

# Introduction to amplicon sequence variant (ASV) analysis

# Outline

- Microbial community analysis using amplicon sequence variants (ASVs)
  - What are ASVs
  - What are the differences if compared to OTUs
- DADA2 workflow

# Amplicon sequence variants

- Biological sequences which can differ by just 1 nucleotide
- Provide increased resolution and sensitivity if compared to OTUs
  - OTU approach clusters sequences together, typically at 97% similarity threshold, in order to remove sequencing errors
  - ASV approach uses abundance and error model in order to remove sequencing errors, so clustering is not needed
- Generated without a reference → no reference bias
- Exact sequences, so ASV tables can be compared across studies
- Chimera detection is simpler than with OTUs
- A given target gene should always generate the same ASV
  - Thus, ASVs can be added to reference databases and merged to other datasets

# OTU vs ASVs

Schematic of OTU and DADA2 approaches towards amplicon sequencing errors.

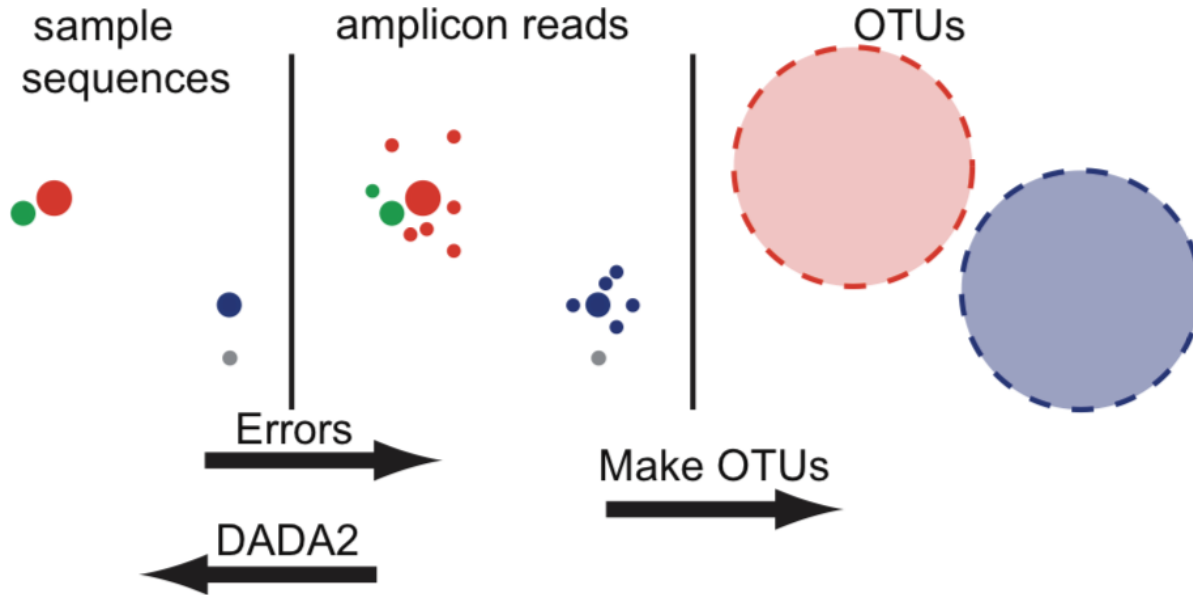


Figure describing the difference between *Amplicon Sequence Variants (ASVs)* vs *Operational Taxonomic Units (OTUs)* (figure adapted from (Callahan et al. 2016. DADA2: High-Resolution Sample Inference from Illumina Amplicon Data)).

# DADA2

- DADA2 = Divisive Amplicon Denoising Algorithm
  - R based library
  - Callahan, B. *et al.* 2016. DADA2: High-resolution sample inference from Illumina amplicon data. <https://doi.org/10.1038/nmeth.3869>
- Uses the abundance data and the error model to remove sequencing errors and to detect ASVs
- Balance between sensitivity and specificity

## DADA2 in Chipster

- The DADA2 analysis tools in Chipster cover the DADA2 tutorial workflow
  - <https://benjjneb.github.io/dada2/tutorial.html>
- Analysis examples use the MiSeq 16S data ([https://mothur.org/wiki/miseq\\_sop/](https://mothur.org/wiki/miseq_sop/))
- Analysis of ITS data and data produced with IonTorrent are possible as well
  - Key differences in the workflow will be covered
- Requirements
  - The samples have to be demultiplexed: Split to per-sample FASTQ files

# Overview of the ASV preprocessing pipeline in Chipster



1. Make a Tar package
2. Quality control with MultiQC
3. (Remove primers/adapters with Cutadapt)
4. Filter and trim reads
5. Sample/ASV inference
6. (Combine paired reads to contigs)
7. Make an ASV table and remove chimeras
8. Assign taxonomy
9. Make a phyloseq object
10. Merge phenodata to the phyloseq object
11. Continue community analysis with the same tools than after the OTU based workflow



# Filter and trim reads with **DADA2**



# Outline

- Why read filtering and trimming is important
- Things to take into account
- What trimming and filtering options are available
- How to set the parameters in Chipster
- What are the output files

## Filter and trim reads with DADA2

- Critical step in ASV workflow
  - Ns need to be removed
  - Remove low quality ends and too short reads
  - Cleaned data runs more efficiently and gives more accurate results
  - Reads matching the phiX genome are removed
- Based on the DADA2 tool `filterAndTrim()`
- Optimal parameter values depend on your data
- If paired end reads, both reads need to pass the filter in order to be kept
- Input file: Tar package of FASTQ files
  - You can make it with the tool “Make a tar package”
  - FASTQ files can be compressed

# Use MultiQC report to decide on trimming parameters

- Quality scores tend to decrease towards the end of the reads
- The quality of reverse reads is often worse in Illumina sequencing

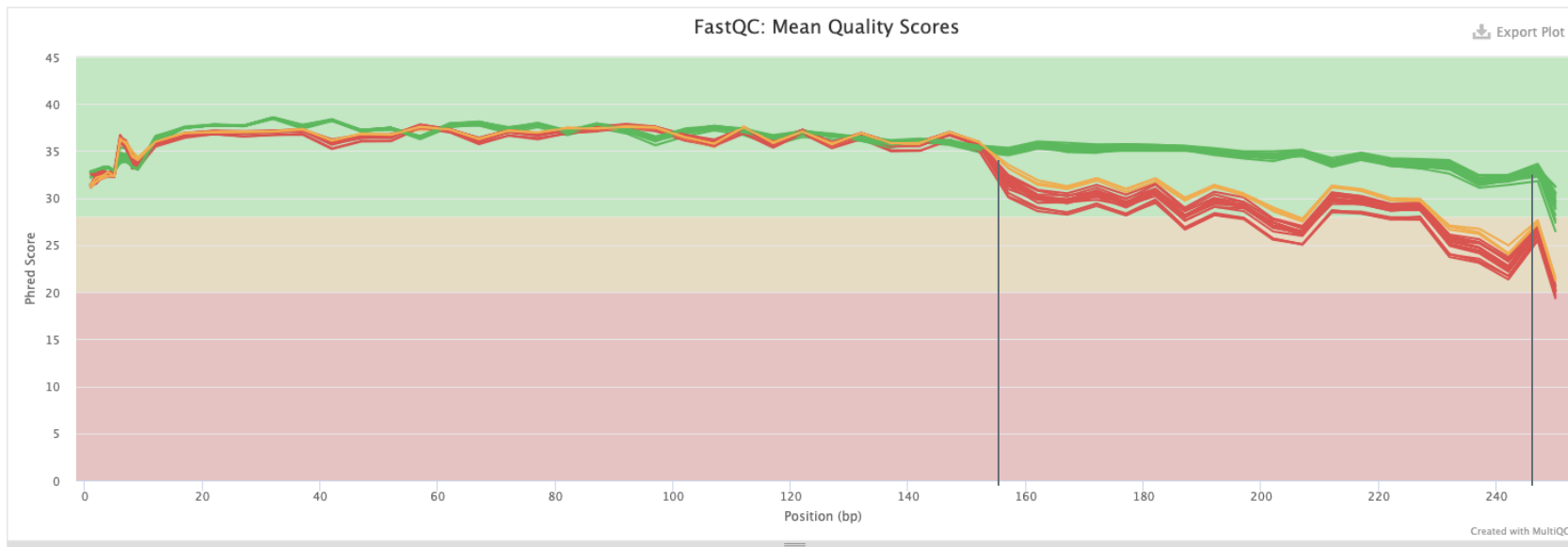
## Sequence Quality Histograms

19 3 16

The mean quality value across each base position in the read.

Help

Y-Limits: on



# Trim based on position or quality of a base

## Truncate forward reads after this amount of bases

Default 0 means no truncation. Truncate reads after truncLen bases. Reads shorter than this are discarded. You can use this parameter for single and paired end reads.



## Truncate reverse reads after this amount of bases

Default 0 means no truncation. Truncate reads after truncLen bases. Reads shorter than this are discarded. Use only for paired end reads.



## Truncate reads after this base quality

Truncate reads at the first instance of a quality score less than or equal to the specified number. Setting this parameter to 0, turns this behaviour off.

## The number of nucleotides to remove from start of each read

The number of nucleotides to remove from the start of each read. If both truncLen and trimLeft are provided, filtered reads will have length truncLen - trimLeft.

## Remove reads which are shorter than this

Removes reads which are shorter than the specified value. Min length is enforced after all other trimming and truncation. This parameter is especially usefull when truncLen parameter is not used for example with ITS data.

- You can define values for reverse and forward reads independently
- NOTE: paired reads need to overlap so that they can be combined into contigs later
- If read length or the length of the amplified (ITS) region varies, use the parameter “Remove reads which are shorter than this” instead.

# Filter based on the number of Ns and expected errors

- Ns need to be removed before the next analysis steps
- Maximum number of expected errors allowed for a read
  - Expected errors are calculated as the sum of error probabilities,  $EE = \sum 10^{-Q/10}$
  - More info <https://doi.org/10.1093/bioinformatics/btv401>
  - Default is 2

**Discard input sequences with more than specified number of Ns**

Sequences with more than the specified number of Ns will be discarded.  
Note that the dada function does not allow any Ns.

**Discard forward sequences with more than the specified number of expected errors**

After truncation, reads with more than this amount of expected errors will be discarded. If this parameter is not set, no expected error filtering is done. You can use this parameter for single and paired end reads.

**Discard reverse sequences with more than the specified number of expected errors**

After truncation, reads with more than this amount of expected errors will be discarded. If this parameter is not set, no expected error filtering is done. Use only for paired end reads.

# Output files

- Filtered.fastqs.tar
- Summary.tsv
- Samples.fastqs.txt

## Samples.fastqs.txt

F3D0	F3D0_S188_L001_R1_001.fastq	F3D0_S188_L001_R2_001.fastq
F3D141	F3D141_S207_L001_R1_001.fastq	F3D141_S207_L001_R2_001.fastq
F3D142	F3D142_S208_L001_R1_001.fastq	F3D142_S208_L001_R2_001.fastq
F3D143	F3D143_S209_L001_R1_001.fastq	F3D143_S209_L001_R2_001.fastq
F3D144	F3D144_S210_L001_R1_001.fastq	F3D144_S210_L001_R2_001.fastq
F3D145	F3D145_S211_L001_R1_001.fastq	F3D145_S211_L001_R2_001.fastq
F3D146	F3D146_S212_L001_R1_001.fastq	F3D146_S212_L001_R2_001.fastq
F3D147	F3D147_S213_L001_R1_001.fastq	F3D147_S213_L001_R2_001.fastq
F3D148	F3D148_S214_L001_R1_001.fastq	F3D148_S214_L001_R2_001.fastq
F3D149	F3D149_S215_L001_R1_001.fastq	F3D149_S215_L001_R2_001.fastq
F3D150	F3D150_S216_L001_R1_001.fastq	F3D150_S216_L001_R2_001.fastq

## Summary.tsv

Showing all 19 rows.

	reads.in	reads.out
F3D0	7793	7135
F3D141	5958	5477
F3D142	3183	2928
F3D143	3178	2959
F3D144	4827	4342
F3D145	7377	6787
F3D146	5021	4580
F3D147	17070	15719
F3D148	12405	11452
F3D149	13083	12059
F3D150	5509	5046
F3D1	5869	5316
F3D2	19620	18137
F3D3	6758	6282
F3D5	4448	4069
F3D6	7989	7394
F3D7	5129	4777
F3D8	5294	4885
F3D9	7070	6520



# Trim primers and adapters with Cutadapt

# Outline

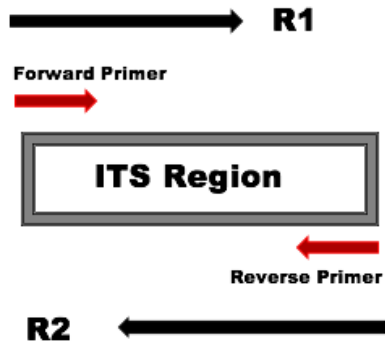
- What are adapter sequences
- How to identify adapters and check the correct orientation
- What are the adapter trimming options in Chipster
- How to use Cutadapt to trim adapters
- Things to take into account
- What are the resulting output files



# Adapter / primer sequences

- Adapter sequences need to be removed prior to analysis
- Removal is more complicated if some reads extend into the opposite primer
  - Can occur if the amplified region is shorter than the read length
  - If paired end reads, each read may or may not have the forward and reverse primer

**a.**



**b.**

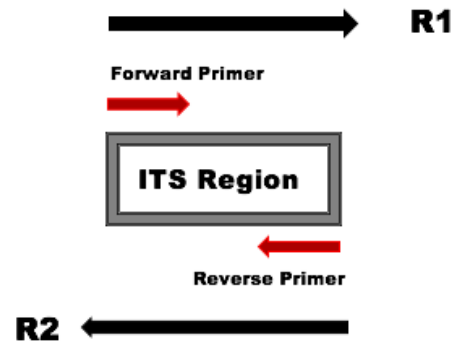


Image from DADA2 tutorial,  
[https://benjjneb.github.io/dada2/ITS\\_workflow.html](https://benjjneb.github.io/dada2/ITS_workflow.html)

# Tools to remove and identify adapters in Chipster

- Cutadapt
- Trimmomatic
  - Can be used to trim adapters from paired end and single end reads
  - Has many quality trimming and filtering options
- Identify primers and the correct orientation
  - Can be used to check if reads contain adapter sequences and in which orientation
- Predict primers/adapters with TagCleaner
  - Can be used to find adapter sequences if those are not known.
- Many trimming tools have an option to trim a fixed length of bases from the start or end of the reads

## Identify primers and the correct orientation

- Checks if given 5' and 3' primer/adaptor sequences are present in the reads and in which orientation
  - Run before and after a adaptor trimming tool to see if the adaptors were removed
- Remove ambiguous nucleotides first with the tool "Filter and trim reads with DADA2"
- Test every orientation of the 3' and 5' adaptors
  - Forward – Complement – Reverse – Reverse complement
- Input
  - FASTQ files of one sample
  - Tar package of FASTQ files

# Output files

## Primer\_summary.tsv

Showing all 4 rows.

	Forward	Complement	Reverse	RevComp
Forward_reads_5adapter	4214	0	0	0
Forward_reads_3adapter	0	0	0	3590
Reverse_reads_5adapter	0	0	0	3743
Reverse_reads_3adapter	4200	0	0	0

## Orientations\_summary.txt

Check all orientations of the given primers/adapters.

Reads are paired end.

The forward file used to check the adapters: SRR5314314\_F\_filt.fastq.gz

The reverse file used to check the adapters: SRR5314314\_R\_filt.fastq.gz

All orientations of the 5' adapter:

Forward: ACCTGCGGARGGATCA

Complement: TGGACGCCTYCCTAGT

Reverse: ACTAGGRAGGCGTCCA

RevComp: TGATCCYTCCGCAGGT

All orientations of the 3' adapter:

Forward: GAGATCCRTTGYTRAAAGTT

Complement: CTCTAGGYAACRAYTTTCAA

Reverse: TTGAAARTYGTTRCCTAGAG

RevComp: AACTTTYARCAAYGGATCTC

# Trim adapters with Cutadapt

- Based on Cutadapt
  - MARTIN, Marcel. 2011. Cutadapt removes adapter sequences from high-throughput sequencing reads. <https://doi.org/10.14806/ej.17.1.200>.
- Input
  - Tar package of FASTQ files
- Need to specify whether the reads are paired or single end
- Give the 5' and 3' adapters in the orientation they are found in the forward sequence
  - Use the tool "Identify primers and the correct orientation" to find and copy the right orientation
- 3' adapters and any sequence following it is removed
- 5' adapter and any sequence preceding it is removed
- Be carefull not to trim adapters from wrong ends!

# Example

Showing all 4 rows.

	Forward	Complement	Reverse	RevComp
Forward_reads_5adapter	4214	0	0	0
Forward_reads_3adapter	0	0	0	3590
Reverse_reads_5adapter	0	0	0	3743
Reverse_reads_3adapter	4200	0	0	0

1)

All orientations of the 5' adapter:

Forward: ACCTGCGGARGGATCA

Complement: TGGACGCCTYCCTAGT

Reverse: ACTAGGRAGGCGTCCA

RevComp: TGATCCYTCCGCAGGT

3)

## Parameters

 Reset All

Is the data paired end or single end reads

paired 

Are all the reads paired end, so one forward and one reverse FASTQ file for one sample.

The 5' adapter:

ACCTGCGGARGGATCA 

Give here the 5 end adapter/primer

The 3' adapter:

AACTTTYARCAAYGGATCTC 

Give here the 3 end adapter/primer

All orientations of the 3' adapter:

Forward: GAGATCCRTTGYTRAAAGTT

Complement: CTCTAGGYAACRAYTTTCAA

Reverse: TTGAAARTYGTTRCCTAGAG

RevComp: AACTTTYARCAAYGGATCTC

# Output files



- Adapters\_removed.tar
- Report.txt
- Samples.fastqs.txt

## Samples.fastqs.txt

SRR5314314	SRR5314314_1.fastq.gz	SRR5314314_2.fastq.gz
SRR5314315	SRR5314315_1.fastq.gz	SRR5314315_2.fastq.gz
SRR5314316	SRR5314316_1.fastq.gz	SRR5314316_2.fastq.gz
SRR5314317	SRR5314317_1.fastq.gz	SRR5314317_2.fastq.gz
SRR5314331	SRR5314331_1.fastq.gz	SRR5314331_2.fastq.gz
SRR5314332	SRR5314332_1.fastq.gz	SRR5314332_2.fastq.gz
SRR5314333	SRR5314333_1.fastq.gz	SRR5314333_2.fastq.gz
SRR5314334	SRR5314334_1.fastq.gz	SRR5314334_2.fastq.gz
SRR5314335	SRR5314335_1.fastq.gz	SRR5314335_2.fastq.gz

# Report.txt

- Big file, contains a report for every sample
- Contains all the relevant information how the adapters were removed
- Always check where the adapters were found and how many bases were removed

```
This is cutadapt 4.1 with Python 3.8.11
Command line parameters: -g ACCTGGGARGGATCA -a AACTTTYARCAAYGGATCTC -G GAGATCCRTTGYTRAAAGTT -A TGATCCYTCGCGAGGT -n 2 -j 2 -o out
Processing paired-end reads on 2 cores ...
Finished in 2.77 s (44 µs/read; 1.37 M reads/minute).
```

=== Summary ===

```
Total read pairs processed:      63,398
  Read 1 with adapter:          47,992 (75.7%)
  Read 2 with adapter:          48,032 (75.8%)
Pairs written (passing filters): 63,398 (100.0%)
```

```
Total basepairs processed: 31,737,011 bp
  Read 1: 15,867,972 bp
  Read 2: 15,869,039 bp
Total written (filtered):      22,831,611 bp (71.9%)
  Read 1: 11,374,379 bp
  Read 2: 11,457,232 bp
```

=== First read: Adapter 1 ===

Sequence: ACCTGGGARGGATCA; Type: regular 5'; Length: 16; Trimmed: 46986 times

Minimum overlap: 3  
No. of allowed errors:  
1-9 bp: 0; 10-16 bp: 1

Overview of removed sequences

length	count	expect	max.err	error counts
3	4	990.6	0	4
58	2	0.0	1	1 1
59	3	0.0	1	1 2
60	6	0.0	1	3 3
61	10	0.0	1	7 3
62	15	0.0	1	9 6
63	51	0.0	1	18 33
64	550	0.0	1	224 326
65	46191	0.0	1	18648 27543
66	143	0.0	1	60 83
67	4	0.0	1	1 3
68	3	0.0	1	0 3
70	1	0.0	1	0 1
71	1	0.0	1	0 1
73	1	0.0	1	0 1
94	1	0.0	1	1

=== First read: Adapter 2 ===

Sequence: AACTTTYARCAAYGGATCTC; Type: regular 3'; Length: 20; Trimmed: 45969 times

Minimum overlap: 3  
No. of allowed errors:  
1-9 bp: 0; 10-19 bp: 1; 20 bp: 2



## Run “Identify primers and the correct orientation” again

- That looks great, we managed to remove the adapters

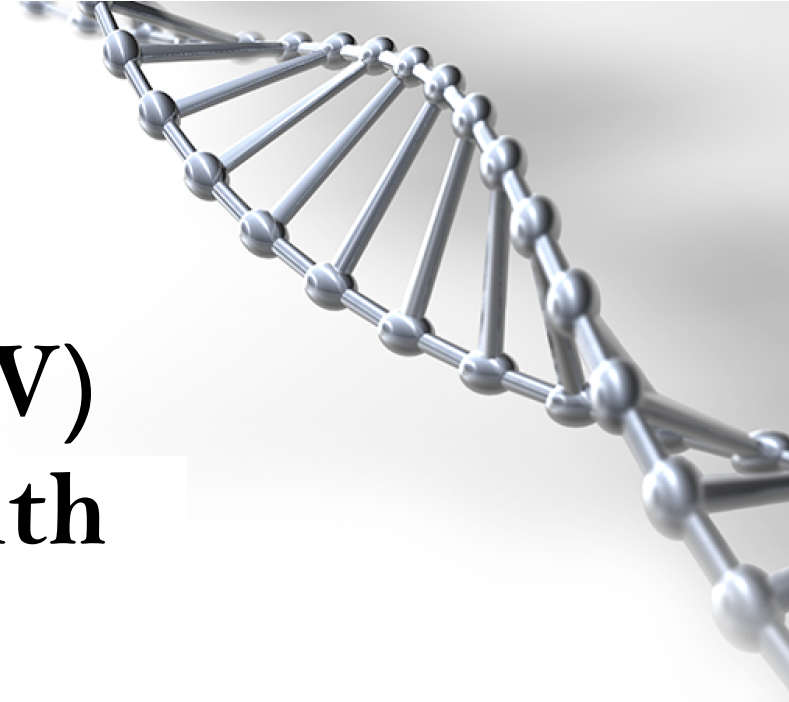
Showing all 4 rows.

	Forward	Complement	Reverse	RevComp
Forward_reads_5adapter	0	0	0	0
Forward_reads_3adapter	0	0	0	0
Reverse_reads_5adapter	0	0	0	0
Reverse_reads_3adapter	0	0	0	0

- If we didn't remove those sequences with 'N' bases, the result would likely look like this:

Showing all 4 rows.

	Forward	Complement	Reverse	RevComp
Forward_reads_5adapter	1060	1060	1060	1060
Forward_reads_3adapter	1060	1060	1060	1060
Reverse_reads_5adapter	1023	1023	1023	1023
Reverse_reads_3adapter	1023	1023	1023	1023



**Sample (ASV)  
inference with  
DADA2**

# Outline

- How are sequencing errors separated from amplicon sequence variants (ASVs)
  - How is error model learned
  - How are ASVs detected using the error model and abundance information
- What are the parameter options
- Things to take into account
- What are the resulting output files

# ASV inference



- Two step process
  1. Learn error model in order to remove sequencing errors
  2. Identify real biological sequences (ASVs) using the error model and abundance information
- Based on the learnErrors() and dada() functions of the DADA2 package
- Note: input reads should not contain ambiguous bases (Ns) or be too short
  - Use the Filter and trim tool
- Input file: Tar package of FASTQ files containing filtered reads
  - Specify if reads are paired end or single end and if they were produced with IonTorrent
  - Forward and reverse reads are processed independently
- Check the original publications:
  - B.Callahan et al. (2016), DADA2: High resolution sample inference from illumina amplicon data
  - M. Rosen et.al (2012), Denoising PCR –amplified metagenome data

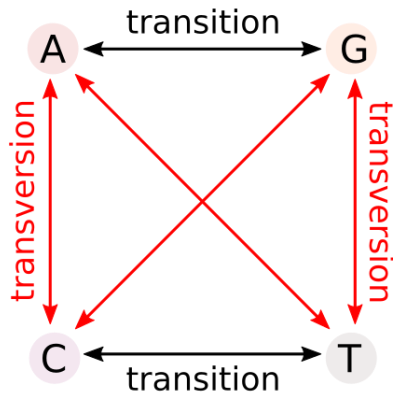
# LearnErrors()

- Creates an error rate model to be used by the dada() algorithm
  - Infer error rates for all possible transition or transversion point mutations
- Error model is learned by alternating estimation of the error rates and inference of sample composition until they converge
  - Starts with the assumption that the error rates are the maximum (takes the most abundant sequence ("center") and assumes it's the only sequence not caused by errors)
  - Compares the other sequences to the most abundant sequence
  - Uses at most  $10^8$  nucleotides for the error estimation
  - Uses parametric error estimation function of loess fit of the observed error rates
- Every amplicon dataset has a different set of error rates

Partition: Parwise alignments - center vs unique reads

Center:	ATGCATGCTACGTGCAGCTAGCTC <b>A</b> CATGCTAG...	10k reads	Q30
Unique2:	ATGCATGCTACGTGCAGCTAGCTC <b>G</b> CATGCTAG...	150 reads	Q3
Unique3:	ATGCATGCTACGTGCAGCTAGCTC <b>T</b> CATGCTAG...	50 reads	Q2
Unique4:	ATGCATGCTACGTGCAGCTAGCTC <b>C</b> CATGCTAG...	25 reads	Q1

...  
 Position: 1 5 10 15 20 25 30 ...



### transition probabilities

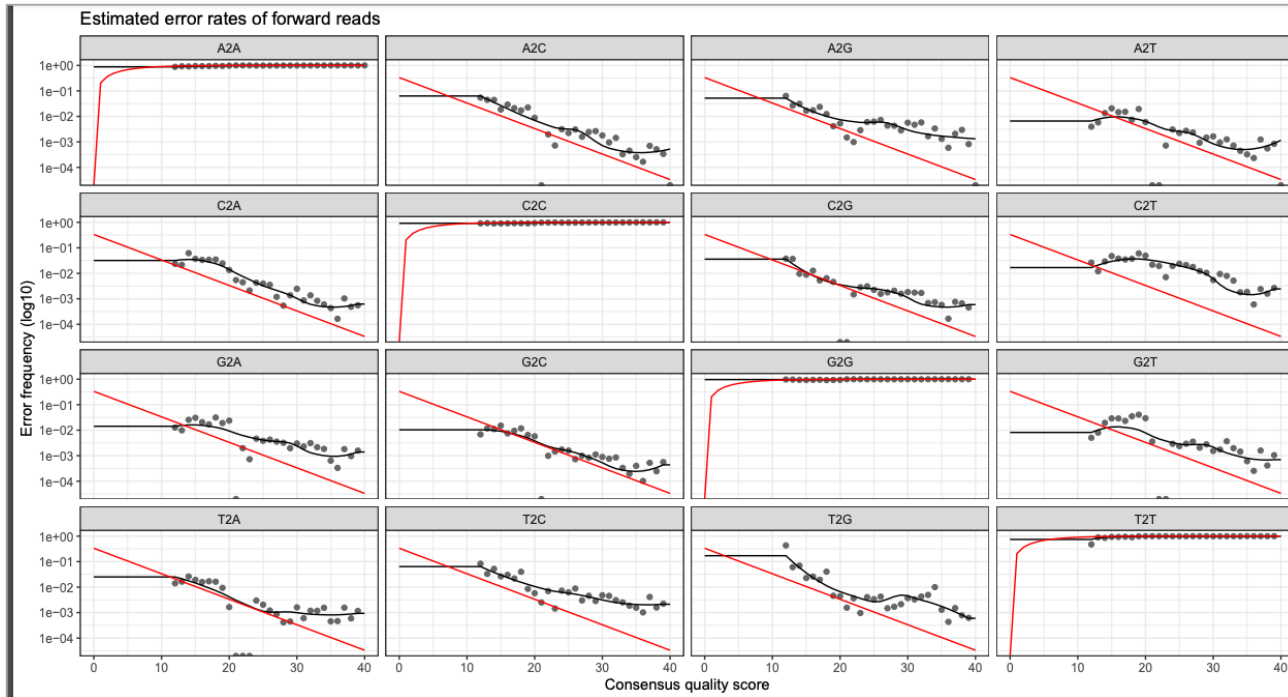
transitions	16	41	0	1	...	40	} Q scores
	A2A	0.99	0.99	0.99	0.99		
	A2T	0.00	0.00	0.00	0.00		
	...	...	...	...	...		
	C2C	0.99	0.99	0.99	0.99		

**nbases = 1e+08**

$\frac{1 \times 10^8 \text{ bases}}{250 \text{ bp long}} = 4 \times 10^5 \text{ reads}$

estimate error rates - *learnErrors()*

# Visualize estimated error rates



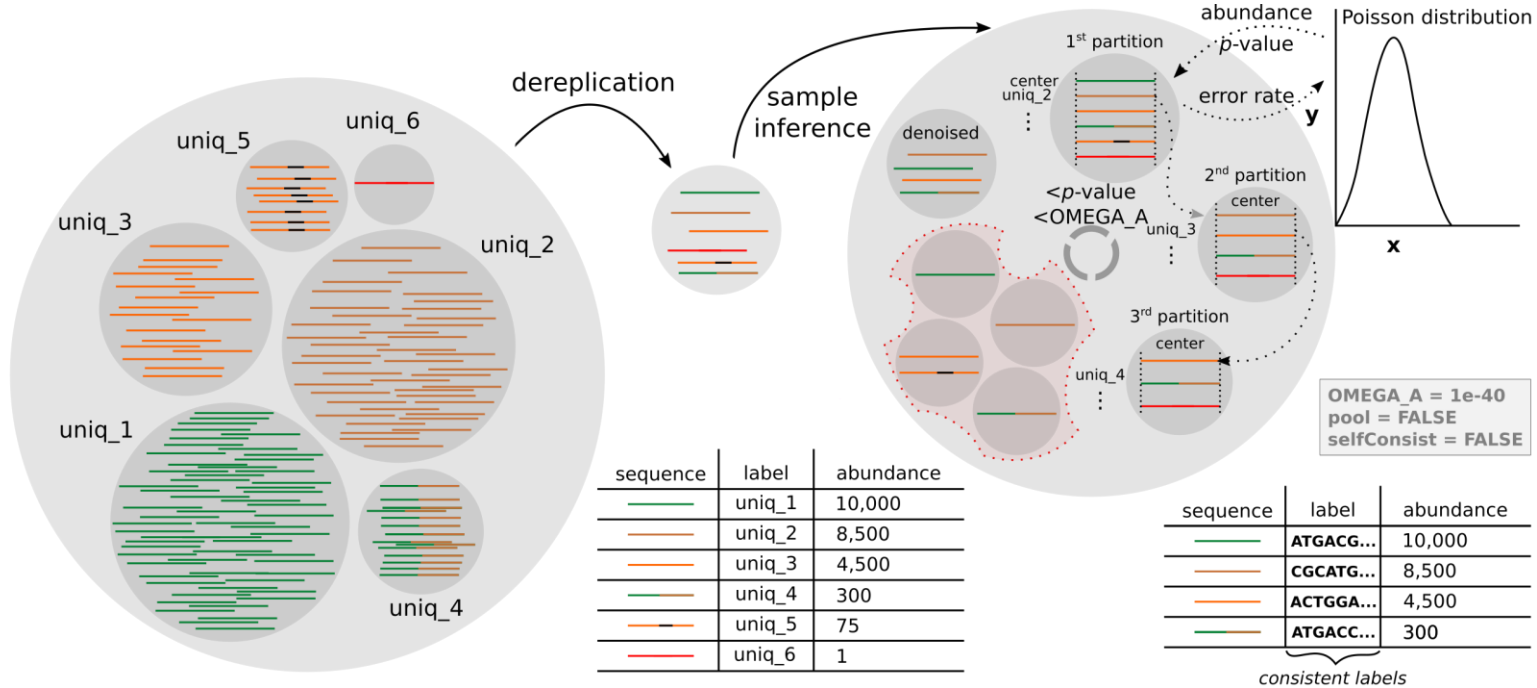
- Points = observed error rates
- Black line = estimated error rates after convergence
- Red line = Expected error rates based on the definition of the Q-scores

# Dada() – Divisive Amplicon Denoising Algorithm



- The core denoising algorithm to infer ASVs using the error rate matrix and abundance information
- It's a divisive hierarchical clustering algorithm
  - All unique sequences are assigned to one cluster
  - Subdivide the sequences to new clusters until it fits the error model
- Removes sequencing errors in order to reveal the ASVs
  1. Dereplicate sequences
  2. Use the most abundant unique sequence as the center of the cluster
  3. Calculate p-values for other sequences in the cluster based on the abundance and the quality profile for each unique sequence
  4. Compare the p-values to the OMEGA-A parameter ( $10^{-40}$ ) to decide whether the sequence was too abundant to be caused by sequencing errors
  5. Take the sequence with the lowest p-value smaller than the OMEGA-A parameter and use that sequence as the center of the second cluster
- Balance between
  - sensitivity (infer as many real ASVs as possible)
  - specificity (do not infer false positives)





denoise unique sequences (dereplicate and exclude singletons) - *dada()*

# Parameter options - pooling

- Independent
  - ASVs are inferred individually from each sample
  - Computationally easiest way
- Pseudo-pooling
  - Can increase sensitivity: might find rare variants
  - Runs the individual processing twice. Uses the ASVs found in all samples in the first run as priors in the second run.
  - Computationally harder
- The number of different ASVs found from all the samples is the same
  - The abundance of different ASVs in each sample can increase with pseudo-pooling
- The best choice depends on the data
- Note that singletons are not detected
  - Sequences with an abundance of 1 can't form a new cluster

# IonTorrent

- If you have IonTorrent data, a few parameter changes are recommended based on the error profile
  - `BAND_SIZE = 32` | By default use 16
    - Uses banded Needleman-Wunsch algorithm
    - Forgoes alignment in which the net number of gaps of one sequence relative to the other is higher than `BAND_SIZE`
  - `HOMOPOLYMER_GAP_PENALTY = -1` | By default use -8
    - Homopolymer regions are those with more than 2 repeated bases
    - The cost of gaps in homopolymer regions is set to -1 for alignment
    - By default gaps in homopolymer regions are treated as normal gaps

# Output

- DADA class objects saved as .Rda objects
  - Forward and reverse objects in separate files
- Summary.txt -> Information on learnErrors() and dada() functions
  - Key parameters used
  - Number of unique sequences in each file
  - Number of ASVs inferred from each file

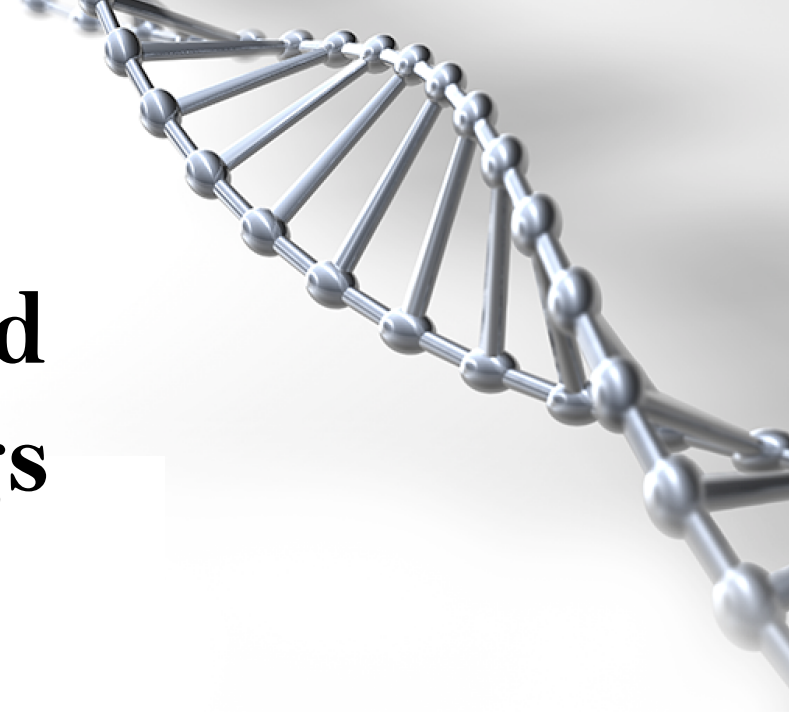
```
$F3D0_F_filt.fastq.gz
```

```
128 sequence variants were inferred from 1979 input unique sequences.
```

```
$F3D141_F_filt.fastq.gz
```

```
97 sequence variants were inferred from 1477 input unique sequences.
```

# Combine paired reads to contigs with DADA2



# Outline

- How paired reads are merged to contigs with DADA2
- What are the parameter options
- What are the resulting output files

## Combine paired reads to contigs with DADA2

- Tries to merge each denoised pair of forward and reverse reads
  - Aligns forward read with the reverse complement of the reverse read
  - Performs a Needleman-Wunsch alignment between the read pairs
- Input files
  - Tar package containing all the filtered forward and reverse FASTQ files
  - 2 denoised dada-class objects saved as .Rda files
    - Generated by the “Sample inference” tool
- Based on the mergePairs() function of DADA2 library

# Parameter options



By default:

- Overlap region needs to be at least 12 base pairs long
- No mismatches are allowed in the overlap region
- Overhangs are not trimmed off
  - Can result when the reverse reads extend past the start of the forward reads and vice versa

## Parameters

↶ Reset All

The minimum length of the overlap required for merging the forward and reverse reads

12

By default the overlap area should be at least 12 base pairs long.

The maximum number of mismatches allowed in the overlap region

0

By default no mismatches are allowed in the overlap region.

Should the overhangs in the alignment be trimmed off?

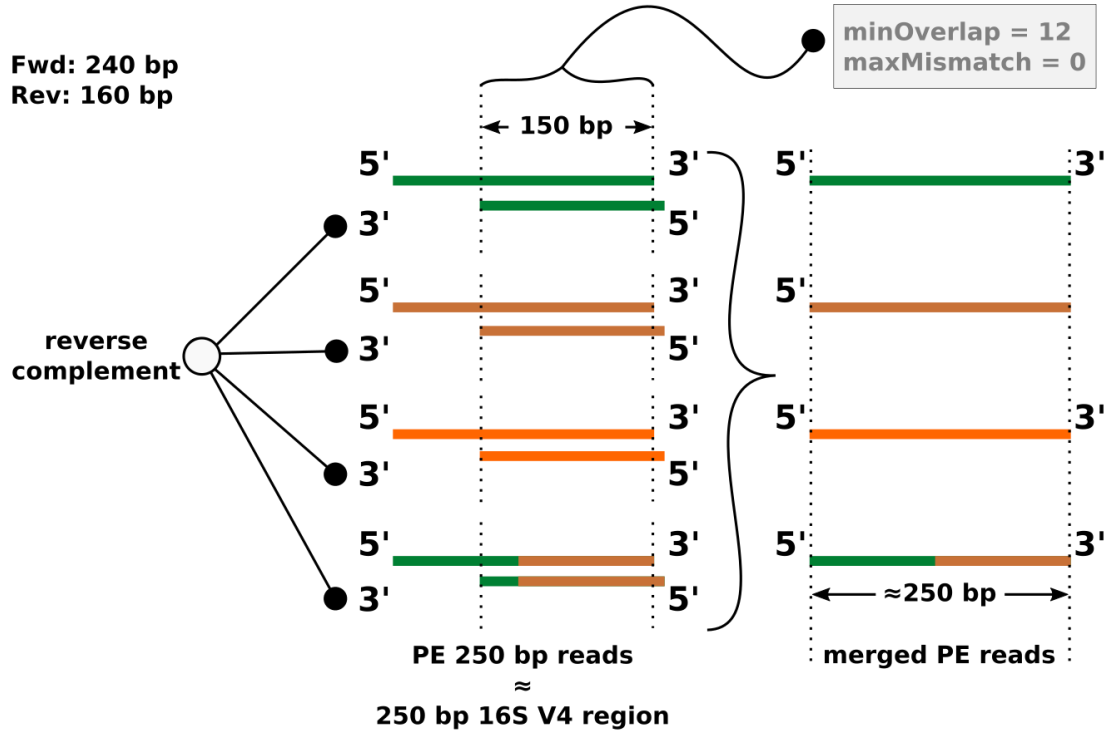
No

Should the overhangs be removed, when the reverse read extend past the start of the forward read and vice versa.



# Example: Paired end Illumina data

Fwd: 240 bp  
Rev: 160 bp



merge denoised forward and reverse reads - *mergePairs()*

Image by Antonio Sousa, 16S rRNA gene amplicon - upstream data analysis,  
<https://igcbioinformatics.github.io/biomeshiny/course/pages/dada2/Biodata.ptCrashCourses.html>

# Output files

1. Object `contigs.Rda` containing a list of data frames
2. Summary table `contigs_summary.tsv`
  - how many read pairs were rejected

	Forward dada object	Reverse dada object	After make contigs
F3D0	6976	6978	6533
F3D141	5331	5350	4973
F3D142	2799	2832	2595
F3D143	2822	2867	2552
F3D144	4151	4224	3643
F3D145	6592	6628	6079
F3D146	4447	4470	3968



**Make an ASV table  
and remove chimeras  
with DADA2**

# Outline

- How to create an amplicon sequence variant table
- How does it look like
- What are chimeras and how to remove those
- What are the parameter options
- What are the resulting output files

## Amplicon sequence variant table

- ASV table is created with the `makeSequenceTable()` command of DADA2 library
- Input file can be either:
  1. “contigs.Rda” object created when paired end reads were merged to contigs
  2. “dada\_forward.Rda” object containing a list of dada-class objects if having single end reads
- Shows the distribution of ASVs in each sample
  - Very similar to OTU table
- ASV table contains still chimeras which need to be removed

# ASV table

*Amplicon Sequence Variants - ASVs*

	<b>ATGACG...</b>	<b>CGCATG...</b>	<b>ACTGGA...</b>	<b>ATGACC...</b>
sample 1	263	3389	782	85
sample 2	1809	1388	877	40
sample 3	4146	2072	2365	175
sample 4	3782	1651	476	0

**ASVs table**

construct an ASV table - *makeSequenceTable()*

Image by Antonio Sousa, 16S rRNA gene amplicon - upstream data analysis,  
<https://igcbioinformatics.github.io/biomeshtiny/course/pages/dada2/Biodata.ptCrashCourses.html>

## Remove chimeras

- Chimeras are artifact sequences formed by two or more biological sequences
  - Incomplete amplification during PCR allows subsequent PCR cycles to use a partially extended strand to bind to the template of a similar sequence
  - The partially extended strand then acts as a primer to extend and form a chimeric sequence
- Based on `removeBimeradenovo()` of DADA2 library
- Used to identify and remove chimeras from the ASV table

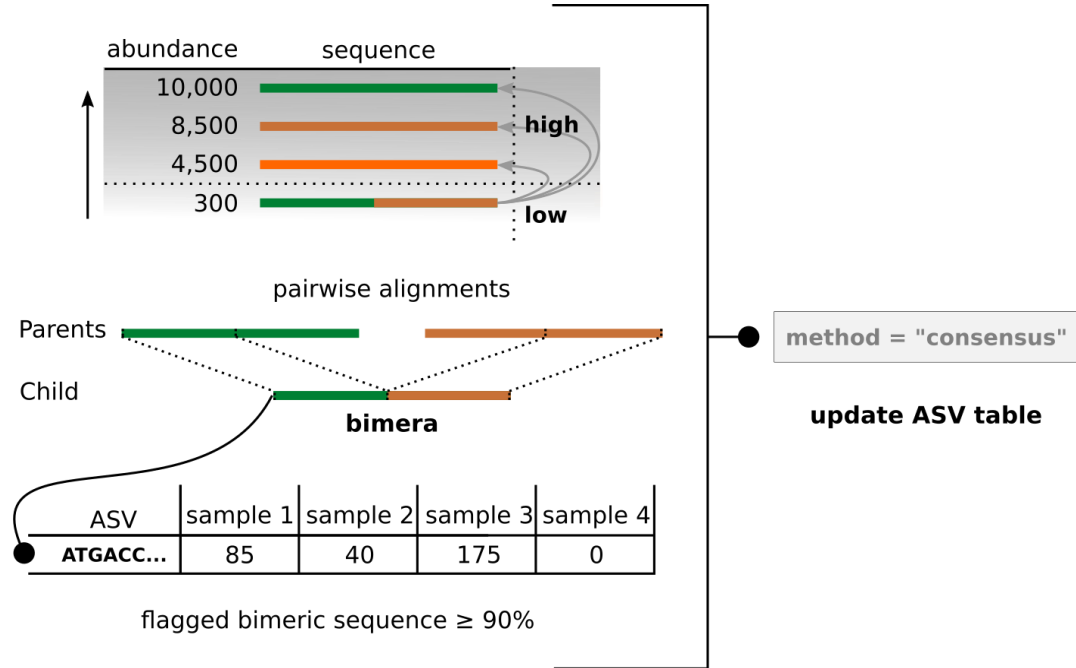
# Parameter options

- Consensus
  - Chimeras are identified on sample-by-sample basis
  - If an ASV is identified as a chimera in most of the samples, it will be removed
- Pooled
  - Each sequence is compared against the more abundant sequences of all samples



## How it works

- Assumes that chimeras arise from two parent sequences
  - Bimera is a chimera formed by exactly two parent sequences
- Performs Needleman-Wunsch pairwise alignments to compare less abundant sequences with the most abundant sequences
- If the less abundant sequence can be reconstructed by combining a left segment and a right segment from two more abundant sequences, the sequence is removed as chimeric



remove chimeras - *removeBimeraDenovo()*

Image by Antonio Sousa, 16S rRNA gene amplicon - upstream data analysis,  
<https://igcbioinformatics.github.io/biomeshiny/course/pages/dada2/Biodata.ptCrashCourses.html>

# Output files

1. Seqtab\_nochim.Rda
2. Sequence\_table\_nochim.tsv
3. Summary.txt
4. Reads\_summary.tsv

## sequence\_table\_nochim.tsv

Showing all 19 rows.

	ASV1	ASV2	ASV3	ASV4	ASV5	ASV6	ASV7	ASV8	ASV9	ASV10	ASV11	ASV12	ASV13	ASV14	ASV15	ASV16	ASV17	ASV18
F3D0	579	345	449	430	154	470	282	184	45	158	17	217	52	104	93	80	100	69
F3D141	444	362	345	502	189	331	243	321	167	130	168	146	12	65	33	103	149	43
F3D142	289	304	158	164	180	181	163	89	89	78	42	98	103	64	12	52	116	30
F3D143	228	176	204	231	130	244	152	83	109	67	78	111	43	61	9	40	0	20
F3D144	421	277	302	357	104	353	240	41	158	155	269	146	16	81	11	113	0	45
F3D145	645	489	522	583	307	476	396	125	202	229	317	258	22	125	15	126	195	105
F3D146	325	230	254	388	179	275	214	71	113	88	178	147	4	58	25	35	0	35
F3D147	1495	1215	913	1089	453	1182	861	75	769	269	448	560	147	292	74	306	260	143
F3D148	863	729	581	853	443	872	579	507	409	198	411	432	18	199	56	270	259	117
F3D149	883	779	723	897	417	637	560	515	426	288	478	301	88	164	42	177	82	119
F3D150	317	229	399	471	169	216	238	120	241	149	61	98	64	75	19	30	0	49
F3D1	405	353	231	69	140	41	96	190	69	106	102	40	129	28	325	0	0	31
F3D2	3488	1587	1175	472	338	115	325	1211	434	609	55	41	330	107	368	17	70	190
F3D3	988	602	465	200	402	25	167	381	307	298	171	0	94	61	46	24	0	103
F3D5	327	268	284	158	151	23	123	207	178	207	61	0	48	38	87	36	57	37
F3D6	1014	674	588	404	476	17	282	261	205	242	45	0	422	106	57	0	0	39
F3D7	648	504	438	314	470	11	195	213	176	276	16	0	117	73	40	0	0	22
F3D8	272	352	349	147	582	0	130	286	113	197	19	0	145	65	45	6	0	22
F3D9	511	423	482	206	596	0	210	438	146	225	27	0	182	72	99	0	0	38

ASVs renamed for visualization purposes

# Reads\_summary.tsv

Showing all 19 rows.

	After make contigs	Removed chimeras
F3D0	6533	6521
F3D141	4973	4850
F3D142	2595	2521
F3D143	2552	2518
F3D144	3643	3504
F3D145	6079	5820
F3D146	3968	3879
F3D147	14233	13006
F3D148	10528	9935
F3D149	11155	10653
F3D150	4349	4240
F3D1	5028	5017
F3D2	17431	16835
F3D3	5850	5486
F3D5	3716	3716
F3D6	6865	6678
F3D7	4426	4215
F3D8	4560	4531
F3D9	6093	6016

❖ Note: it's common that many of the ASVs are removed as chimeric, but most of the reads should not!

## Summary.txt

```
After dada() algorithm sequence table consist of:
19 samples and 279 amplicon sequence variants
```

```
Distribution of the amplicon sequence variant's lengths: Column names are the sequence lengths
```

```
          251 252 253 254 255
Counts:   1 85 186  5  2
```

```
###Removing Chimeras:###
```

```
Identified 61 bimeras out of 279 input sequences
Total amount of ASVs is: 218
```



# Assign taxonomy with DADA2

# Outline

- Methods to assign taxonomy to ASVs
- What are the parameter options
- What things to take into account
- What are the resulting output files

# Assign Taxonomy

- Based on the `assignTaxonomy()` function of DADA2 library
- Uses the naive Bayesian classifier method of Wang to assign taxonomy across multiple ranks
  - Compares the k-mer profile of the query sequences against the reference sequences with assigned taxonomies
  - Calculates the bootstrapping confidence score for the assignment
  - Uses k-mer size of 8 and 100 iterations
- There is a separate command for species level assignment called `addSpecies()`

## Assing Taxonomy – Input files

- As input you need to give the ASV table saved as .Rda file
- Uses SILVA reference files by default if no reference files were given
  - To use other reference databases, download the DADA2 supported reference file and bring it to Chipster
  - Find DADA2 supported reference databases:  
<https://benjjneb.github.io/dada2/training.html>



# Parameter options

- Specify the minimum bootstrap confidence score to assign taxonomy to ASVs
  - Threshold of 50% means that at least half of the iterations should return the same assignment for each level
  - By default threshold is set to 50% which is recommended for sequences shorter than 250 bases
    - Otherwise 80% is recommended

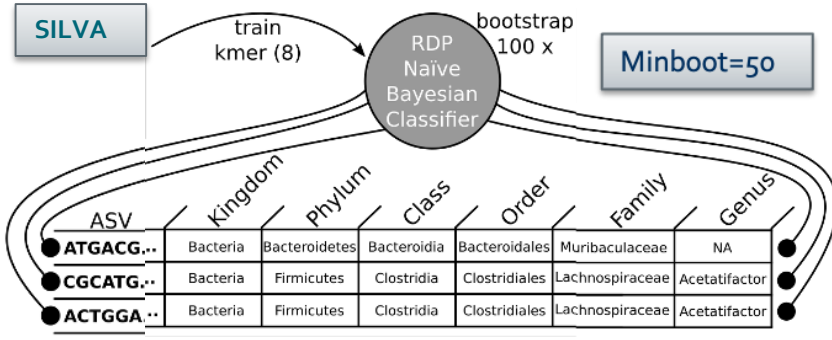
## Parameters

<b>The minimum bootstrap confidence for assigning a taxonomic level</b>	<input type="text" value="50"/>
The minimum bootstrap confidence score for assigning a taxonomic level	
<b>Try the reverse-complement of each sequence for classification if it is a better match to the reference sequences</b>	<input type="text" value="no"/>
If set to yes, use the reverse-complement of each sequences for classification if it is a better match to the reference sequences than the original sequence.	
<b>Exact species level assignment?</b>	<input type="text" value="yes"/>
Do you want to assign the sequences to the species level if there is an exact match 100% identity between ASVs and sequenced reference strains?	
<b>Combine the taxonomy and the sequence table</b>	<input type="text" value="yes"/>
If set to yes, it combines the taxonomy and the sequence/ASV table into one .tsv file, otherwise the tsv file consist only of the taxonomy table.	

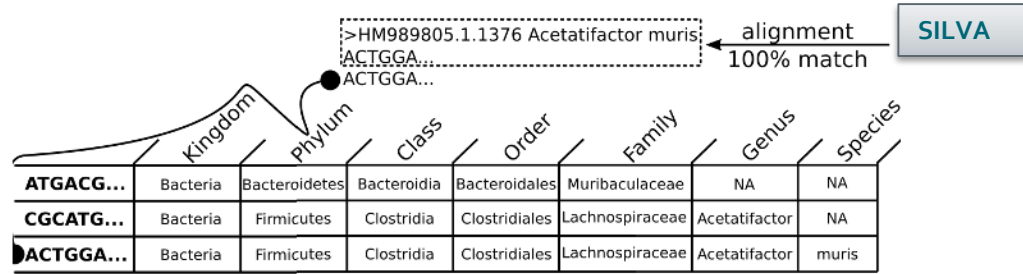
## AddSpecies()

- If the parameter “Exact species level assignment “ is set to yes, the addSpecies() function for species level assignment is used
- Based on exact (100% identity) string matching against a reference database
  - Assign to species level if there is no ambiguity and all exact matches were to the same species
  - 100% identity matching is recommended for 16S amplicon data
    - See Robert. C. Edgar, (2018), Updating the 97% identity threshold for 16S ribosomal RNA OTUs
- Uses SILVA reference files if those were used for assignTaxonomy() as well
  - Otherwise give your own reference file as input

# Example:



**AssignTaxonomy()**



**addSpecies()**

Images by Antonio Sousa, 16S rRNA gene amplicon - upstream data analysis,  
<https://igcbioinformatics.github.io/biomeshinycourse/pages/dada2/Biodata.ptCrashCourses.html>

# Output files

1. taxonomy-assignment-matrix.Rda
2. taxa\_seqtab\_combined.tsv
  - If “Combine the taxonomy and the sequence table” is set to yes, both of those tables are combined to taxa\_seqtab\_combined.tsv
  - The names of the ASVs are renamed for visualization purposes. The .Rda object has still the full DNA sequences

Showing the first 100 of 218 rows. View in [full screen](#) to see all rows.

[Full Screen](#)

	Kingdom	Phylum	Class	Order	Family	Genus
ASV1	Bacteria	Bacteroidota	Bacteroidia	Bacteroidales	Muribaculaceae	NA
ASV2	Bacteria	Bacteroidota	Bacteroidia	Bacteroidales	Muribaculaceae	NA
ASV3	Bacteria	Bacteroidota	Bacteroidia	Bacteroidales	Muribaculaceae	NA
ASV4	Bacteria	Bacteroidota	Bacteroidia	Bacteroidales	Muribaculaceae	NA
ASV5	Bacteria	Bacteroidota	Bacteroidia	Bacteroidales	Bacteroidaceae	Bacteroides
ASV6	Bacteria	Bacteroidota	Bacteroidia	Bacteroidales	Muribaculaceae	NA
ASV7	Bacteria	Bacteroidota	Bacteroidia	Bacteroidales	Muribaculaceae	NA
ASV8	Bacteria	Bacteroidota	Bacteroidia	Bacteroidales	Rikenellaceae	Alistipes
ASV9	Bacteria	Bacteroidota	Bacteroidia	Bacteroidales	Muribaculaceae	NA
ASV10	Bacteria	Bacteroidota	Bacteroidia	Bacteroidales	Muribaculaceae	NA



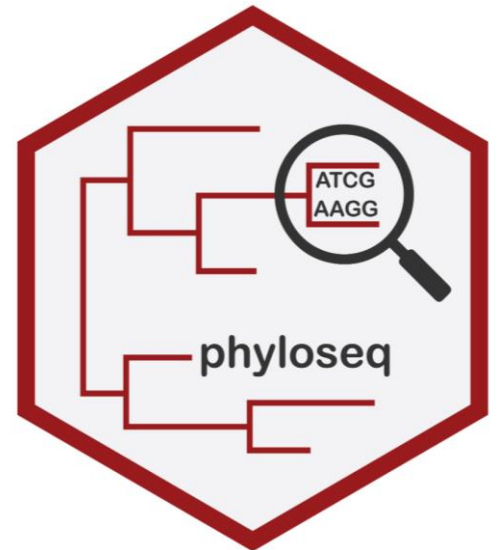
**Make a phyloseq  
object**

# Outline

- What is a phyloseq object
- How to create it
- What are the resulting output files
- How to extract information from the phyloseq object

# Make a phyloseq object

- Phyloseq is an R package to import, store and analyze microbial community data
  - Stores all related sequencing data which makes the analyses easier
- To create a phyloseq .Rda object you need to give as input
  1. ASV table saved as .Rda object
  2. Taxonomy table saved as .Rda object



## Output files

- Creates a phyloseq object called ps\_nophe.Rda
  - Produces a phenodata file used to specify sample information
- Summary file : ps\_summary.txt
  - The full DNA sequences of ASVs stored to refseq() slot
  - You can get those with the tool “Extract information from the phyloseq object”

```
### Phyloseq object ###
```

```
phyloseq-class experiment-level object
otu_table()   OTU Table:          [ 218 taxa and 19 samples ]
tax_table()   Taxonomy Table:     [ 218 taxa by 7 taxonomic ranks ]
refseq()      DNASTringSet:      [ 218 reference sequences ]
```





# Phenodata table

- Table you can edit via the Chipster interface
- Use to specify and sort samples to different groups
  - Makes data analysis easier

sample	original_name	group	×	diet	×	description
F3D0		a		low		
F3D141		b		low		
F3D142		b		low		
F3D143		b		low		
F3D144		b		low		
F3D145		b		low		
F3D146		b		low		
F3D147		b		low		
F3D148		b		high		
F3D149		b		high		
F3D150		b		high		
F3D1		a		high		
F3D2		a		high		
F3D3		a		high		
F3D5		a		high		
F3D6		a		high		
F3D7		a		high		
F3D8		a		high		
F3D9		a		high		

# Merge phenodata to the phyloseq object

- Store the sample information from the Phenodata table to the phyloseq object
- As input, give the phyloseq .Rda object and the Phenodata table
- Specify the column from the Phenodata table which contains the sample names / IDs

## Parameters

A button with a circular arrow icon and the text "Reset All".

Phenodata variable with sequencing sample IDs

Phenodata variable with unique IDs for each community profile.

sample 



# Output files

1. Phyloseq object ps.Rda
2. Summary file ps\_sample\_summary.txt
  - If you modify the phenodata table, you need to merge the phenodata table to the phyloseq object again!

```
### Phyloseq object with sample information combined###  
  
phyloseq-class experiment-level object  
otu_table() OTU Table:      [ 218 taxa and 19 samples ]  
sample_data() Sample Data:  [ 19 samples by 5 sample variables ]  
tax_table()  Taxonomy Table: [ 218 taxa by 7 taxonomic ranks ]  
refseq()     DNASTringSet:   [ 218 reference sequences ]  
  
### Sample names ###  
  
[1] "F3D0"  "F3D141" "F3D142" "F3D143" "F3D144" "F3D145" "F3D146" "F3D147"  
[9] "F3D148" "F3D149" "F3D150" "F3D1"  "F3D2"  "F3D3"  "F3D5"  "F3D6"  
[17] "F3D7"  "F3D8"  "F3D9"  
  
### Sample variables ###  
  
[1] "sample"      "original_name" "chiptype"      "group"  
[5] "description"
```

# Extract information from the phyloseq object

- Used to extract information stored to the phyloseq object
- Give a phyloseq object .Rda as input
  - Can be used to access information after the DADA2 or Mothur based workflow

## Parameters

Do you want to extract the OTU table	<input type="text" value="no"/>
Do you want to extract the taxonomy table	<input type="text" value="no"/>
Do you want to extract the full DNA sequences stored to refseq	<input type="text" value="no"/>
Do you want to extract the sample information	<input type="text" value="no"/>

# Full DNA sequences stored in the refseq() slot of the Phyloseq object

- Get access to the full sequences of each ASV
  - Can be used to merge the found ASVs with other datasets and index into reference databases

Showing the first 100 of 218 rows. View in [full screen](#) to see all rows.

[Full Screen](#)

ASV1	TACGGAGGATGCGAGCGTTATCCGGATTTATGGGTTTTAAAGGGTGCGCAGGCCGGAAGATCAAGTCAGCGGTAAAATTGAGAGGCTCAACCTCTTCGAGCCGTTGA/
ASV2	TACGGAGGATGCGAGCGTTATCCGGATTTATGGGTTTTAAAGGGTGCGCAGGCCGACTCTCAAGTCAGCGGTCAAATCGCGGGGCTCAACCCCGTTCCGCCGTTG/
ASV3	TACGGAGGATGCGAGCGTTATCCGGATTTATGGGTTTTAAAGGGTGCGTAGCGGGCTGTTAAGTCAGCGGTCAAATGTCGGGGCTCAACCCCGGCTGCCGTTG/
ASV4	TACGGAGGATGCGAGCGTTATCCGGATTTATGGGTTTTAAAGGGTGCGTAGCGGGCTTTAAGTCAGCGGTAAAAATTCGGGGCTCAACCCCGTCCGGCCGTTGA/
ASV5	TACGGAGGATCCGAGCGTTATCCGGATTTATGGGTTTTAAAGGGAGCGTAGGTGGATTGTTAAGTCAGTTGTGAAAGTTTGCGGCTCAACCGTAAAATTGCAGTTGA/
ASV6	TACGGAGGATGCGAGCGTTATCCGGATTTATGGGTTTTAAAGGGTGCGTAGCGGCCTGCCAAGTCAGCGGTAAAATTGCGGGGCTCAACCCCGTACAGCCGTTG/
ASV7	TACGGAGGATGCGAGCGTTATCCGGATTTATGGGTTTTAAAGGGTGCGTAGCGGGATGCCAAGTCAGCGGTAAAAAAGCGGTGCTCAACGCCGTCGAGCCGTTG/
ASV8	TACGGAGGATCAAGCGTTATCCGGATTTATGGGTTTTAAAGGGTGCGTAGCGGTTTCGATAAGTTAGAGGTGAAATCCCGGGGCTCAACTCCGGCACTGCCTCTG/
ASV9	TACGGAGGATGCGAGCGTTATCCGGATTTATGGGTTTTAAAGGGTGCGTAGCGGATCGTTAAGTCAGTGGTCAAATGAGGGGCTCAACCCCTTCCCGCCATTGA/
ASV10	TACGGAGGATGCGAGCGTTATCCGGATTTATGGGTTTTAAAGGGTGCGTAGCGGGATGTCAAGTCAGCGGTAAAATTGTGGAGCTCAACTCCATCGAGCCGTTGA/
ASV11	TACGTAGGTGGCAAGCGTTGTCCGGATTTATGGGCGTAAAGCGAGTGCAGGCGGTTCAATAAGTCTGATGTGAAAGCCTTCGGCTCAACCGGAGAATTGCATCAG/
ASV12	TACGGAGGATGCGAGCGTTATCCGGATTTATGGGTTTTAAAGGGTGCGTAGCGGCCTGCCAAGTCAGCGGTAAAATTGCGGGGCTCAACCCCGTACAGCCGTTG/
ASV13	TACGTAGGTGGCAAGCGTTATCCGGATTTATGGGCGTAAAGGGAACGCAGGCGGCTTTAAGTCTGATGTGAAAGCCTTCGGCTTAACCGGAGTAGTGCATTGGA/
ASV14	TACGGAGGATGCGAGCGTTATCCGGATTTATGGGTTTTAAAGGGTGCGTAGCGGGATGCCAAGTCAGCGGTAAAAATGCGGTGCTCAACGCCGTCGAGCCGTTG/
ASV15	TACGTAGGGGGCAAGCGTTATCCGGATTTACTGGGTGTAAAGGGAGCGTAGACGGCAGCGCAAGTCTGGAGTGAAATGCCGGGGCCCAACCCCGGAAGTGCCTTTG/
ASV16	TACGTAGGTGGCAAGCGTTATCCGGAATTATGGGCGTAAAGAGCGCGCAGGTGGTTAATTAAGTCTGATGTGAAAGCCCACGGCTTAACCGTGGAGGGTCATTGG/
ASV17	TACGGAGGATGCGAGCGTTATCCGGATTTATGGGTTTTAAAGGGTGCGCAGGCCGGATGCCAAGTCAGCGGTCAAATTCGGGGGCTCAACCCCGACCTGCCGTTG/
ASV18	TACGGAGGATGCGAGCGTTATCCGGATTTATGGGTTTTAAAGGGTGCGTAGCGGTCGGTTAAGTCAGCGGTAAAATTGCGGGGCTCAACCCCGTCGAGCCGTTGA/
ASV19	TACGTAGGGAGCGAGCGTTATCCGGATTTATGGGTGTAAAGGGTGCGTAGCGGTAATACAGGTCTTTGGTATAAGCCCGAAGCTTAACCTCGGTAAGCCAGAGAA/



# IonTorrent or ITS data with DADA2

# Outline

- Main parts of ITS and IonTorrent analysis with DADA2
- What are the differences compared to Illumina data
- Things to take into account

## Things to note

- Use cutadapt to remove the adapter/primer sequences
- When filtering and trimming your reads, use “Remove reads which are shorter than this” instead of truncation
  - If the read length varies
- ITS
  - Download the UNITE reference files from the website and give as input
- IonTorrent:
  - Use parameter IonTorrent in tool sample inference
    - Uses specific denoising algorithm options recommended for IonTorrent data
    - HOMOPOLYMER\_GAP\_PENALTY = -1
    - BANDSIZE= 32





**New features for  
microbial  
community  
analyses in  
Chipster 01.2023**

## Notes

- DADA workflow installed, as well as Cutadapt
- Tool “Extract information from a phyloseq object”
- Phyloseq, FastQC and MultiQC versions updated
- Alpha diversity estimate creates a boxplot with Wilcoxon rank sum test
- Mothur updated to new version 1.48.0
  - Not using group file anymore