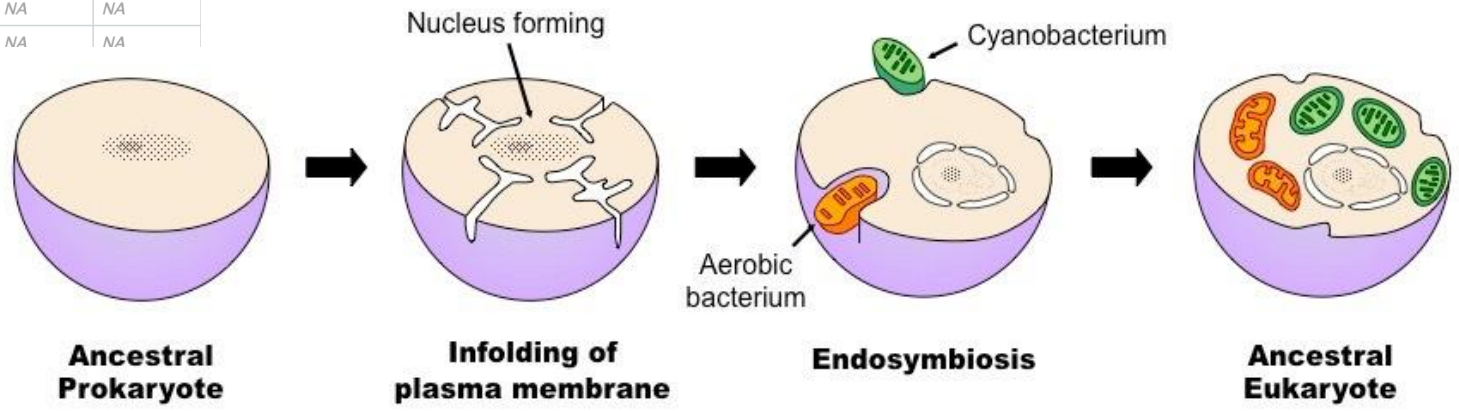
- Saving results:
  - Tables
  - *write.csv(taxonomy, file="./results/tax_table.csv", row.names=T)*
  - RDS Objects – highly recomended
  - *saveRDS(taxonomy, file="./results/tax_table.RDS")*

Filtering and the most common contaminants

- Chloroplasts and mitochondria
- Unclassified reads
- Eucaryota ?

	Kingdom	Phylum	Class	Order	Family	Genus	Species
ASV49	Bacteria	Proteobacteria	Alphaproteobacteria	Rickettsiales	Mitochondria	*NA*	*NA*
ASV118	Bacteria	Proteobacteria	Alphaproteobacteria	Rickettsiales	Mitochondria	*NA*	*NA*
ASV144	Bacteria	Proteobacteria	Alphaproteobacteria	Rickettsiales	Mitochondria	*NA*	*NA*
ASV158	Bacteria	Proteobacteria	Alphaproteobacteria	Rickettsiales	Mitochondria	*NA*	*NA*
ASV221	Bacteria	Proteobacteria	Alphaproteobacteria	Rickettsiales	Mitochondria	*NA*	*NA*
ASV237	Bacteria	Proteobacteria	Alphaproteobacteria	Rickettsiales	Mitochondria	*NA*	*NA*
ASV243	Bacteria	Proteobacteria	Alphaproteobacteria	Rickettsiales	Mitochondria	*NA*	*NA*

	Kingdom	Phylum	Class	Order	Family	Genus	Species
ASV160	Bacteria	Cyanobacteria	Cyanobacteriia	Chloroplast	*NA*	*NA*	*NA*
ASV183	Bacteria	Cyanobacteria	Cyanobacteriia	Chloroplast	*NA*	*NA*	*NA*
ASV297	Bacteria	Cyanobacteria	Cyanobacteriia	Chloroplast	*NA*	*NA*	*NA*
ASV335	Bacteria	Cyanobacteria	Cyanobacteriia	Chloroplast	*NA*	*NA*	*NA*
ASV362	Bacteria	Cyanobacteria	Cyanobacteriia	Chloroplast	*NA*	*NA*	*NA*
ASV535	Bacteria	Cyanobacteria	Cyanobacteriia	Chloroplast	*NA*	*NA*	*NA*



suplement

We ended with creating two files !
>seqtab.nochim
>taxonomy

Those are JUST tables

seqtab.nochim

	TACGGAGGGTGCAAGCGTTAATCGGAATTACTGGGCGTAAAGCGCACGCAGGCGGTCTGTTAAGTCAGATGTGAAATCCCC
1776	27767
1777	26603
1778	10539
1779	52375
1780	56209
1781	70882
1782	49657
1783	41002
1784	37660

taxonomy

	Kingdom	Phylum	Class	Order
CGGAATTACTGGGCGTAAAGC...	Bacteria	Proteobacteria	Gammaproteobacteria	Enterobacterales
CGGAATTACTGGGCGTAAAGC...	Bacteria	Proteobacteria	Gammaproteobacteria	Pseudomonadales
CGGAATTACTGGGCGTAAAGC...	Bacteria	Proteobacteria	Gammaproteobacteria	Enterobacterales
GGATTTATTGGGCGTAAAGCG...	Bacteria	Firmicutes	Bacilli	Lactobacillales
CGGAATTACTGGGCGTAAAGC...	Bacteria	Proteobacteria	Gammaproteobacteria	Enterobacterales
CGGAATTACTGGGCGTAAAGC...	Bacteria	Proteobacteria	Gammaproteobacteria	Enterobacterales
CGGAATTACTGGGCGTAAAGC...	Bacteria	Proteobacteria	Alphaproteobacteria	Rhodobacterales
CGGAATCATTGGGCGTAAAGA...	Bacteria	Actinobacteriota	Actinobacteria	Micrococcales
CGGAATTACTGGGCGTAAAGC...	Bacteria	Proteobacteria	Alphaproteobacteria	Sphingomonadales
CGGAATTACTGGGCGTAAAGC...	Bacteria	Proteobacteria	Alphaproteobacteria	Sphingomonadales
CGGAATTACTGGGCGTAAAGC...	Bacteria	Proteobacteria	Alphaproteobacteria	Sphingomonadales
CGGAATTATTGGGCGTAAAGG...	Bacteria	Firmicutes	Desulfitobacteriia	Desulfitobacteriales
CGGAATTACTGGGCGTAAAGC...	Bacteria	Proteobacteria	Gammaproteobacteria	Enterobacterales

Creating **phyloseq object** is a way to keep all the data in one place as one object containing:

- ASV/OUT object
- Taxonomy table
- metadata
- ASV/OUT sequences
- Potentially other stuff


```
phyloseq-class experiment-level object
otu_table()   OTU Table:         [ 577 taxa and 9 samples ]
sample_data() Sample Data:       [ 9 samples by 4 sample variables ]
tax_table()   Taxonomy Table:    [ 577 taxa by 6 taxonomic ranks ]
refseq()      DNAStringSet:      [ 577 reference sequences ]
>
```

Phyloseq:

- ASV/OUT object (obligatory)
- Taxonomy table (recomended)
- Metadata (recomended)
- ASV/OUT sequences (optional)
- Phylogeny (optional)

metadata

	organism	habitat	replicate	treatment
I776	unicorn	magic_wood	1	uni_wood_1_16S
I777	unicorn	magic_wood	2	uni_wood_2_16S
I778	unicorn	magic_wood	3	uni_wood_3_16S
I779	dragon	wonderland	1	dra_wonder_1_16S
I780	dragon	wonderland	2	dra_wonder_2_16S
I781	dragon	wonderland	3	dra_wonder_3_16S
I782	troll	magic_wood	1	tro_wood_1_16S
I783	troll	magic_wood	2	tro_wood_2_16S
I784	troll	magic_wood	3	tro_wood_3_16S