UNIVERSITÉ DE FRIBOURG
UNIVERSITÄT FREIBURG

WORKSHOP FRIBOURG 2025

TARGETED OR AMPLICON SEQUENCING

Laurent Falquet

SIB
Swiss Institute of Bioinformatics

The choice of the method depends on 3 main questions

Who is there?

catalogue of species, diversity, genus, etc.
distribution (how many of each)

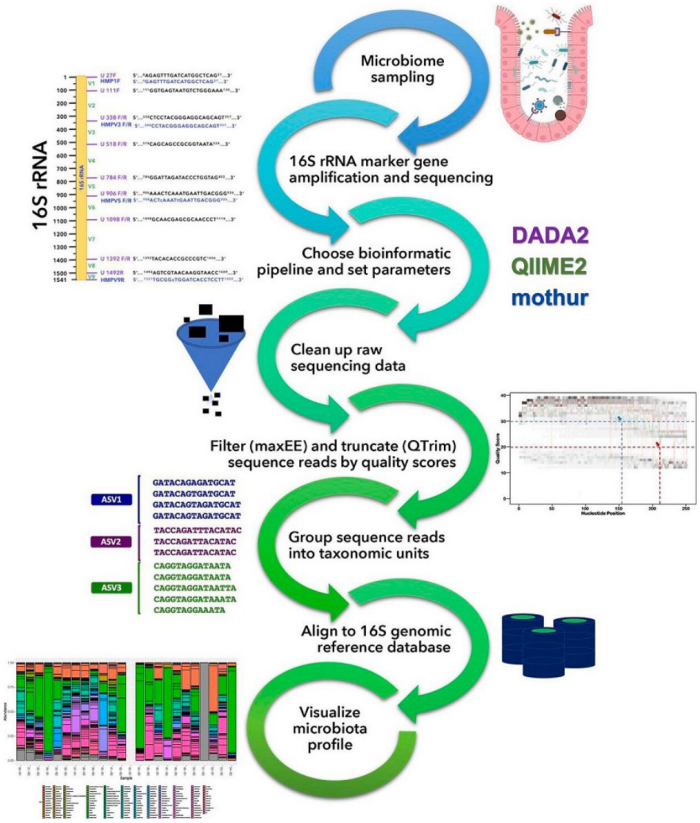
What are they doing?

genes/proteins, GO terms, metabolic pathways
functional annotation, bioprospection

How do they compare?

pairwise or multiple comparisons
correlation with environmental factors

A typical pipeline



Nayman et al., Journal of Medical Microbiology 2023;72:001756 DOI 10.1099/jmm.0.001756

Targeted or Amplicon sequencing

Pipeline examples

Mothur (to OTUs)

DADA2 (to ASVs)

QIIME2 (to OTUs or to ASVs)

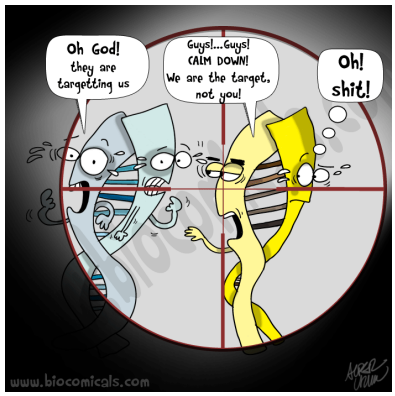
PIPITS3 or PipeCraft2 or NextITS (to OTUs or ASVs for fungi)

Targeted or Amplicon sequencing

Library preparation requires PCR amplification
QC
Clean reads
optional: combine overlapping PE
Align reads to reference database
Clusterize reads to OTUs
Assign OTUs/ASVs to taxonomy

Tools (examples):

- Mothur
- Qiime2
- DADA2
- MapSeq
- MEGAN
- Metaphlan
- PIPITS (for fungi)
- …

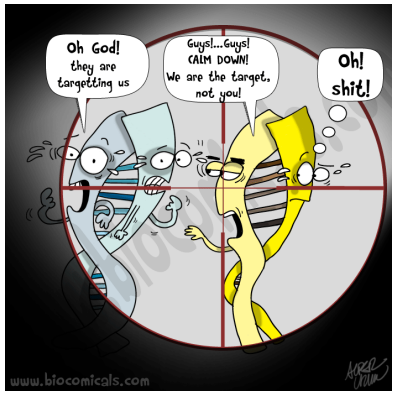


Targeted or Amplicon sequencing

Library preparation requires PCR amplification
QC
Clean reads
optional: combine overlapping PE
Align reads to reference database
Clusterize reads to OTUs or ASVs
Assign OTUs/ASVs to taxonomy

Tools (examples):

- Mothur
- Qiime2
- DADA2
- MapSeq
- MEGAN
- Metaphlan
- PIPITS (for fungi)
- …



OTUs vs ASVs

OTU = Operational Taxonomic Unit
ASV = Amplicon Sequence Variant

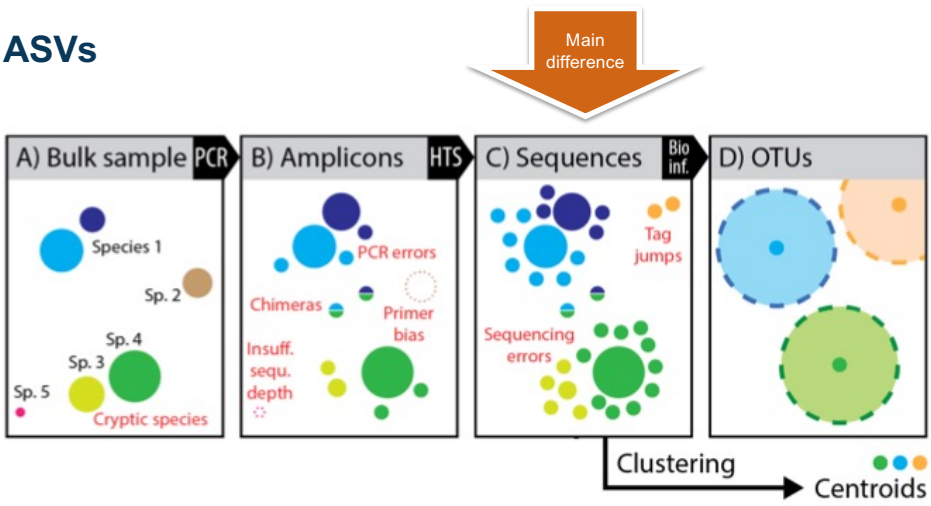
Both are *in silico* classifications of the PCR amplicons with two opposite philosophies:

OTUs are clustered sequences to centroids
ASVs are denoised sequences to haplotypes

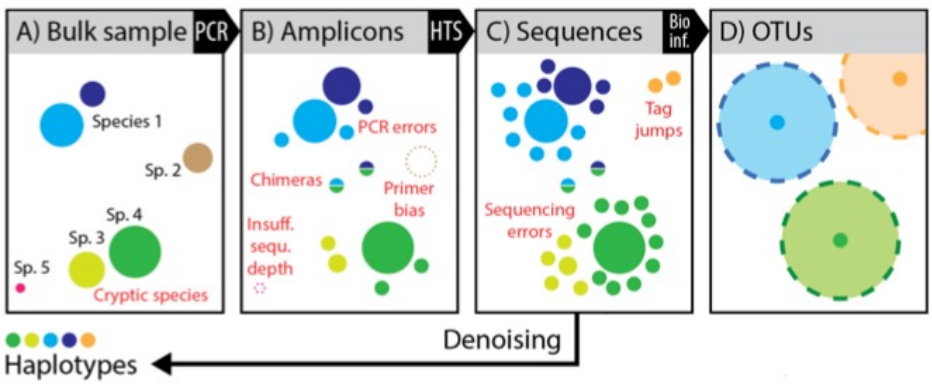
The final goal is to obtain feature tables and representative sequences

OTUs vs ASVs

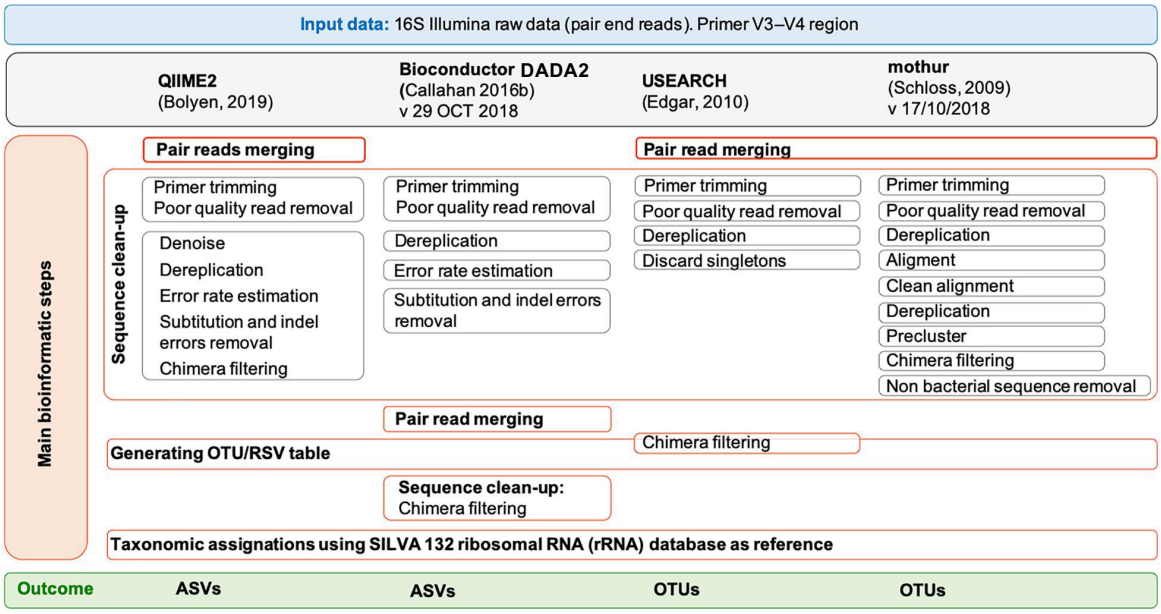
OTUs



ASVs



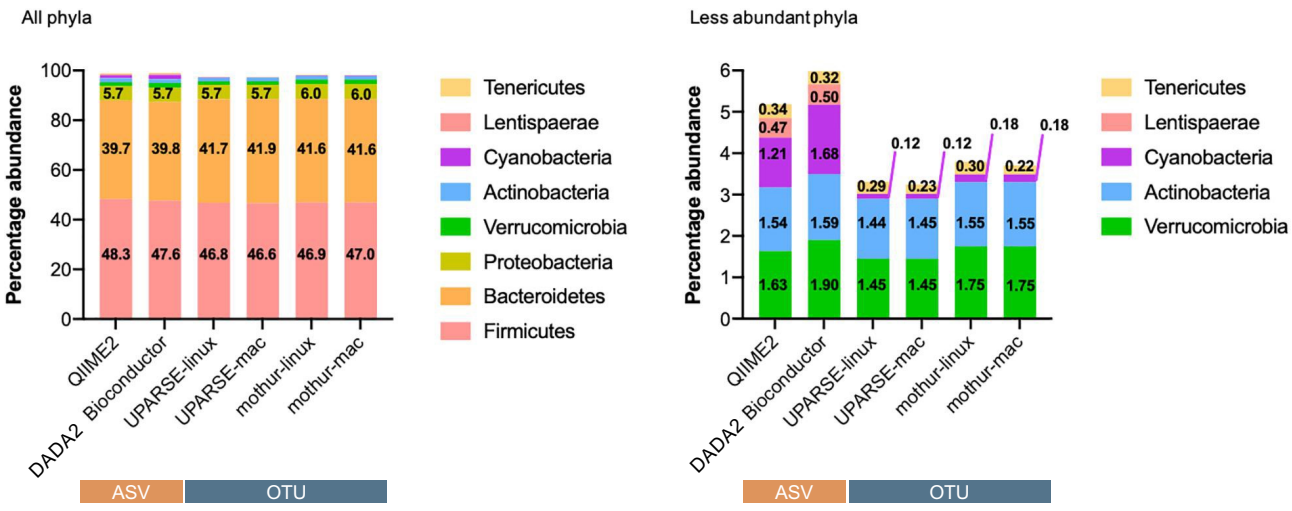
Comparison of pipelines



	QIIME2	Bioconductor DADA2	UPARSE	Mothur
Open source	√ Yes	√ Yes	Ø **32-bit only**	√ Yes
Installation	√ Clear online instruction Ø Specific version dependency[a]	√Ø Easy for R user	√ Installation not required	√ Installation not required
User support	√ Tutorials, forum, SOPs	√ Tutorials, forum, SOPs	√ Video tutorials, forum, SOPs	√ Tutorials, forum, SOPs
Specific computer skills	√ Not required	Ø Required	√ Not required	Ø Required

[a]*Marks indicate pros and cons, respectively. "Ø" and "√" Miniconda 2 or 3 depending on python 2.7 or 3.7, respectively.*

Comparison of pipelines



Marizzoni et al, Front. Microbiol., 17 June 2020
https://doi.org/10.3389/fmicb.2020.01262

What is an OTU? Operational taxonomic unit (OTU)

From wikipedia

An **Operational Taxonomic Unit** (OTU) is an operational definition used to classify groups of closely related individuals.

For several years, OTUs have been the most commonly used units of diversity, especially when analysing small subunit 16S (for prokaryotes) or 18S rRNA (for eukaryotes) marker gene sequence datasets.

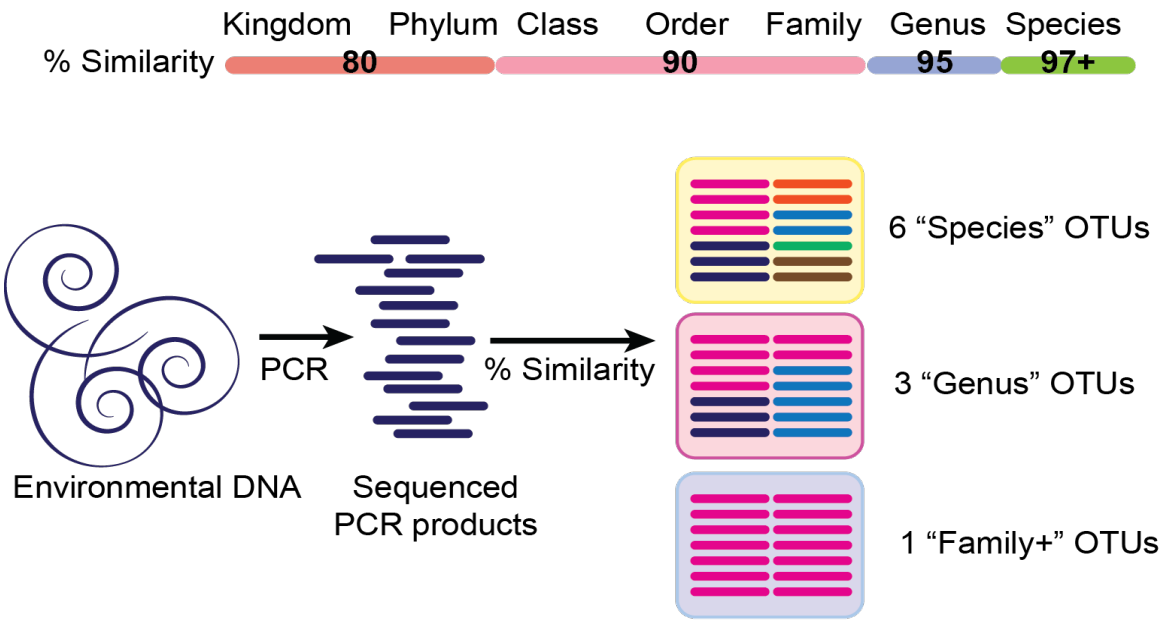
Sequences can be clustered according to their similarity to one another, and operational taxonomic units are defined based on the similarity threshold set by the researcher.

The number of OTUs defined may be inflated due to errors in DNA sequencing.

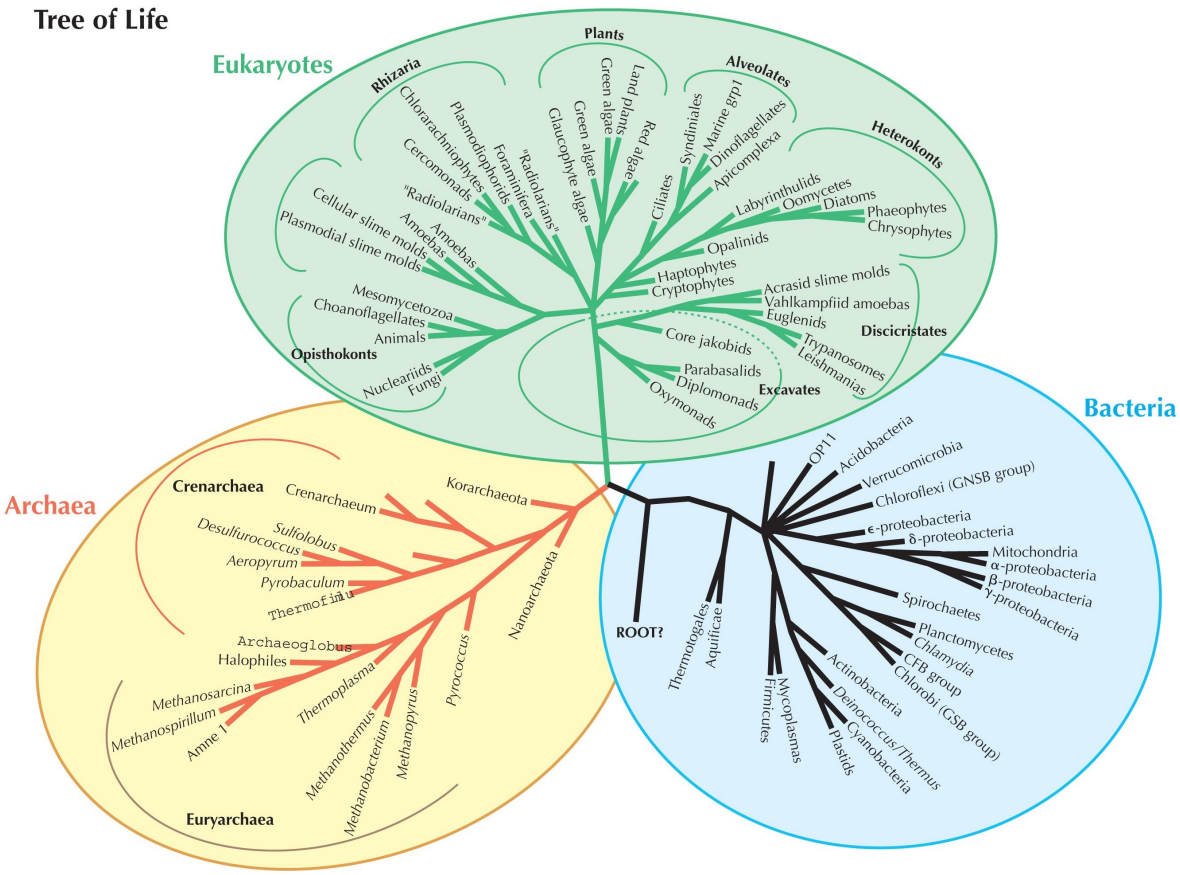
For 16S rRNA OTUs (~species) are delineated with a 3% sequence dissimilarity and higher taxa with increasingly larger dissimilarity

OTUs or groups of OTUs can later be assigned taxonomic names

OTU Operational Taxonomic Unit

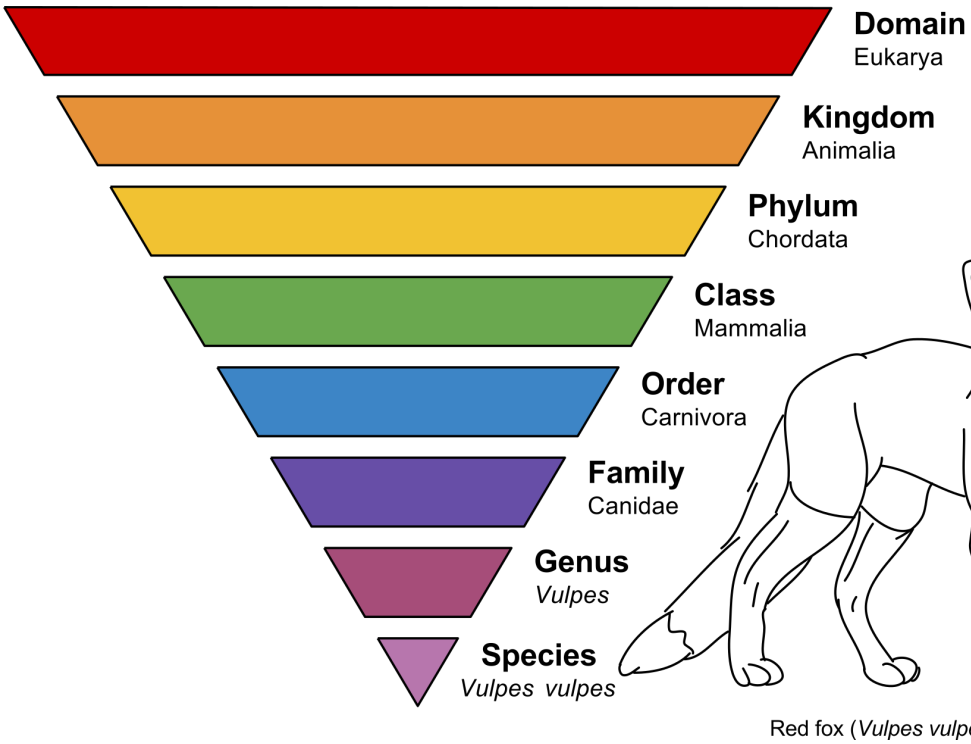


The tree of life (genealogy of the living organisms)



UNI FR

Taxonomic classification?

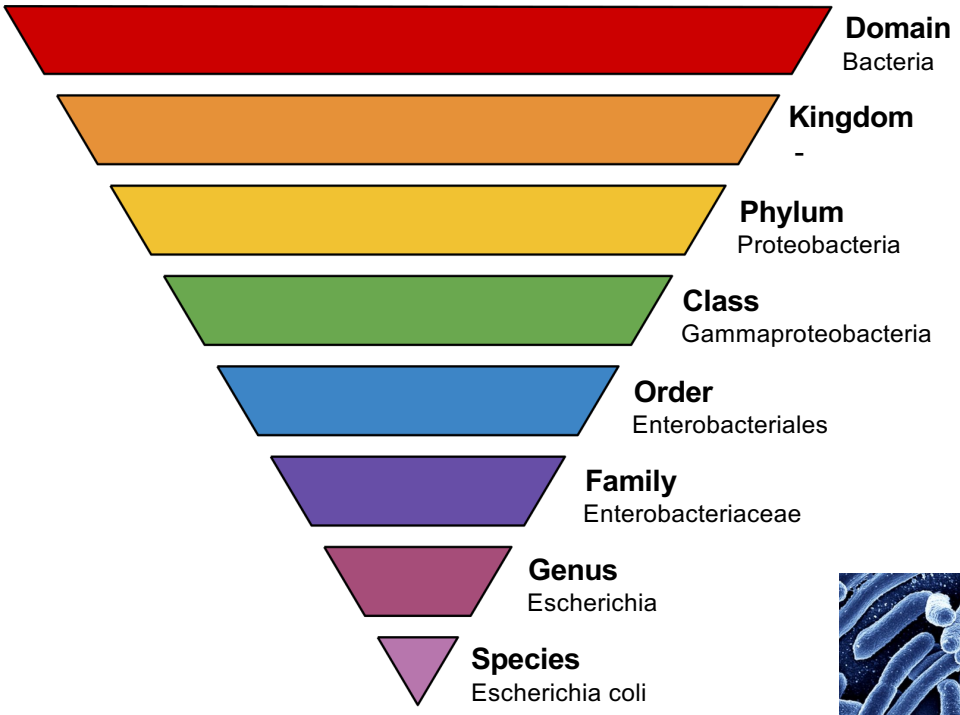


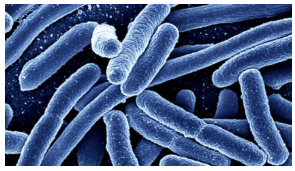
Red fox (*Vulpes vulpes*)

https://en.wikipedia.org/wiki/Taxonomic_rank

UNI FR

Taxonomy of bacteria?





Two feature tables and representative sequences

The list of OTUs or ASVs and read counts associated to a taxon

OTU	Size	Taxonomy
Otu00001	530053	Bacteria(100);Proteobacteria(100);Gammaproteobacteria(100);Pseudomonadales(100);Pseudomonadaceae(100);Pseudomonas(100);
Otu00002	298855	Bacteria(100);Proteobacteria(100);Betaproteobacteria(100);Methylophilales(100);Methylophilaceae(100);Methylophilus(90);
Otu00003	546152	Bacteria(100);Proteobacteria(100);Alphaproteobacteria(100);Rhizobiales(100);Rhizobiaceae(100);Rhizobium(100);
Otu00004	269985	Bacteria(100);Proteobacteria(100);Gammaproteobacteria(100);Pseudomonadales(100);Pseudomonadaceae(100);Pseudomonas(99);
Otu00005	206811	Bacteria(100);Proteobacteria(100);Betaproteobacteria(100);Burkholderiales(100);Burkholderiaceae(100);Burkholderia(100);
Otu00006	100666	Bacteria(100);Proteobacteria(100);Betaproteobacteria(100);Burkholderiales(100);Comamonadaceae(100);Variovorax(96);
Otu00007	54191	Bacteria(100);Proteobacteria(100);Gammaproteobacteria(100);Xanthomonadales(100);Xanthomonadaceae(100);Stenotrophomonas(58);
...		

The list of OTUs or ASVs and read counts per sample

label	Group	numOtus	Otu00001	Otu00002	Otu00003	Otu00004	Otu00005	Otu00006	Otu00007	Otu00008 ...
0.03	bdg-ws-1a	11867	17387	27426	11854	207	53	5808	5964	2006 ...
0.03	bdg-ws-1b	2999	28979	12	98712	87	127	17738	727	771 ...
...										

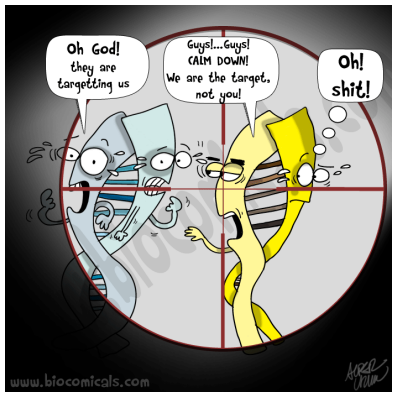
Representative sequences

Each OTU **has a single associated sequence for its cluster (centroid)**

Each ASV **is a unique sequence**

Targeted or Amplicon sequencing

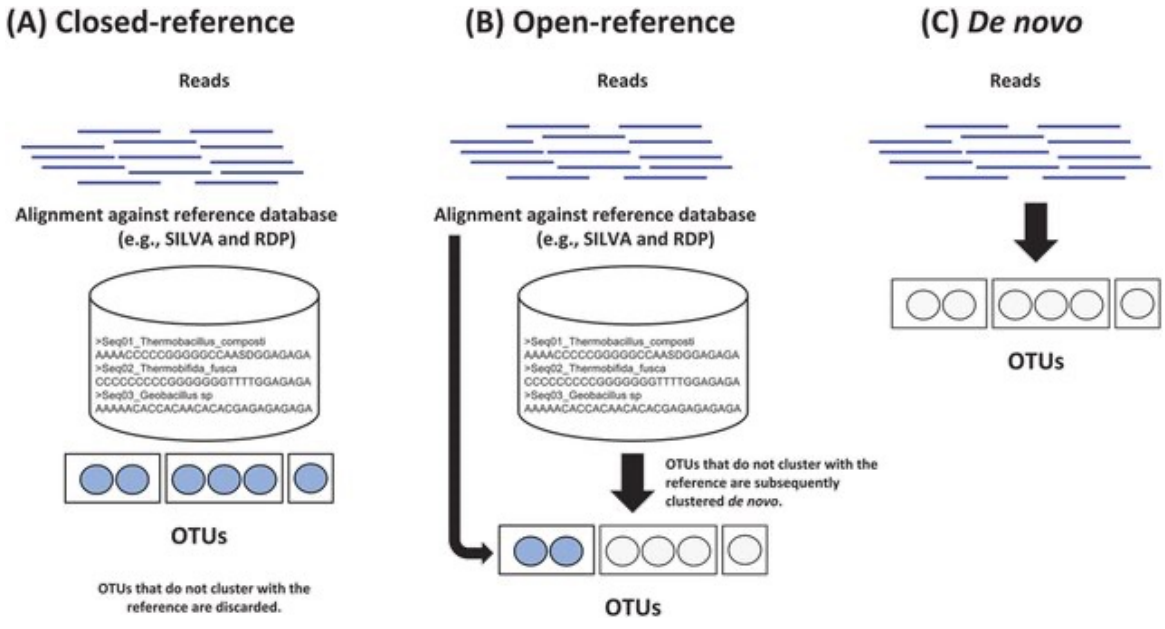
Library preparation requires PCR amplification
QC
Clean reads
optional: combine overlapping PE
Align reads to reference database
Clusterize reads to OTUs or ASVs
Assign OTUs/ASVs to taxonomy



Tools (examples):

- Mothur
- Qiime2
- DADA2
- MapSeq
- MEGAN
- Metaphlan
- PIPITS (for fungi)
- …

De novo vs Closed *vs* Open OTU taxonomy assignment



Advantages of ASVs vs OTUs

ASVs combine advantages of both closed-reference OTUs and *de novo* OTUs

- computational costs that scale linearly with study size
- simple comparison between independently processed data sets
- allow classification prediction
- accurate measurement of diversity
- applicability to communities not fully in reference databases

	ASVs	De novo	Closed-ref
Precise	✓	~	~
Tractable	✓	~	✓
Reproducible	✓	X	✓
Comprehensive	✓	✓	X

Which taxonomic database to choose?

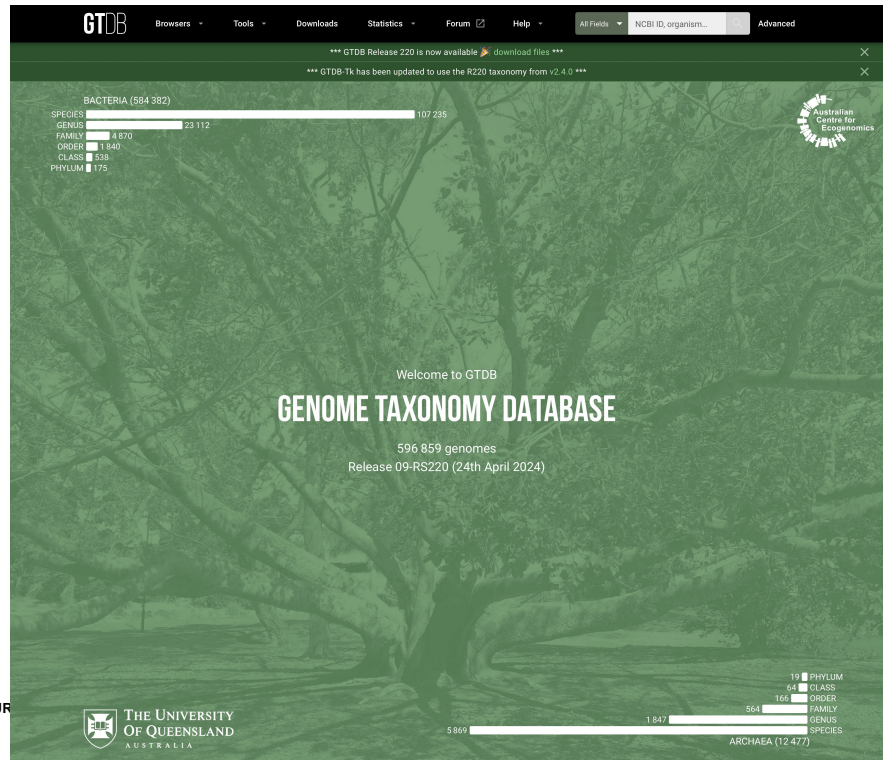
	GG2	SILVA	NCBI	RDP
URL	https://greengenes2.ucsd.edu	https://www.arb-silva.de	https://www.ncbi.nlm.nih.gov/refseq/targeted loci/	https://rdp.cme.msu.edu
Last release	2024.09	138.2 Jul. 2024	228 Feb. 2024	11.5 Sept. 2016
SSU sequences	23‘467‘470	2’224’690	27’319	3‘356‘809
LSU sequences	NA	227’318	11’219	125‘525

GreenGenes recently updated after 10 years dormancy!

RDP not updated anymore

New trend GTDB (classification to species level with whole genome) https://gtdb.ecogenomic.org/stats/r220

Complete genome database, not only 16S



What is a MOCK community?

A MOCK community is a synthetic sample containing known amounts of DNA or cells from known bacterias.
Example:

b|e|i RESOURCES

SUPPORTING INFECTIOUS DISEASE RESEARCH

Prod

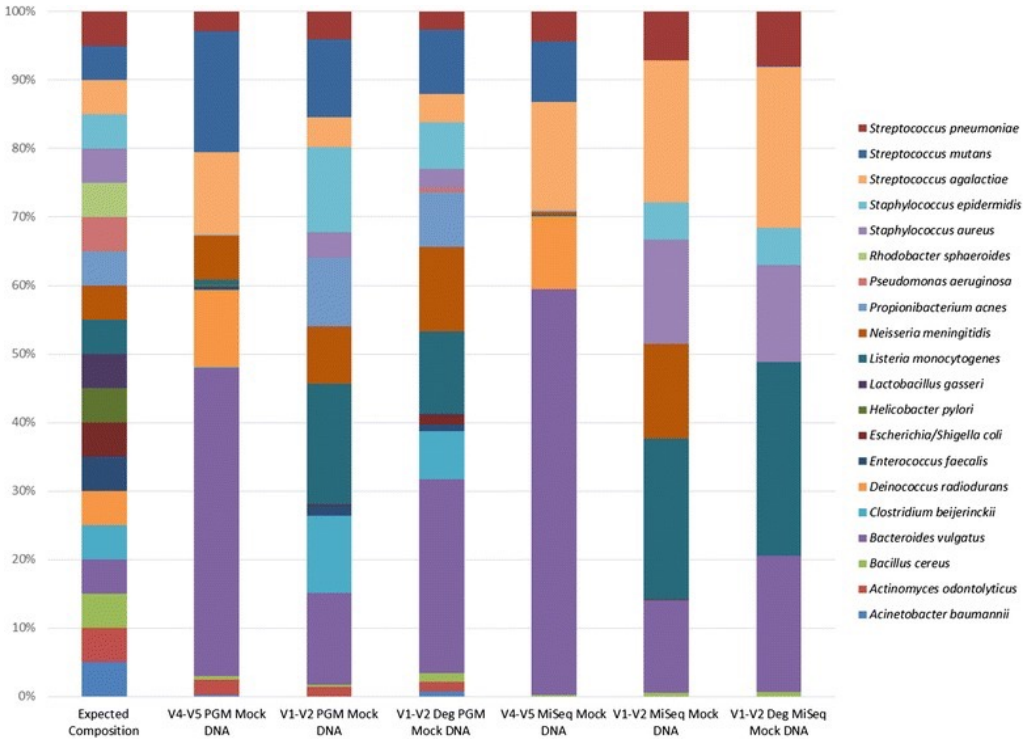
Genomic DNA from Microbial Mock Community B (Even, Low Concentration), v5.1L, for 16S rRNA Gene Sequencing

Catalog No. HM-782D

You can buy it and use it as internal standard for your metagenomics analysis.

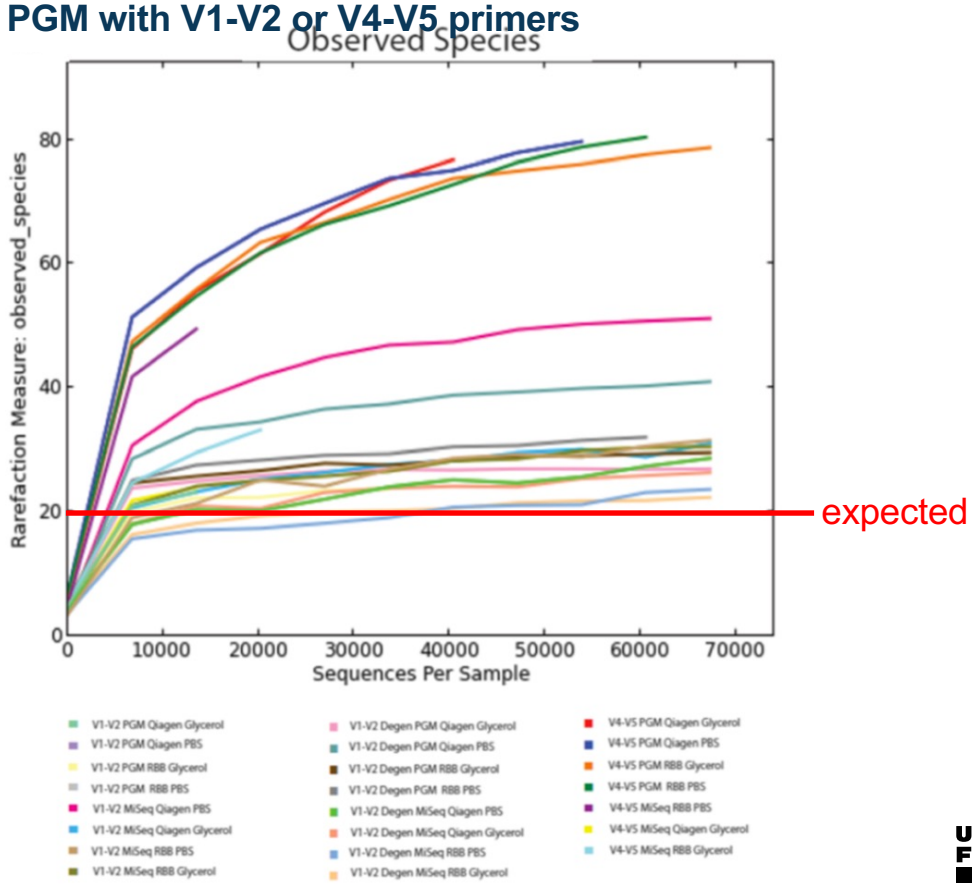
https://www.beiresources.org

Example MOCK with 20 bacterial strains in equal amounts sequenced by MiSeq and PGM with V1-V2 or V4-V5 primers



Fouhy et al. BMC Microbiology (2016) 16:123 DOI 10.1186/s12866-016-0738-z

Example MOCK with 20 bacterial strains in equal amounts sequenced by MiSeq and PGM with V1-V2 or V4-V5 primers



MOCK OTUs assigned to various databases

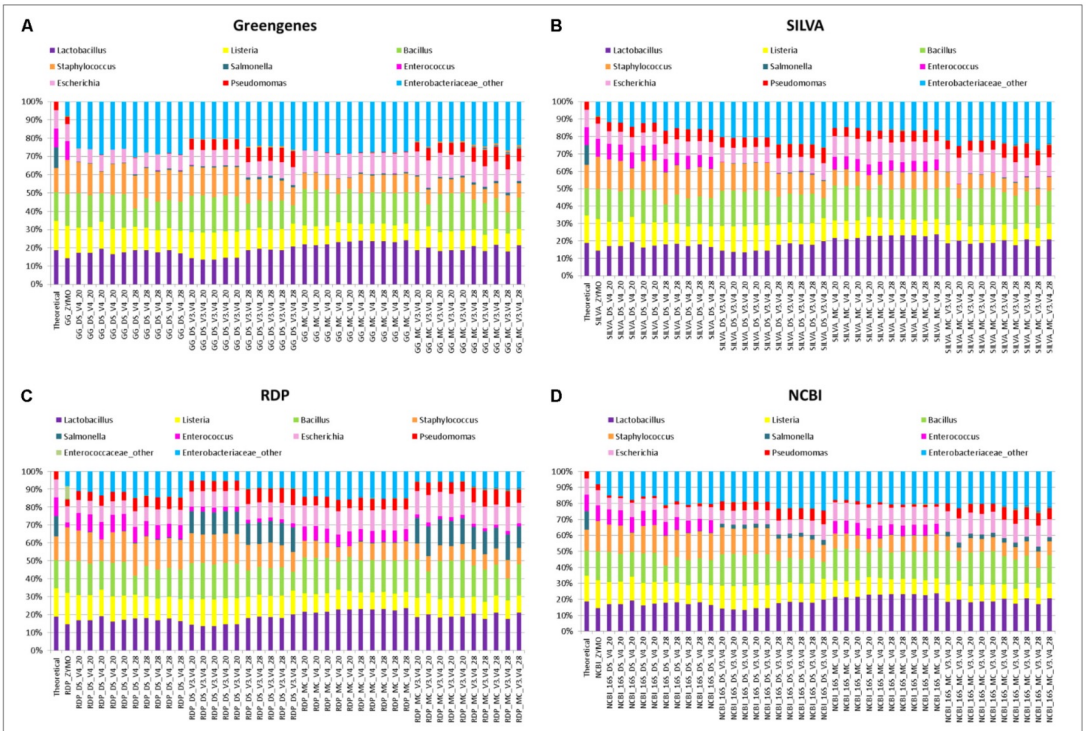


FIGURE 4 | Stack plot comparing the relative abundance of the theoretical mock community compositions (Theoretical), Zymo 16S rRNA sequencing data (Zymo), mock community cells extracted using the RBBC method (MC), and mock community DNA (DS) amplified using primers targeting the V3–V4 and V4 region of the 16S rRNA gene with 20 and 28 cycles. Sequences were clustered into OTUs at 97% identify and taxonomic classification of OTUs was conducted with the RDP classifier within QIIME analyses utilizing the **(A)** Greengenes, **(B)** SILVA, **(C)** RDP, and **(D)** NCBI 16S rRNA databases.

McGovern E, et al. (2018) Front. Microbiol. 9:1365. doi: 10.3389/fmicb.2018.01365

MOCK conclusion

Even with a controlled set of bacteria, none of the method is capable of identifying all the targets correctly.

Species level OTU always overestimate the number of bacteria, probably due to sequencing errors.

Still useful to include a MOCK sample to validate the analysis pipeline.

Specific reference databases emerging (e.g., human gut)

Human Gut Microbiome Blueprint

Combines 553 species of the Human gut reference database with 1,952 UMGS unclassified metagenomes (total 2505 species).
Almeida et al, 2019 https://doi.org/10.1038/s41586-019-0965-1

GutFeelingKB (https://hive.biochemistry.gwu.edu/gfkb)

853 species from healthy human gut
King et al, 2019 https://doi.org/10.1371/journal.pone.0206484

Culturable Genome Reference (CGR)

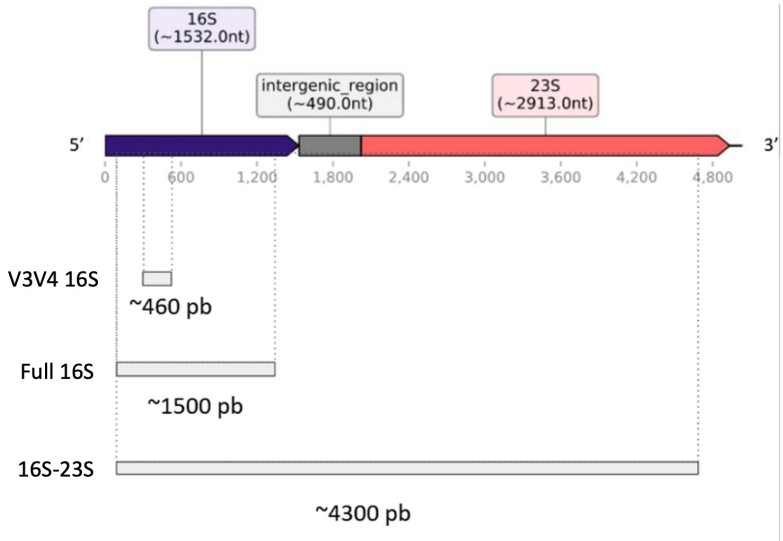
A collection of 1,520 nonredundant high-quality draft genomes
Zou et al. 2019 https://doi.org/10.1038/s41587-018-0008-8

Unified catalog of human gut reference genomes (UHGG & UHGP)

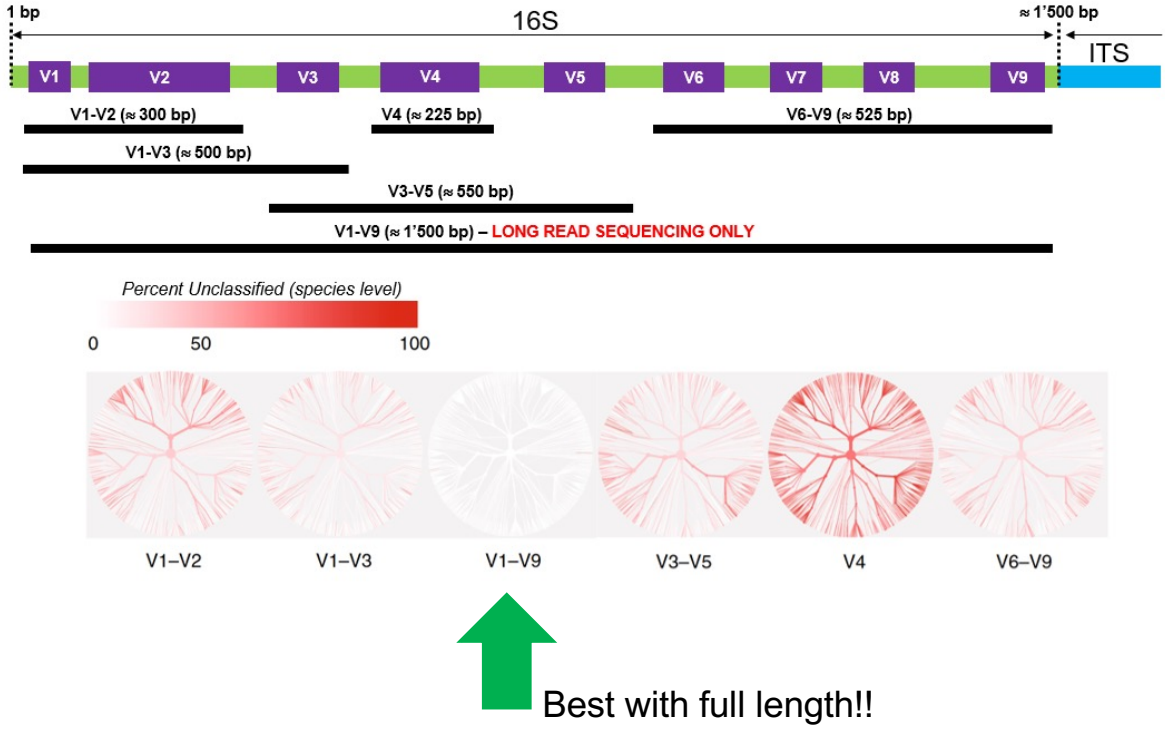
A unified catalog of 204,938 reference genomes from the human gut microbiome.
Almeida *et al.* 2021 https://doi.org/10.1038/s41587-020-0603-3

Full length amplicon sequencing

With the improvement of long read sequencing and the development of other long read techniques like LoopSeq, it becomes possible to sequence full length 16S or even larger amplicons

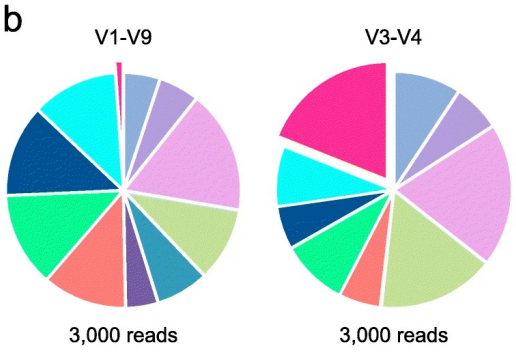


Improved classification with full length 16S

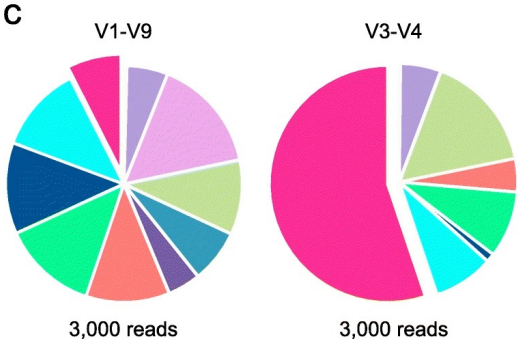


Best with full length!!

Example MOCK analysis with full length 16S *vs* V3-V4 (10 strains even mix)



Genus	V1-V9	V3-V4
Bacillus	5.07%	9.30%
Bifidobacterium	5.63%	6.60%
Clostridium	16.97%	19.70%
Deinococcus	10.23%	16.10%
Enterococcus	7.30%	0.13%
Escherichia	4.50%	0.00%
Lactobacillus	11.63%	5.77%
Rhodobacter	12.97%	9.17%
Staphylococcus	12.73%	5.97%
Streptococcus	11.87%	8.30%
mis- or unclassified	1.10%	18.97%

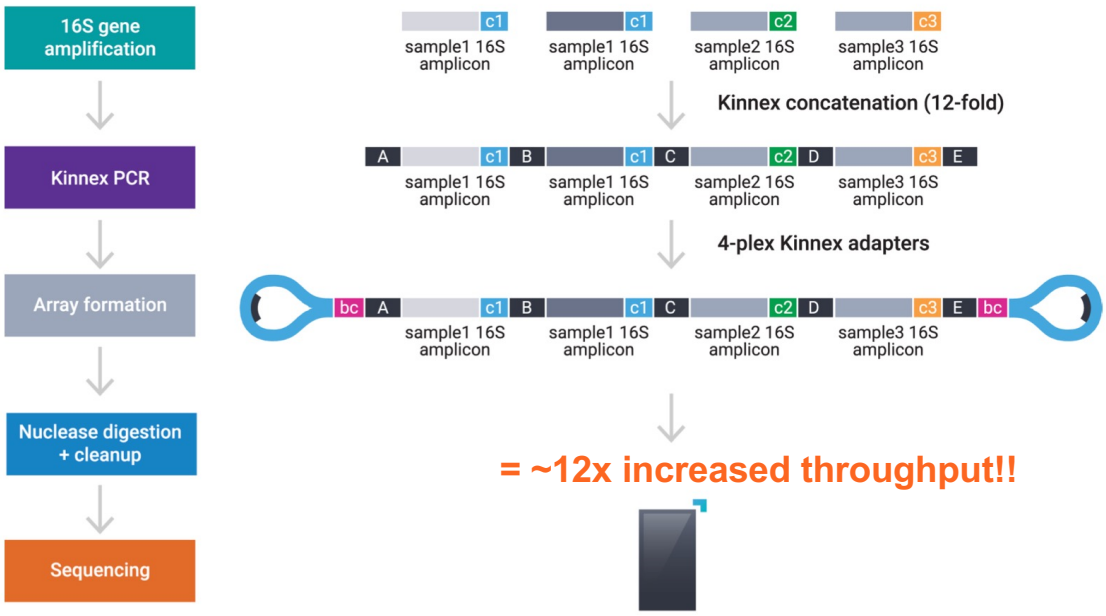


Species	V1-V9	V3-V4
B. cereus	0.40%	0.17%
B. adolescentis	5.63%	5.53%
C. beijerinckii	15.67%	0.03%
D. radiodurans	10.23%	16.07%
E. faecalis	7.30%	0.00%
Escherichia spp.	4.50%	0.00%
L. gasseri	11.47%	4.57%
R. sphaeroides	12.97%	9.07%
S. epidermidis	12.50%	1.20%
S. mutans	11.87%	8.20%
mis- or unclassified	7.47%	55.17%

Matsuo et al. BMC Microbiology (2021) 21:35 https://doi.org/10.1186/s12866-021-02094-5

from PacBio: Kinnex 16S rRNA library

Combining multiple full length 16S (up to 12) in one HiFi read!



Summary of the topic

Targeted or amplicon sequencing is the most used method for microbiome analysis

Choice of primers for the amplicon is critical

Choice of the reference database is critical

MOCK sample helps to validate the pipeline

OTUs are defined arbitrarily

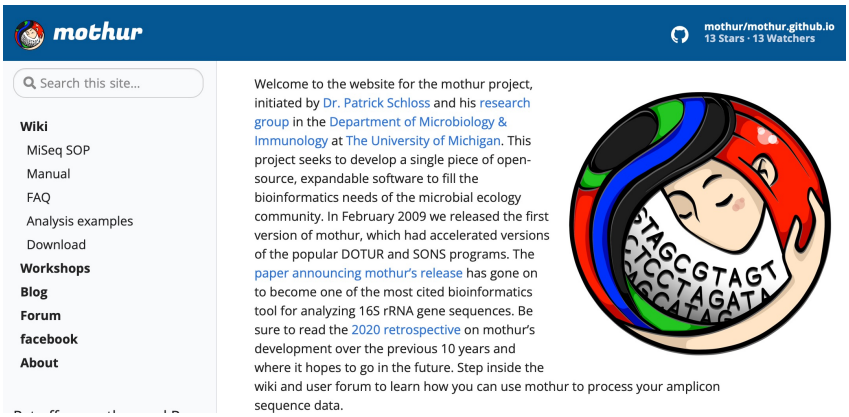
Reference catalogs are emerging

Full length 16S now available!

Some details of the pipelines

Mothur
QIIME2
DADA2

Mothur pipeline (UNIX)



Prepare input data
Make contig
Unique sequences
Align sequences
Screen sequences
Filter sequences
Pre-cluster
Remove chimeric sequences
Classify sequences based on reference DB (e.g., Silva)
Downstream analyses...

Based on MiSeq SOP (Standard Operating Procedure)
https://mothur.org/wiki/miseq_sop/

What is mothur?

A software package developed by Prof. Pat Schloss (University of Michigan) and a small community

Open source, single-piece executable for microbial ecology
https://www.mothur.org
https://github.com/mothur/mothur

Handles lots of different data types
Sanger
PacBio
IonTorrent
454
Illumina MiSeq/HiSeq

Mothur commands

See the manual on the web for commands.
https://mothur.org/wiki/Mothur_manual

Alternatively type help() to mothur prompt to list all commands.
Help on a particular command can be invoked by, e.g.,
summary.seqs(help).

Command are always associated with brackets, and the command arguments are given inside the brackets.
summary.seqs(fasta=all.fasta)

prepare input data, trim by combining paired-end sequences

stability file **# must be prepared by user (one can use the make.file command in Mothur)**

```
more ws.stability
bdg-ws-1a   bdg-ws-1a_S51_L001_R1_001.fastq      bdg-ws-1a_S51_L001_R2_001.fastq
bdg-ws-1b   bdg-ws-1b_S66_L001_R1_001.fastq      bdg-ws-1b_S66_L001_R2_001.fastq
bdg-ws-1c   bdg-ws-1c_S81_L001_R1_001.fastq      bdg-ws-1c_S81_L001_R2_001.fastq
...
```

> mothur **## start mothur environment or run a batch file**

make.contigs(file=ws.stability, processors=48) **# combines the PE reads into single contigs**

summary.seqs(fasta=ws.trim.contigs.fasta)

```
Using 48 processors.
                Start     End       NBases    Ambigs    Polymer   NumSeqs
Minimum:        1         35        35        0         2         1
2.5%-tile:      1         407       407       0         5         204776
25%-tile:       1         413       413       0         5         2047752
Median:         1         414       414       0         5         4095503
75%-tile:       1         415       415       2         5         6143254
97.5%-tile:     1         417       417       26        6         7986229
Maximum:        1         603       602       76        300       8191004
Mean:           1         414.841   414.84    2.96865   5.08294
# of Seqs:      8191004

Output File Names:
ws.trim.contigs.summary

It took 65 secs to summarize 8191004 sequences.
```


UNI FR

Chimeric sequences

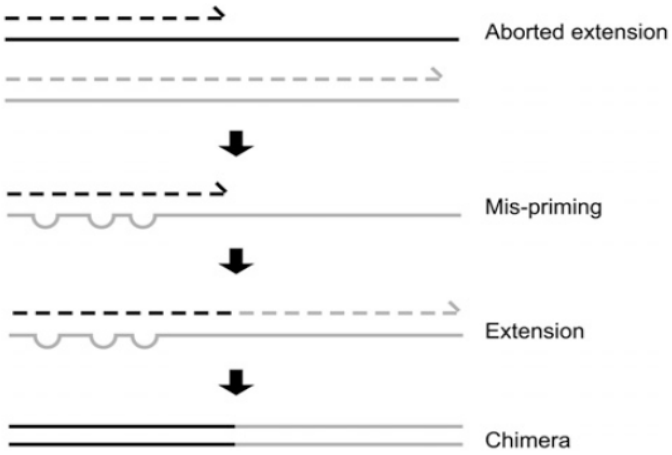


Figure 1. Formation of chimeric sequences during PCR. An aborted extension product from an earlier cycle of PCR can function as a primer in a subsequent PCR cycle. If this aborted extension product anneals to and primes DNA synthesis from an improper template, a chimeric molecule is formed.

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

UNI FR

clusterize, remove & summarize sequences

chimera.vsearch(fasta=ws.trim.contigs.good.unique.good.filter.unique.precluster.fasta,count=ws.trim.contigs.good.unique.good.filter.unique.precluster.count_table, dereplicate=t) **# identify chimeric sequences**

```
Output File Names:
ws.trim.contigs.good.unique.good.filter.unique.precluster.uchime.pick.count_table
ws.trim.contigs.good.unique.good.filter.unique.precluster.uchime.chimeras
ws.trim.contigs.good.unique.good.filter.unique.precluster.uchime.accnos
```

remove.seqs(fasta=ws.trim.contigs.good.unique.good.filter.vsearch.precluster.fasta, accnos=ws.trim.contigs.good.unique.good.filter.unique.precluster.vsearch.accnos) **# removes chimeric sequences**

```
Removed 59623 sequences from your fasta file.

Output File Names:
ws.trim.contigs.good.unique.good.filter.unique.precluster.pick.fasta
```

summary.seqs(fasta=current, count=current)

```
Using 48 processors.
                Start   End     NBases  Ambigs  Polymer NumSeqs
Minimum:        1       1060    386     0       4       1
2.5%-tile:      1       1060    401     0       5       122630
25%-tile:       1       1060    408     0       5       1226295
Median:         1       1060    408     0       5       2452589
75%-tile:       1       1060    409     0       5       3678883
97.5%-tile:     1       1060    411     0       5       4782548
Maximum:        2       1060    430     0       7       4905177
Mean:           1.00008 1060    408.065 0       5.00203
# of unique seqs:   81360
total # of seqs:    4905177

Output File Names:
ws.trim.contigs.good.unique.good.filter.unique.precluster.pick.summary
It took 3 secs to summarize 4905177 sequences.
```


Output from Mothur: feature tables

ws.an.cons.taxonomy

```
OTU             Size            Taxonomy
Otu00001        530053          Bacteria(100);Proteobacteria(100);Gammaproteobacteria(100);Pseudomonadales(100);Pseudomonadaceae(100);Pseudomonas(100);
Otu00002        298855          Bacteria(100);Proteobacteria(100);Betaproteobacteria(100);Methylophilales(100);Methylophilaceae(100);Methylophilus(90);
Otu00003        546152          Bacteria(100);Proteobacteria(100);Alphaproteobacteria(100);Rhizobiales(100);Rhizobiaceae(100);Rhizobium(100);
Otu00004        269985          Bacteria(100);Proteobacteria(100);Gammaproteobacteria(100);Pseudomonadales(100);Pseudomonadaceae(100);Pseudomonas(99);
Otu00005        206811          Bacteria(100);Proteobacteria(100);Betaproteobacteria(100);Burkholderiales(100);Burkholderiaceae(100);Burkholderia(100);
Otu00006        100666          Bacteria(100);Proteobacteria(100);Betaproteobacteria(100);Burkholderiales(100);Comamonadaceae(100);Variovorax(96);
Otu00007        54191           Bacteria(100);Proteobacteria(100);Gammaproteobacteria(100);Xanthomonadales(100);Xanthomonadaceae(100);Stenotrophomonas
...
```

ws.an.shared

```
label Group       numOtus   Otu00001  Otu00002  Otu00003  Otu00004  Otu00005  Otu00006      Otu00007    Otu00008 ...
0.03  bdg-ws-1a   11867     17387     27426     11854     207       53        5808          5964        2006 ...
0.03  bdg-ws-1b ...
...
```

To export results in other tools, convert to BIOM 2.1 HDF5 format:
make.biom(shared=ws.an.shared, constaxonomy=ws.an.cons.taxonomy)

To export results in other tools, convert to BIOM 1.0 JSON format:
make.biom(shared=ws.an.shared, constaxonomy=ws.an.cons.taxonomy, output=simple)

What is the BIOM format?

The BIOM file format (canonically pronounced biome) is designed to be a general-use format for representing biological sample by observation contingency tables. BIOM is a recognized standard for the Earth Microbiome Project and is a Genomics Standards Consortium supported project.

The idea is to combine the (OTU/ASV) abundance table and the metadata tables (sample, taxonomy, etc.) in a single structured file.

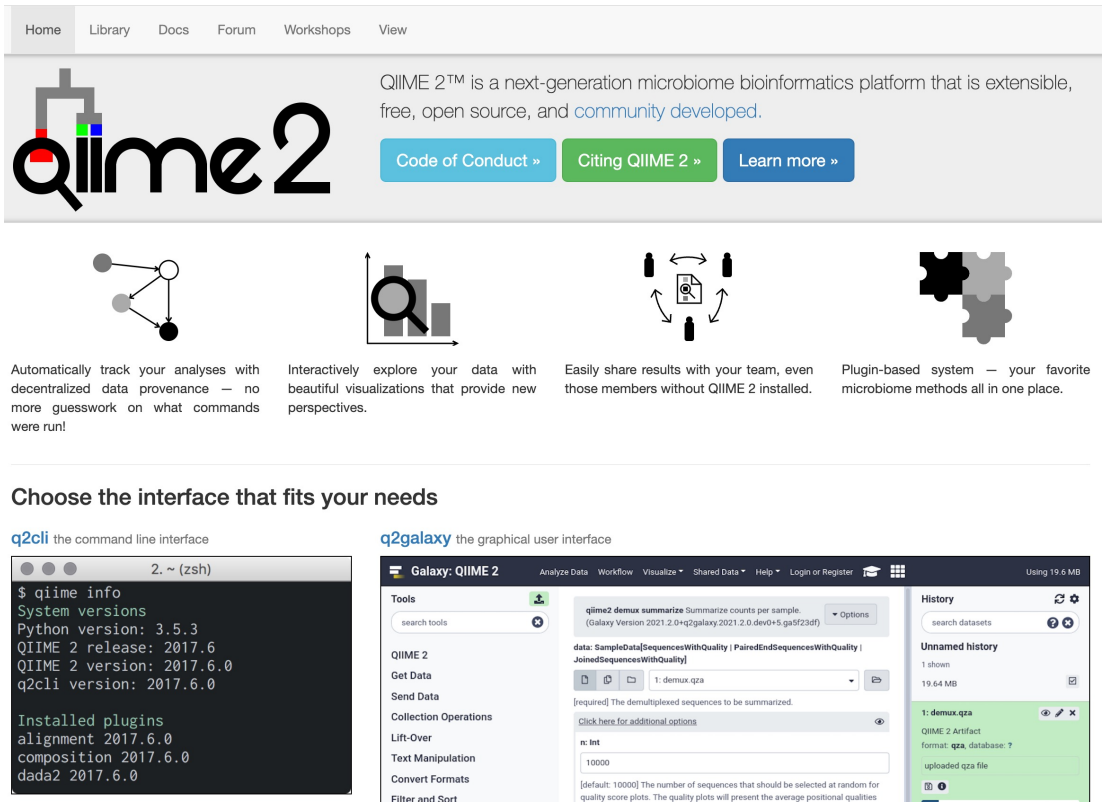
The BIOM format (http://biom-format.org)

The BIOM exist in 2 main versions:
v1 (JSON) and v2 (HDF5).

Some softwares accept only the v1, others accept only the v2, others accept every version of BIOM.

Software using BIOM format:
QIIME, MG-RAST, PICRUSt, Mothur, phyloseq, MEGAN, VAMPS, metagenomeSeq, Phinch, RDP Classifier, USEARCH, PhyloToAST, EBI Metagenomics, GCModeller, MetaPhlAn 2, MetagenomeAnalyst

What is QIIME2?



https://qiime2.org

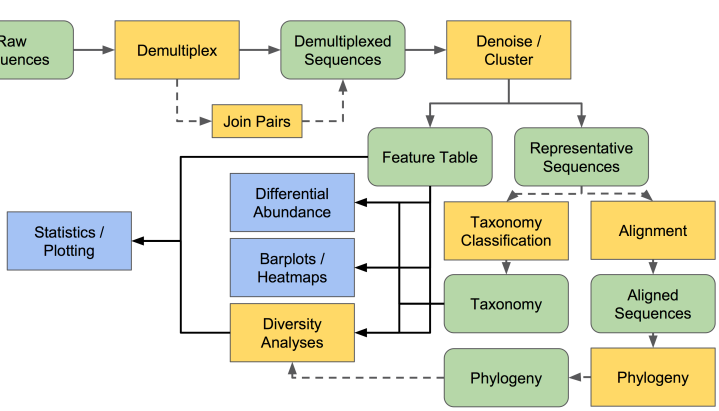
What is QIIME2?

QIIME2 uses specific file formats called « artifacts » and « visualizations »

Extensions: artifacts.qza and visu.qzv

Lots of tutorials are offered:
https://docs.qiime2.org/2024.10/tutorials/

Many flowcharts describe the possible pipelines



Comparison with Mothur for OTU doesn't reveal significant differences

Main differences are only observed when the reference databases are different.

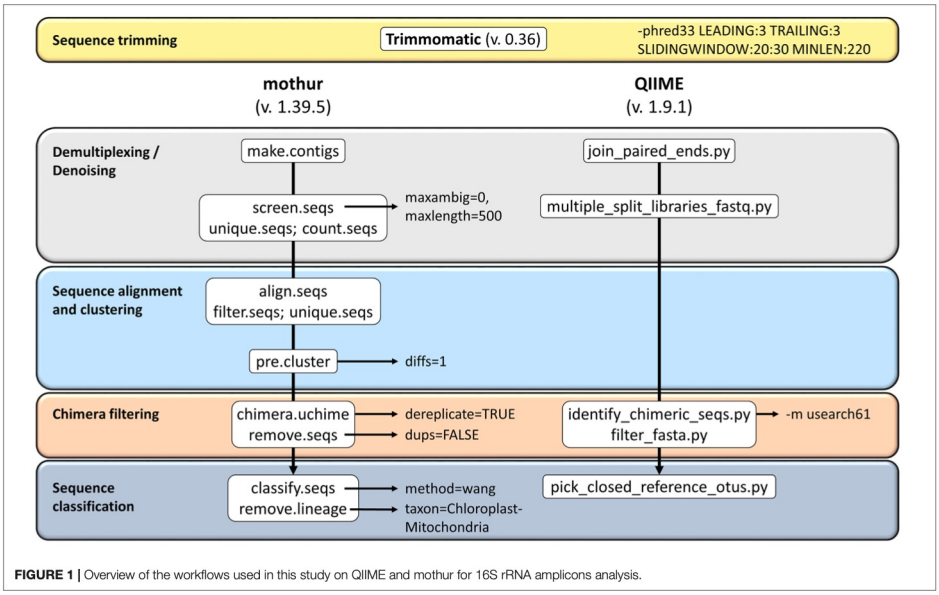


FIGURE 1 | Overview of the workflows used in this study on QIIME and mothur for 16S rRNA amplicons analysis.

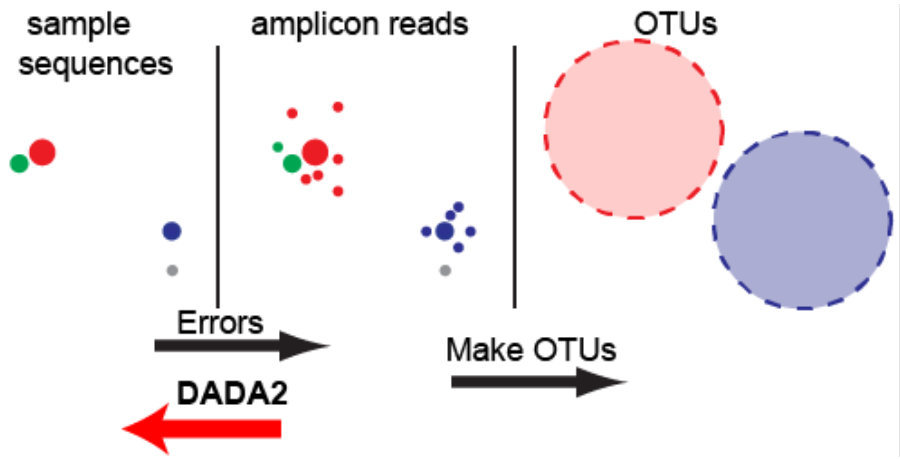
López-García et al, (2018) Comparison of Mothur and QIIME for the Analysis of Rumen Microbiota Composition Based on 16S rRNA Amplicon Sequences. Front. Microbiol. 9:3010. doi: 10.3389/fmicb.2018.03010

Denoising sequences

Amplicon Sequencing. **Exactly.** *Version 1.12*

Removing noise in 16S datasets is highly recommended

DADA2 corrects the reads according to the error levels and produces exact amplicon sequence variant (ASV) instead of operational taxonomic units (OTU).



Other denoising tools: UNOISE2, Deblur

DADA2 pipeline (in R or Qiime2)

Amplicon Sequencing. Exactly. Version 1.12

- Import reads
- QC
- Filter & Trim
- Learn error rate
- Sample inference (correct for errors)
- Merge paired reads
- Construct sequence table
- Remove chimeras
- Track reads
- Assign taxonomy

Load library and import list of reads

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

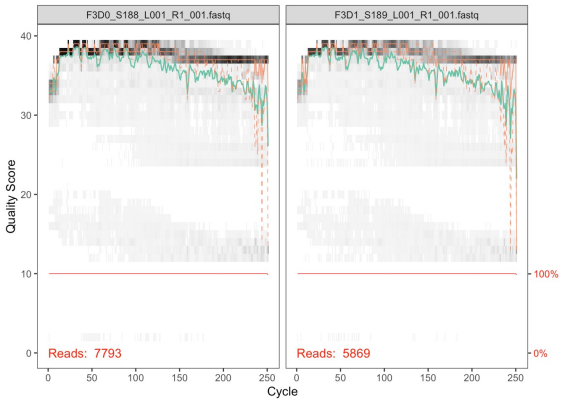
```
## [1] '1.12.1'
```

```
# Forward and reverse fastq filenames have format: SAMPLENAME_R1_001.fastq and SA
MPLENAME_R2_001.fastq
fnFs <- sort(list.files(path, pattern="_R1_001.fastq", full.names = TRUE))
fnRs <- sort(list.files(path, pattern="_R2_001.fastq", full.names = TRUE))
# Extract sample names, assuming filenames have format: SAMPLENAME_XXX.fastq
sample.names <- sapply(strsplit(basename(fnFs), "_"), `[`, 1)
```

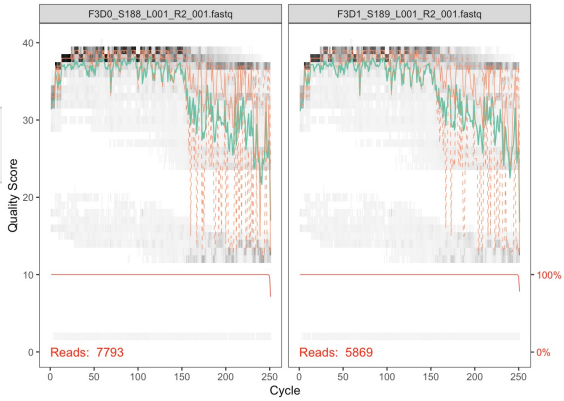
Considerations for your own data: The string manipulations may have to be modified if your filename format is different.

QC

```
plotQualityProfile(fnFs[1:2])
```




```
plotQualityProfile(fnRs[1:2])
```



Filter and trim

```
# Place filtered files in filtered/ subdirectory
filtFs <- file.path(path, "filtered", paste0(sample.names, "_F_filt.fastq.gz"))
filtRs <- file.path(path, "filtered", paste0(sample.names, "_R_filt.fastq.gz"))
names(filtFs) <- sample.names
names(filtRs) <- sample.names
```

```
out <- filterAndTrim(fnFs, filtFs, fnRs, filtRs, truncLen=c(240,160),
              maxN=0, maxEE=c(2,2), truncQ=2, rm.phix=TRUE,
              compress=TRUE, multithread=TRUE) # On Windows set multithread=FALSE
head(out)
```

```
##                               reads.in reads.out
## F3D0_S188_L001_R1_001.fastq       7793      7113
## F3D1_S189_L001_R1_001.fastq       5869      5299
## F3D141_S207_L001_R1_001.fastq     5958      5463
## F3D142_S208_L001_R1_001.fastq     3183      2914
## F3D143_S209_L001_R1_001.fastq     3178      2941
## F3D144_S210_L001_R1_001.fastq     4827      4312
```

Considerations for your own data: The standard filtering parameters are starting points, not set in stone. If you want to speed up downstream computation, consider tightening `maxEE`. If too few reads are passing the filter, consider relaxing `maxEE`, perhaps especially on the reverse reads (eg. `maxEE=c(2,5)`), and reducing the `truncLen` to remove low quality tails. Remember though, when choosing `truncLen` for

Learn error rate

```
errF <- learnErrors(filtFs, multithread=TRUE)
```

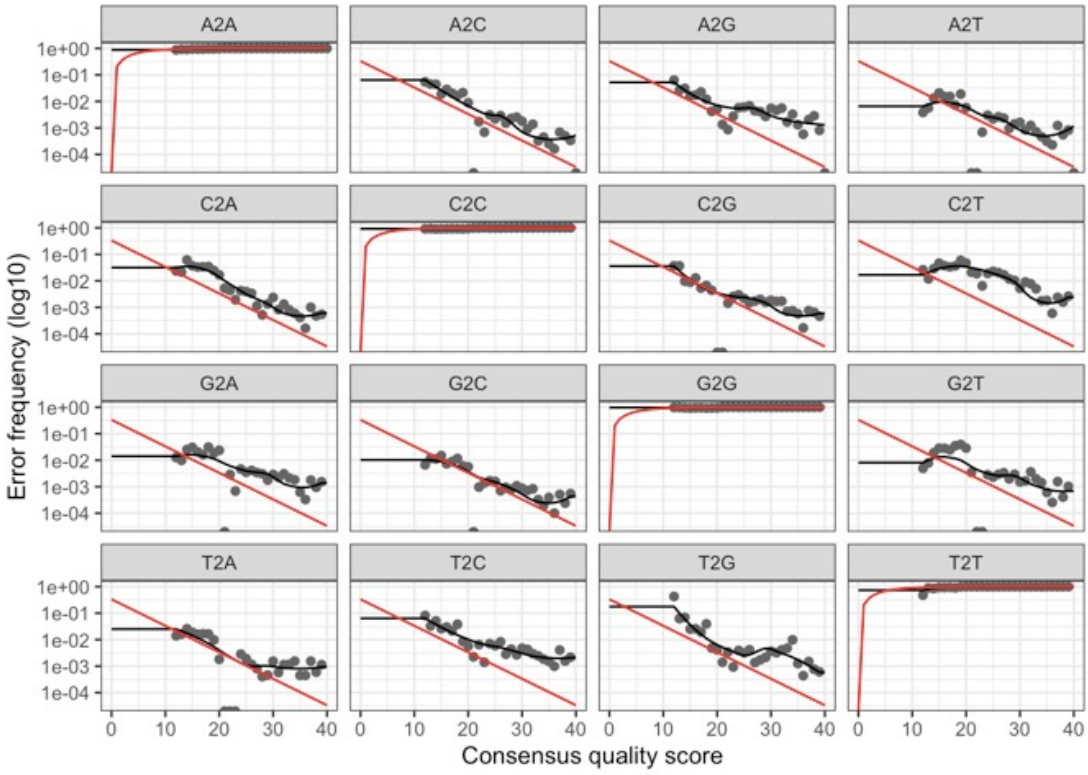
```
## 33514080 total bases in 139642 reads from 20 samples will be used for learning
the error rates.
```

```
errR <- learnErrors(filtRs, multithread=TRUE)
```

```
## 22342720 total bases in 139642 reads from 20 samples will be used for learning
the error rates.
```


Plot error rate

```
plotErrors(errF, nominalQ=TRUE)
```



Sample Inference (correct errors)

```
dadaFs <- dada(filtFs, err=errF, multithread=TRUE)
```

```
## Sample 1 - 7113 reads in 1979 unique sequences.
## Sample 2 - 5299 reads in 1639 unique sequences.
## Sample 3 - 5463 reads in 1477 unique sequences.
## Sample 4 - 2914 reads in 904 unique sequences.
```

```
dadaRs <- dada(filtRs, err=errR, multithread=TRUE)
```

```
## Sample 1 - 7113 reads in 1660 unique sequences.
## Sample 2 - 5299 reads in 1349 unique sequences.
## Sample 3 - 5463 reads in 1335 unique sequences.
## Sample 4 - 2914 reads in 853 unique sequences.
```

```
dadaFs[[1]]
```

```
## dada-class: object describing DADA2 denoising results
## 128 sequence variants were inferred from 1979 input unique sequences.
## Key parameters: OMEGA_A = 1e-40, OMEGA_C = 1e-40, BAND_SIZE = 16
```


Merge paired reads

```
mergers <- mergePairs(dadaFs, filtFs, dadaRs, filtRs, verbose=TRUE)
# Inspect the merger data.frame from the first sample
head(mergers[[1]])
```

```
##
sequence
## 1 TACGGAGGATGCGAGCGTTATCCGGATTTATTGGGTTTAAAGGGTGCGCAGGCGGAAGATCAAGTCAGCGGTAAAA
TTGAGAGGCTCAACCTCTTCGAGCCGTTGAAACTGGTTTTCTTGAGTGAGCGAGAAGTATGCGGAATGCGTGGTGTAGCGG
TGAAATGCATAGATATCACGCAGAACTCCGATTGCGAAGGCAGCATACCGGCGCTCAACTGACGCTCATGCACGAAAGTGT
GGGTATCGAACAGG
## 2 TACGGAGGATGCGAGCGTTATCCGGATTTATTGGGTTTAAAGGGTGCGTAGGCGGCCTGCCAAGTCAGCGGTAAAA
TTGCGGGGCTCAACCCCGTACAGCCGTTGAAACTGCCGGGCTCGAGTGGGCGAGAAGTATGCGGAATGCGTGGTGTAGCGG
TGAAATGCATAGATATCACGCAGAACCCCGATTGCGAAGGCAGCATACCGGCGCCCTACTGACGCTGAGGCACGAAAGTGC
GGGGATCAAACAGG
```

```
##   abundance forward reverse nmatch nmismatch nindel prefer accept
## 1       579       1       1    148          0      0      1   TRUE
## 2       470       2       2    148          0      0      2   TRUE
## 3       449       3       4    148          0      0      1   TRUE
## 4       430       4       3    148          0      0      2   TRUE
## 5       345       5       6    148          0      0      1   TRUE
## 6       282       6       5    148          0      0      2   TRUE
```

Construct sequence table

```
seqtab <- makeSequenceTable(mergers)
dim(seqtab)
```

```
## [1]  20 293
```

```
# Inspect distribution of sequence lengths
table(nchar(getSequences(seqtab)))
```

```
##
## 251 252 253 254 255
##   1  88 196   6   2
```

The sequence table is a `matrix` with rows corresponding to (and named by) the samples, and columns corresponding to (and named by) the sequence variants. This table contains 293 ASVs, and the lengths of our merged sequences all fall within the expected range for this V4 amplicon.

Considerations for your own data: Sequences that are much longer or shorter than expected may be the result of non-specific priming. You can remove non-target-length sequences from your sequence table (eg. `seqtab2 <- seqtab[,nchar(colnames(seqtab)) %in% 250:256]`). This is analogous to "cutting a band" in-silico to get amplicons of the targeted length.

Remove chimeras

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

```
## [1]  20 232
```

```
sum(seqtab.nochim)/sum(seqtab)
```

```
## [1] 0.964064
```

Track reads

```
getN <- function(x) sum(getUniques(x))
track <- cbind(out, sapply(dadaFs, getN), sapply(dadaRs, getN), sapply(mergers, g
etN), rowSums(seqtab.nochim))
# If processing a single sample, remove the sapply calls: e.g. replace sapply(dad
aFs, getN) with getN(dadaFs)
colnames(track) <- c("input", "filtered", "denoisedF", "denoisedR", "merged", "no
nchim")
rownames(track) <- sample.names
head(track)
```

```
##        input filtered denoisedF denoisedR merged nonchim
## F3D0    7793     7113      6976      6979   6540    6528
## F3D1    5869     5299      5227      5239   5028    5017
## F3D141  5958     5463      5331      5357   4986    4863
## F3D142  3183     2914      2799      2830   2595    2521
## F3D143  3178     2941      2822      2867   2552    2518
## F3D144  4827     4312      4151      4228   3627    3488
```


Assign taxonomy

```
taxa <- assignTaxonomy(seqtab.nochim, "~/tax/silva_nr_v132_train_set.fa.gz", multith
read=TRUE)
```

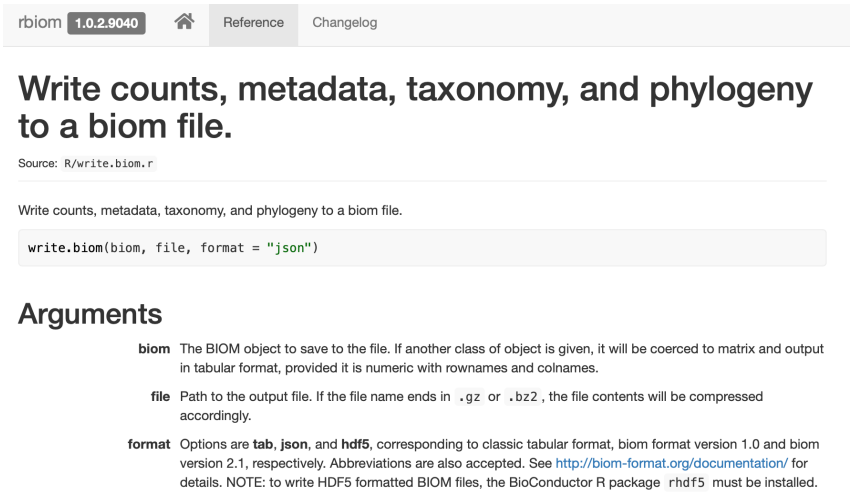
```
taxa.print <- taxa # Removing sequence rownames for display only
rownames(taxa.print) <- NULL
head(taxa.print)
```

```
##      Kingdom    Phylum          Class         Order
## [1,] "Bacteria" "Bacteroidetes" "Bacteroidia" "Bacteroidales"
## [2,] "Bacteria" "Bacteroidetes" "Bacteroidia" "Bacteroidales"
## [3,] "Bacteria" "Bacteroidetes" "Bacteroidia" "Bacteroidales"
## [4,] "Bacteria" "Bacteroidetes" "Bacteroidia" "Bacteroidales"
## [5,] "Bacteria" "Bacteroidetes" "Bacteroidia" "Bacteroidales"
## [6,] "Bacteria" "Bacteroidetes" "Bacteroidia" "Bacteroidales"
##      Family           Genus          Species
## [1,] "Muribaculaceae" NA             NA
## [2,] "Muribaculaceae" NA             NA
## [3,] "Muribaculaceae" NA             NA
## [4,] "Muribaculaceae" NA             NA
## [5,] "Bacteroidaceae" "Bacteroides" NA
## [6,] "Muribaculaceae" NA             NA
```

Export to biom from R (only BIOM version 1)

```
library(biomformat)
st.biom <- make_biom(t(seqtab.nochim), observation_metadata = taxa)
write_biom(st.biom, "path/to/my.biom")
```

Other recent package: library(rbiom) is capable to save into multiple formats



rbiom 1.0.2.9040 Reference Changelog

Write counts, metadata, taxonomy, and phylogeny to a biom file.

Source: `R/write.biom.r`

Write counts, metadata, taxonomy, and phylogeny to a biom file.

```
write.biom(biom, file, format = "json")
```

Arguments

biom The BIOM object to save to the file. If another class of object is given, it will be coerced to matrix and output in tabular format, provided it is numeric with rownames and colnames.

file Path to the output file. If the file name ends in `.gz` or `.bz2`, the file contents will be compressed accordingly.

format Options are **tab**, **json**, and **hdf5**, corresponding to classic tabular format, biom format version 1.0 and biom version 2.1, respectively. Abbreviations are also accepted. See http://biom-format.org/documentation/ for details. NOTE: to write HDF5 formatted BIOM files, the BioConductor R package `rhdf5` must be installed.

Full length 16S with DADA2

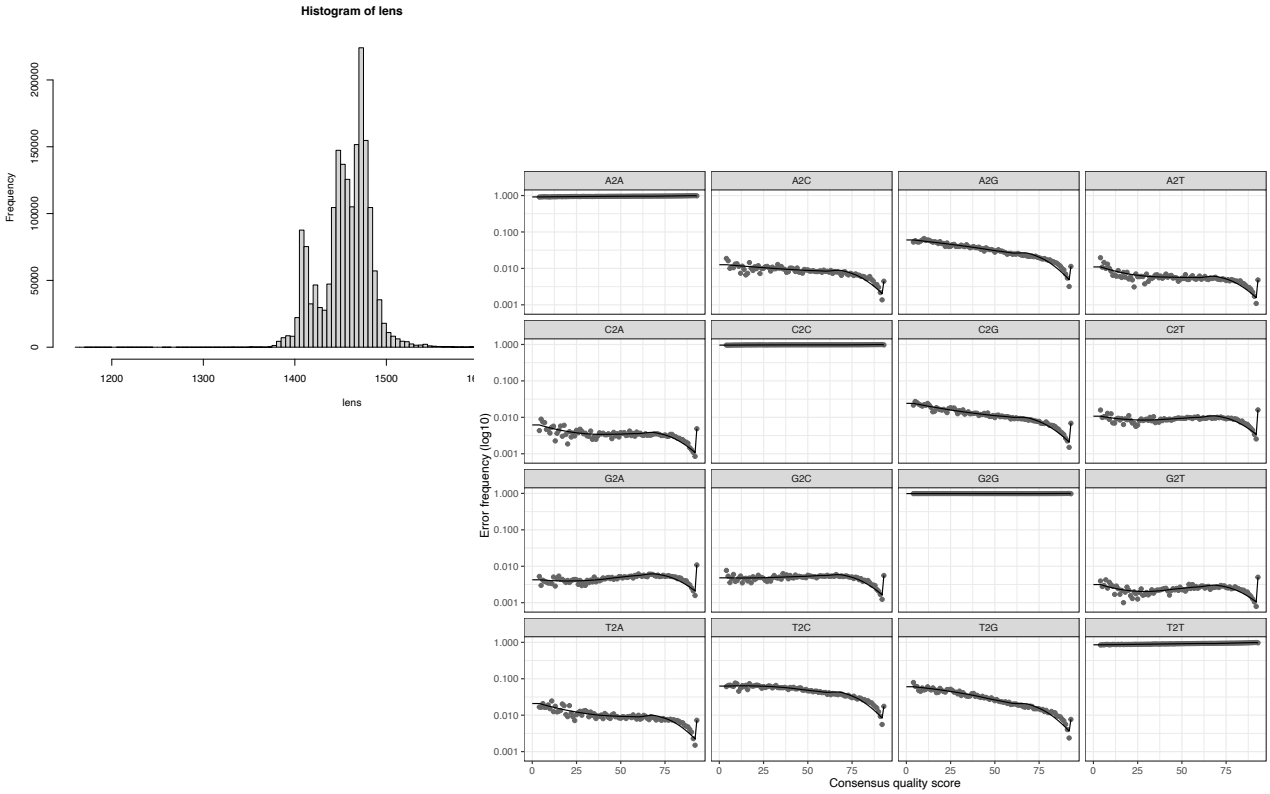
A small modification of DADA2 pipeline allows for analysis of full-length 16S dataset (e.g. ,from PacBio).

```
#filter and trim the reads and save clean reads
filts2 <- file.path(path2, "noprimers", "filtered", basename(fns2))
track2 <- filterAndTrim(nops2, filts2, minQ=3, minLen=1000,
maxLen=1600, maxN=0, rm.phix=FALSE, maxEE=2)

#learn errors
err2 <- learnErrors(drp2, errorEstimationFunction=PacBioErrfun,
BAND_SIZE=32, multithread=TRUE)

#denoise according to error learned
dd2 <- dada(drp2, err=err2, BAND_SIZE=32, multithread=TRUE)
```

Full length 16S with DADA2



Summary

Multiple pipelines exists (DADA2, Mothur, Qiime2, PIPITS3, Pipecraft2, NextITS etc.)

ASV is the new OTU (also for ITS)

Overlapping short reads pairs (read merging) helps

Chimera detection is an important cleaning step

Full length 16S analysis is possible with DADA2

ITS requires a different pipeline, long reads would be ideal

Thank you for your attention. Questions?