PARIS Cimi Centre d'Immunologie et des Maladies Infectieuses

LU3SV530 - Metagenomique du Sol

Analyse bioinformatique de données gène de l'ARNr 16S

TD 1 - Dada2

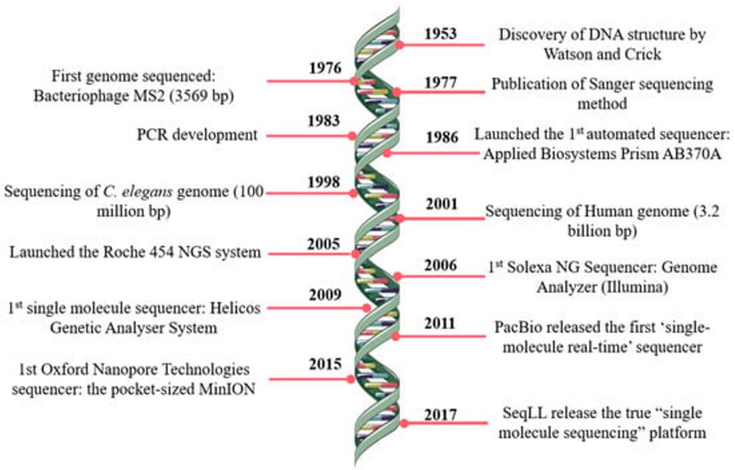
Djelika TRAORE & Martin LARSEN

www.immulab.fr

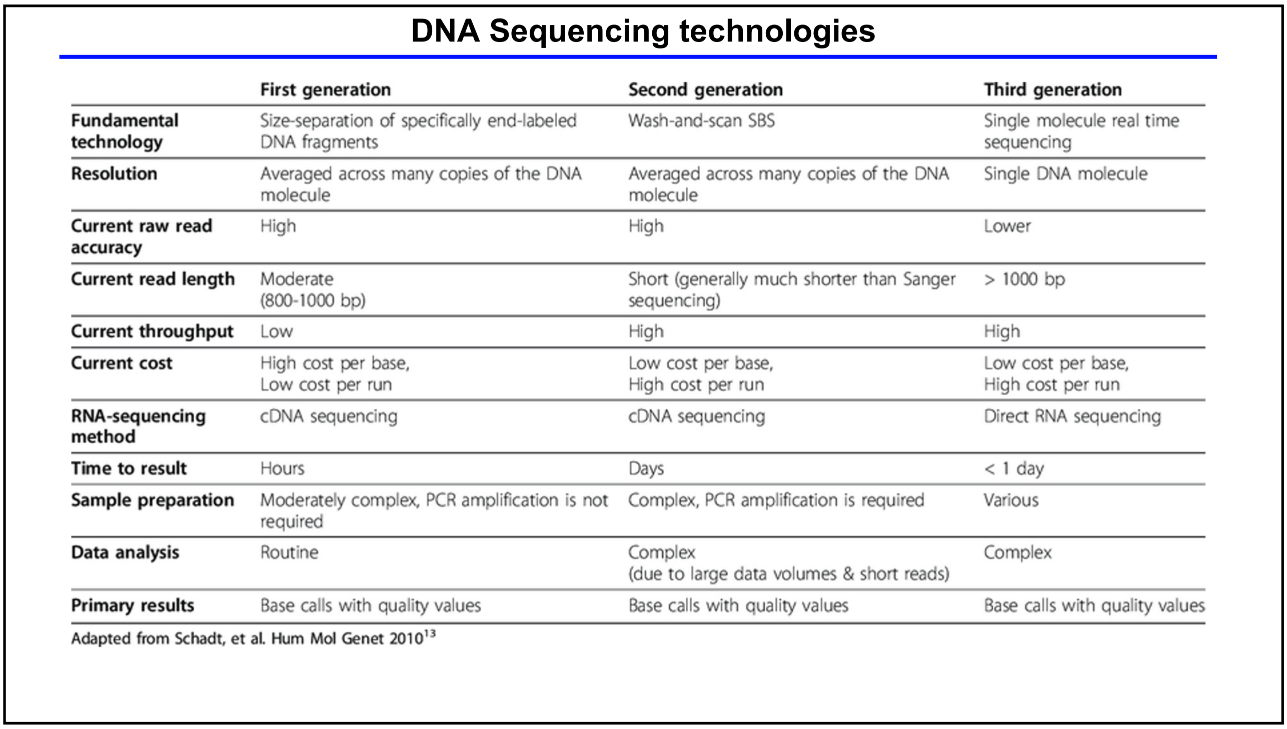
INSERM U1135, CHU Pitié-Salpetrière, Paris, France

1

History of DNA Sequencing


1953 Discovery of DNA structure by Watson and Crick
1976 First genome sequenced: Bacteriophage MS2 (3569 bp)
1977 Publication of Sanger sequencing method
1983 PCR development
1986 Launched the 1st automated sequencer: Applied Biosystems Prism AB370A
1998 Sequencing of C. elegans genome (100 million bp)
2001 Sequencing of Human genome (3.2 billion bp)
2005 Launched the Roche 454 NGS system
2006 1st Solexa NG Sequencer: Genome Analyzer (Illumina)
2009 1st single molecule sequencer: Helicos Genetic Analyser System
2011 PacBio released the first 'single-molecule real-time' sequencer
2015 1st Oxford Nanopore Technologies sequencer: the pocket-sized MinION
2017 SeqLL release the true "single molecule sequencing" platform

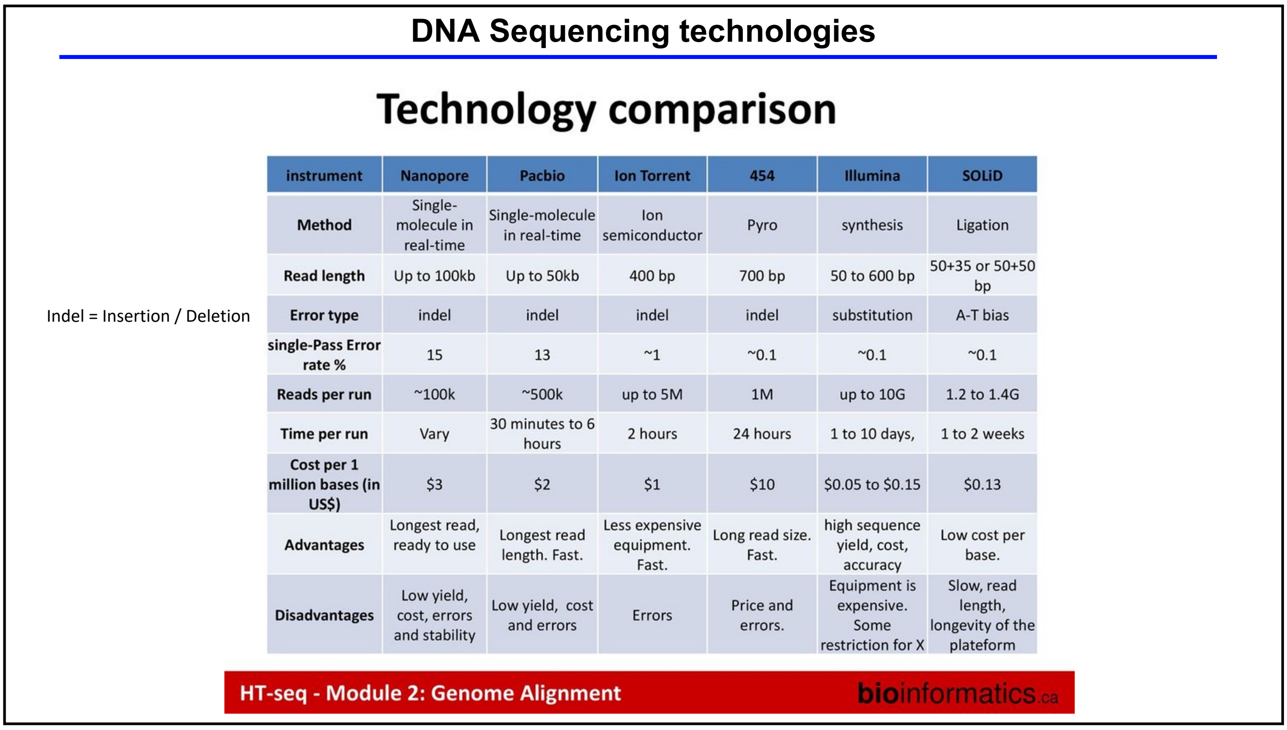
2



DNA Sequencing technologies

	First generation	Second generation	Third generation
Fundamental technology	Size-separation of specifically end-labeled DNA fragments	Wash-and-scan SBS	Single molecule real time sequencing
Resolution	Averaged across many copies of the DNA molecule	Averaged across many copies of the DNA molecule	Single DNA molecule
Current raw read accuracy	High	High	Lower
Current read length	Moderate (800-1000 bp)	Short (generally much shorter than Sanger sequencing)	> 1000 bp
Current throughput	Low	High	High
Current cost	High cost per base, Low cost per run	Low cost per base, High cost per run	Low cost per base, High cost per run
RNA-sequencing method	cDNA sequencing	cDNA sequencing	Direct RNA sequencing
Time to result	Hours	Days	< 1 day
Sample preparation	Moderately complex, PCR amplification is not required	Complex, PCR amplification is required	Various
Data analysis	Routine	Complex (due to large data volumes & short reads)	Complex
Primary results	Base calls with quality values	Base calls with quality values	Base calls with quality values

Adapted from Schadt, et al. Hum Mol Genet 2010[13]



DNA Sequencing technologies

Technology comparison

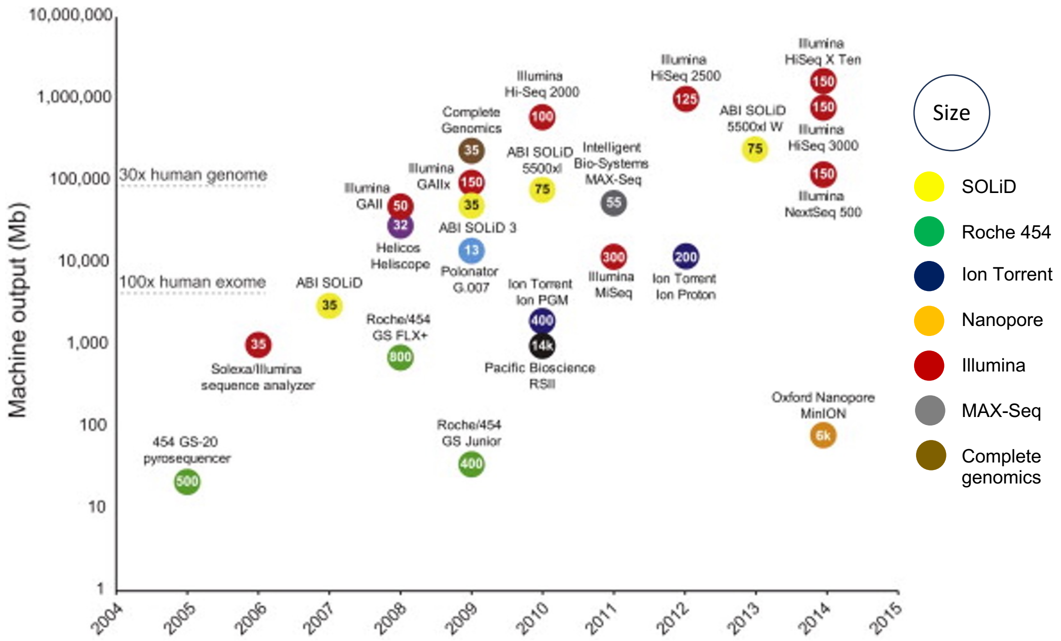
Indel = Insertion / Deletion

instrument	Nanopore	Pacbio	Ion Torrent	454	Illumina	SOLiD
Method	Single-molecule in real-time	Single-molecule in real-time	Ion semiconductor	Pyro	synthesis	Ligation
Read length	Up to 100kb	Up to 50kb	400 bp	700 bp	50 to 600 bp	50+35 or 50+50 bp
Error type	indel	indel	indel	indel	substitution	A-T bias
single-Pass Error rate %	15	13	~1	~0.1	~0.1	~0.1
Reads per run	~100k	~500k	up to 5M	1M	up to 10G	1.2 to 1.4G
Time per run	Vary	30 minutes to 6 hours	2 hours	24 hours	1 to 10 days,	1 to 2 weeks
Cost per 1 million bases (in US$)	$3	$2	$1	$10	$0.05 to $0.15	$0.13
Advantages	Longest read, ready to use	Longest read length. Fast.	Less expensive equipment. Fast.	Long read size. Fast.	high sequence yield, cost, accuracy	Low cost per base.
Disadvantages	Low yield, cost, errors and stability	Low yield, cost and errors	Errors	Price and errors.	Equipment is expensive. Some restriction for X	Slow, read length, longevity of the plateform

HT-seq - Module 2: Genome Alignment

bioinformatics.ca

DNA Sequencing technologies - Output

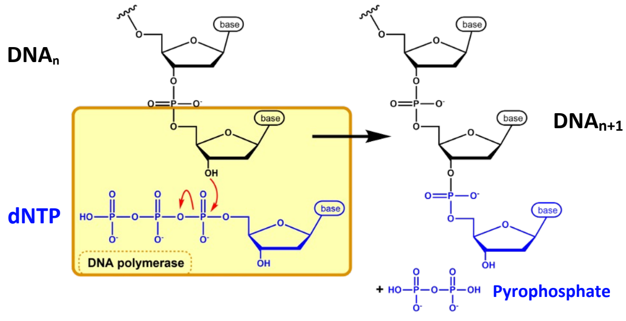


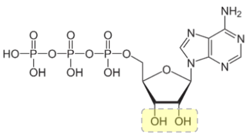
8

Sequencing by Synthesis

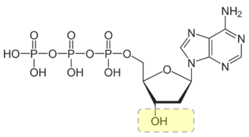
Sequencing by synthesis is a method in which DNA polymerases and dNTPs are used to replicate (synthesize) the DNA strand to be sequenced

DNA replication catalyzed by DNA polymerase

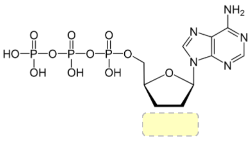




ATP
Adenosine triphosphate
Cellular energy currency



dATP
Deoxyadenosine triphosphate
Substrate for DNA polymerase

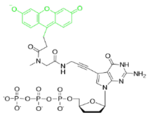


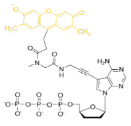
ddATP
Dideoxyadenosine triphosphate
Substrate for DNA polymerase,
but chain terminates (Sanger seq)

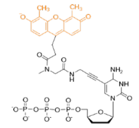
9

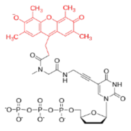
1st generation sequencing by size

Sanger

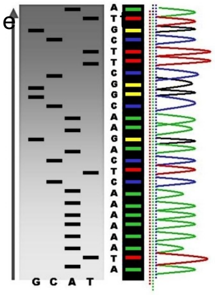

G-505


A-512


C-519


T-526

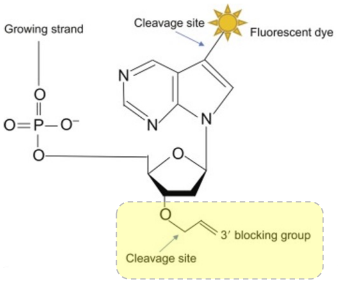
radioactiv fluorescen
e

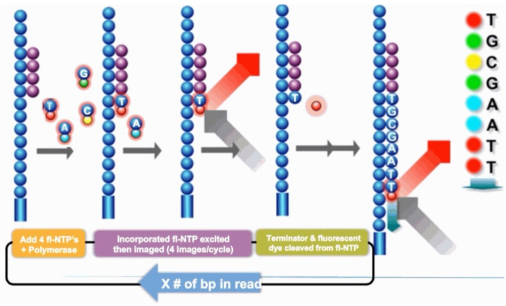


10

2nd generation sequencing by synthesis

Illumina

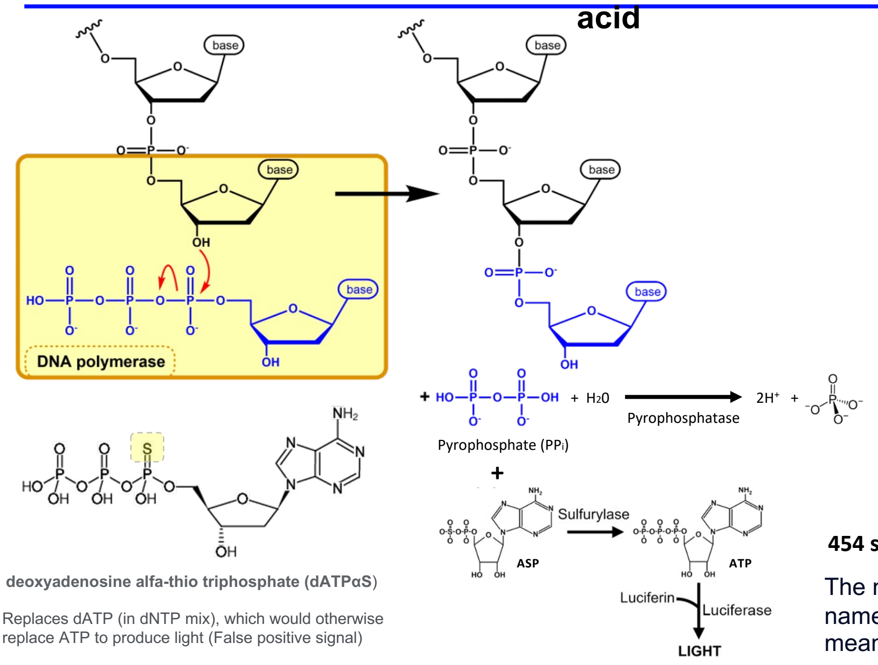



G
T
A
C
A
T
T
T
G
C
G
A
A
T
T
Add 4 fl-NTP's + Polymerase
Incorporated fl-NTP excited then imaged (4 images/cycle)
Terminator & fluorescent dye cleaved from fl-NTP
X # of bp in read

Alternatives:
454 FLX
PACBIO
Ion Torrent

13

Sequencing by synthesis - Markers of dNTP incorporation - Pyrophosphate and acid



deoxyadenosine alfa-thio triphosphate (dATPαS)

Replaces dATP (in dNTP mix), which would otherwise replace ATP to produce light (False positive signal)

Ion torrent

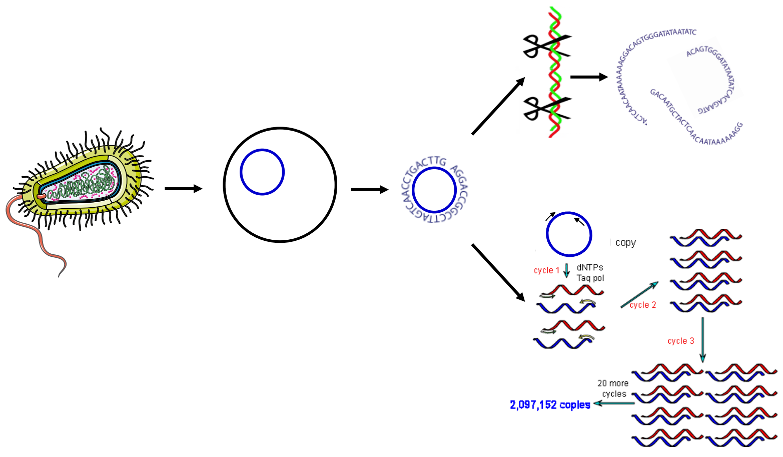
Based on detection of change in pH during DNA synthesis

454 sequencing

The name 454 was a project code name with no known special meaning

14

Shotgun versus amplicon sequencing

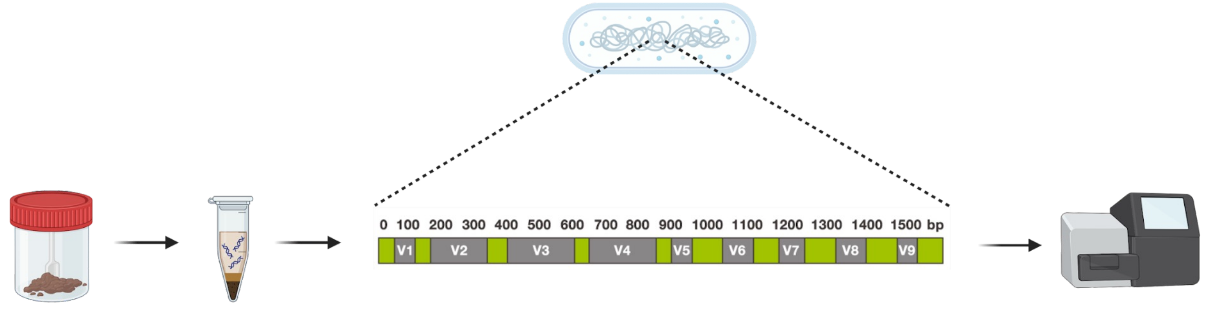


Shotgun sequencing
Minimum amplification

Amplicon sequencing
Maximum amplification

17

Méthodes d'analyses de la métagénomique ciblée


0 100 200 300 400 500 600 700 800 900 1000 1100 1200 1300 1400 1500 bp
V1
V2
V3
V4
V5
V6
V7
V8
V9

Collecte des échantillons

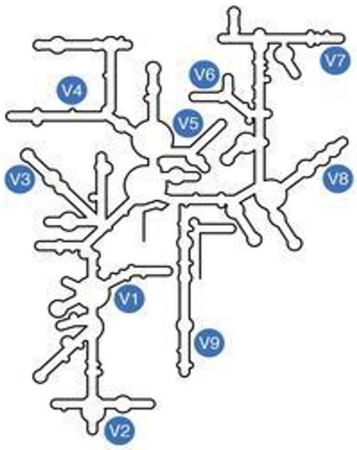
Extraction d'ADN

Amplification d'une région variable du gène de l'ARNr 16S

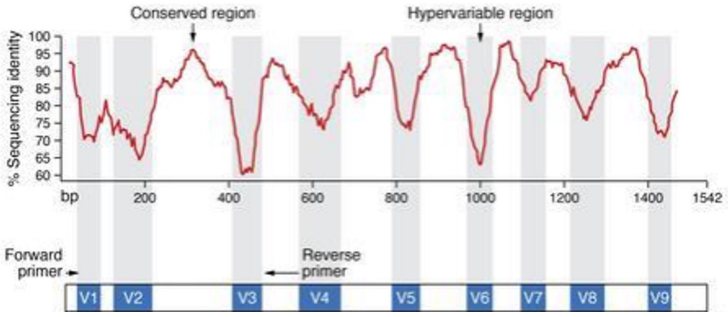
Séquençage des amplicons

18

Analyse de l'amplicon 16S



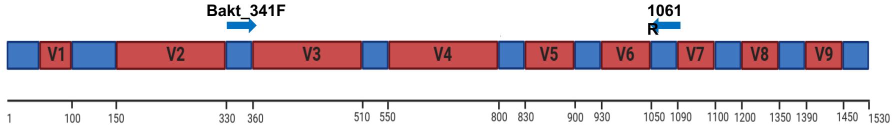
Structure secondaire de l'ARNr 16S d' E. *coli*



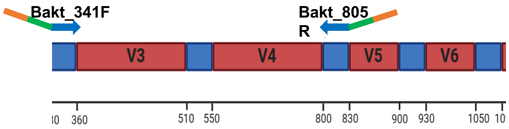
19

PCR amplification strategy

PCR 1:
bakt_341F : 5'- CCTACGGGNGGCWGCAG -3'
1061R : 5'- CRRCACGAGCTGACGAC -3'
bakt_805R : 5'- GACTACHVGGGTATCTAATCC -3'



PCR 2: Nexted PCR with Tagged primers for illumina library creation:



bakt_341F_Eurofins (+341) : 5' ACACTCTTTCCCTACACGAC - GCTCTTCCGATCT - CCTACGGGNGGCWGCAG 3'
bakt_805R_Eurofins (+805) : 5' GACTGGAGTTCAGACGTGT - GCTCTTCCGATCT - GACTACHVGGGTATCTAATCC 3'
Green = complement of sequence primer
Orange = complement of index primers 1 and 2, respectively

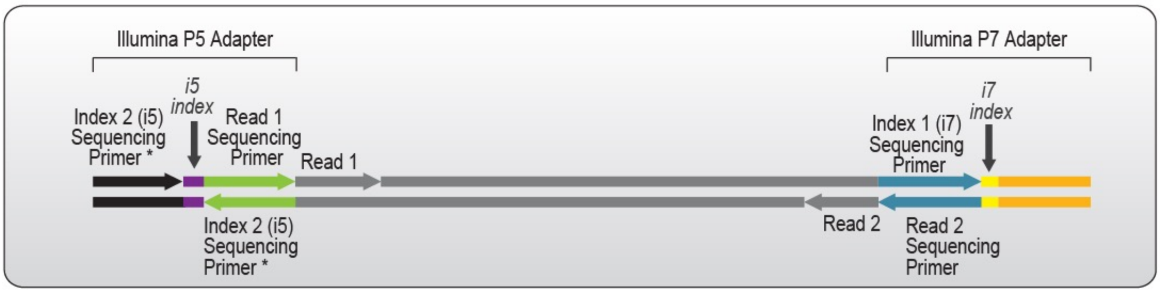
Code	Name	Bases
A	Adenine	A
C	Cytosine	C
G	Guanine	G
T	Thymine (DNA)	T
U	Uracil (RNA)	U
W	Weak	A/T
S	Strong	C/G
M	Amino	A/C
K	Keto	G/T
R	Purine	A/G
Y	Pyrimidine	C/T
B	Not A	C/G/T
D	Not C	A/G/T
H	Not G	A/C/T
V	Not T	A/C/G
N	Any	A/C/G/T

20

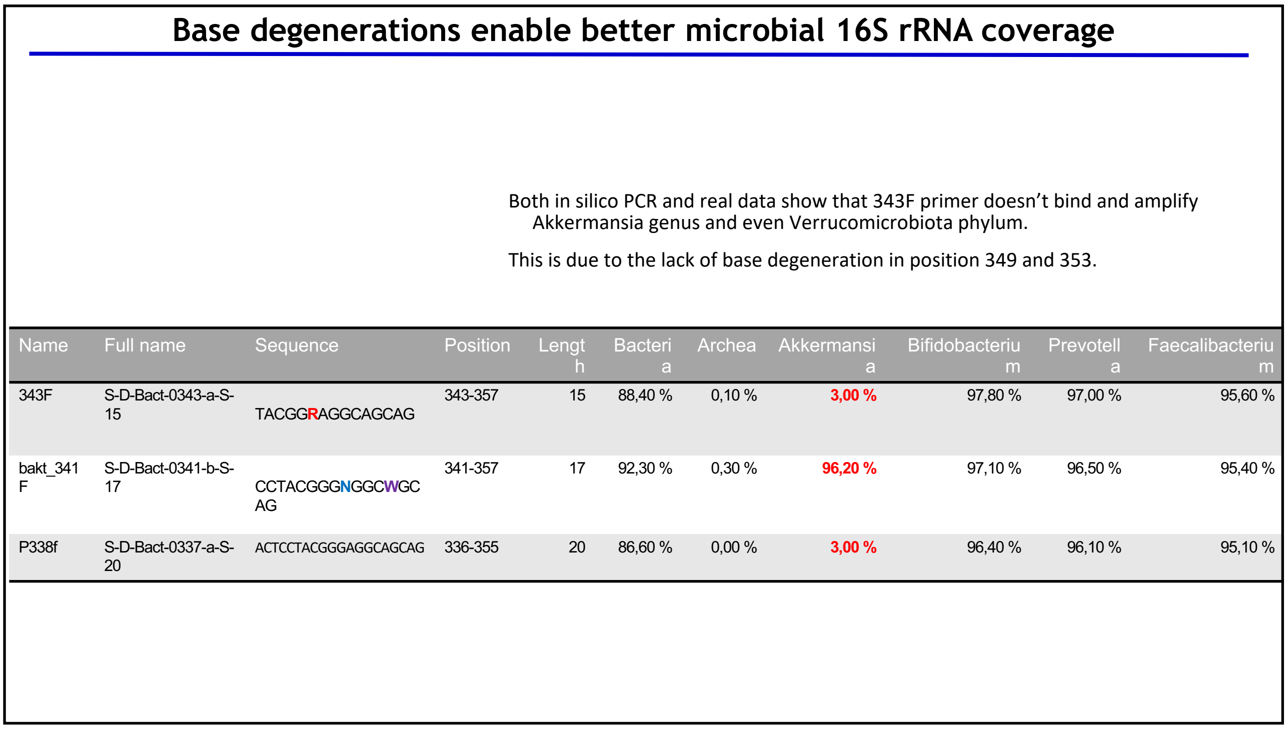
Illumina library creation

PCR 2: Nexted PCR with Tagged primers for illumina library creation:
bakt_341F_Eurofins (+341) : 5' ACACTCTTTCCCTACACGAC - GCTCTTCCGATCT - CCTACGGGNGGCWGCAG 3'
bakt_805R_Eurofins (+805) : 5' GACTGGAGTTCAGACGTGT - GCTCTTCCGATCT - GACTACHVGGGTATCTAATCC 3'
Green = complement of sequence primer
Orange = complement of index primers 1 and 2, respectively

PCR 3: Barcoding



21



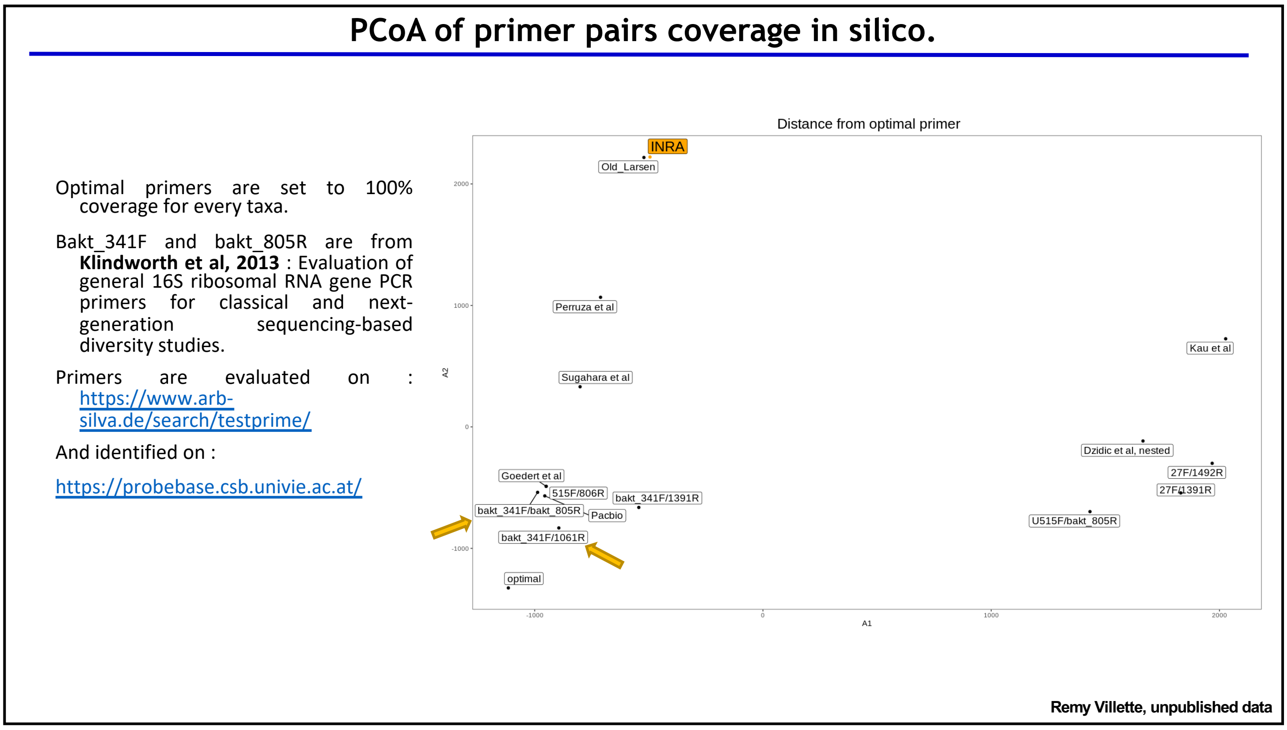
Base degenerations enable better microbial 16S rRNA coverage

Both in silico PCR and real data show that 343F primer doesn’t bind and amplify Akkermansia genus and even Verrucomicrobiota phylum.

This is due to the lack of base degeneration in position 349 and 353.

Name	Full name	Sequence	Position	Length	Bacteria	Archea	Akkermansia	Bifidobacterium	Prevotella	Faecalibacterium
343F	S-D-Bact-0343-a-S-15	TACGGRAGGCAGCAG	343-357	15	88,40 %	0,10 %	3,00 %	97,80 %	97,00 %	95,60 %
bakt_341F	S-D-Bact-0341-b-S-17	CCTACGGGNGGCWGCAG	341-357	17	92,30 %	0,30 %	96,20 %	97,10 %	96,50 %	95,40 %
P338f	S-D-Bact-0337-a-S-20	ACTCCTACGGGAGGCAGCAG	336-355	20	86,60 %	0,00 %	3,00 %	96,40 %	96,10 %	95,10 %

22



PCoA of primer pairs coverage in silico.

Optimal primers are set to 100% coverage for every taxa.

Bakt_341F and bakt_805R are from **Klindworth et al, 2013** : Evaluation of general 16S ribosomal RNA gene PCR primers for classical and next-generation sequencing-based diversity studies.

Primers are evaluated on : https://www.arb-silva.de/search/testprime/

And identified on :

https://probebase.csb.univie.ac.at/

Remy Villette, unpublished data

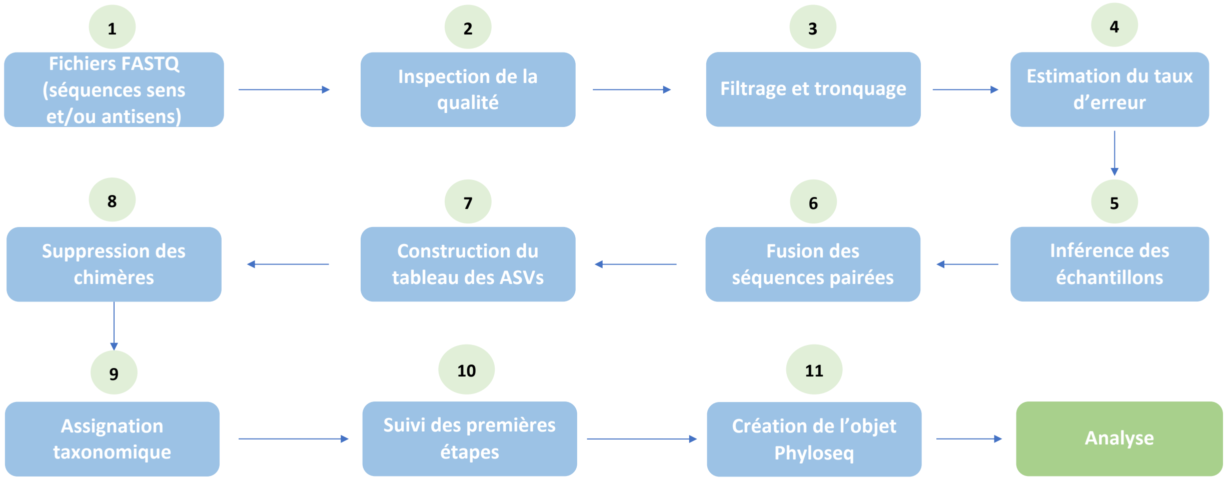
23

Pipeline bio-informatique DADA2

DADA2 = **Divisive Amplicon Denoising Algorithm 2**

- Package R utilisable sur QIIME2 ou directement sur R
- Utilise une inférence statistique pour corriger les erreurs d'amplicon après séquençage
- DADA2 propose des ASV qui ont une resolution plus fines que les OTUs
  - ➢ Discriminations des genres (des fois les especes) genetiquement proches entre elles

Etapes de DADA2



1 Fichiers FASTQ (séquences sens et/ou antisens)

2 Inspection de la qualité

3 Filtrage et tronquage

4 Estimation du taux d'erreur

5 Inférence des échantillons

6 Fusion des séquences pairées

7 Construction du tableau des ASVs

8 Suppression des chimères

9 Assignation taxonomique

10 Suivi des premières étapes

11 Création de l'objet Phyloseq

Analyse

OTU versus ASV

Operational Taxonomic Unit (OTU):

- ➢ An OTU is a way of grouping together sequences of microbial DNA that are similar to each other, typically based on a defined **sequence similarity threshold (e.g., 97% similarity)**.
- ➢ OTUs are used to represent clusters of closely related organisms at a particular taxonomic level.
- ➢ In the context of microbial community analysis, OTUs are often used as a proxy for species or other taxonomic units. The clustering of sequences into OTUs helps simplify the complexity of microbial communities and provides a more manageable unit for analysis.

Amplicon Sequence Variant (ASV):

ASV is a more recent concept that has emerged with the advancement of high-throughput sequencing technologies.
ASVs represent unique, high-resolution sequence variants obtained from the raw sequence data **without the need for clustering.**
ASVs aim to capture the exact biological sequence variants present in a sample.
ASVs are typically identified through methods that consider errors introduced during sequencing and PCR amplification, providing a **more accurate representation of the diversity** within microbial communities.

Fichier FASTQ

FASTQ = **FASTA** + **Q**uality

FASTA

- ○ Un format de fichier texte séquence de nucléotides et d'acides aminés d'acides nucléiques et de protéines
- ○ Un fichier FASTA peut contenir plusieurs séquences
- ○ Extensions du fichier : nomdufichier.fasta

```
>NG_008679.1:5001-38170 Homo sapiens paired box 6 (PAX6)
ACCCTCTTTTCTTATCATTGACATTTAAACTCTGGGGCAGGTCCTCGCGTAGAACGCGGCTGTCAGATCT
GCCACTTCCCCTGCCGAGCGGCGGTGAGAAGTGTGGGAACCGGCGCTGCCAGGCTCACCTGCCTCCCCGC
CCTCCGCTCCCAGGTAACCGCCCGGGCTCCGGCCCCGGCCCGGCTCGGGGCCCGCGGGGCCTCTCCGCTG
CCAGCGACTGCTGTCCCCAAATCAAAGCCCGCCCCAAGTGGCCCCGGGGCTTGATTTTTGCTTTTAAAAG
GAGGCATACAAAGATGGAAGCGAGTTACTGAGGGAGGGATAGGAAGGGGGGTGGAGGAGGGACTTGTCTT
TGCCGAGTGTGCTCTTCTGCAAAAGTAGCAAAATGTTCCACTCCTAAGAGTGGACTTCCAGTCCGGCCCT
GAGCTGGGAGTAGGGGGCGGGAGTCTGCTGCTGCTGTCTGCTAAAGCCACTCGCGACCGCGAAAAATGCA
GGAGGTGGGGACGCACTTTGCATCCAGACCTCCTCTGCATCGCAGTTCACGACATCCACGCTTGGGAAAG
TCCGTACCCGCGCCTGGAGCGCTTAAAGACACCCTGCCGCGGGTCGGGCGAGGTGCAGCAGAAGTTTCCC
GCGGTTGCAAAGTGCAGATGGCTGGACCGCAACAAAGTCTAGAGATGGGGTTCGTTTCTCAGAAAGACGC
```

Quality (Phred Scores)

- ➢ Score déterminant la qualité de chaque paire de base
  - ➢ Probabilité que la base soit séquencée et identifiée correctement

$Q = -10\log_{10}(p)$ $P = 10^{-Q/10}$

Phred Quality Score	Probability of incorrect base call	Base call accuracy
10	1 in 10	90%
20	1 in 100	99%
30	1 in 1000	99.9%
40	1 in 10000	99.99%
50	1 in 100000	99.999%

```
ASCII_BASE=33 Illumina, Ion Torrent, PacBio and Sanger
 Q  P_error  ASCII     Q  P_error  ASCII     Q  P_error  ASCII     Q  P_error  ASCII

 0  1.00000   33 !    11  0.07943   44 ,    22  0.00631   55 7    33  0.00050   66 B
 1  0.79433   34 "    12  0.06310   45 -    23  0.00501   56 8    34  0.00040   67 C
 2  0.63096   35 #    13  0.05012   46 .    24  0.00398   57 9    35  0.00032   68 D
 3  0.50119   36 $    14  0.03981   47 /    25  0.00316   58 :    36  0.00025   69 E
 4  0.39811   37 %    15  0.03162   48 0    26  0.00251   59 ;    37  0.00020   70 F
 5  0.31623   38 &    16  0.02512   49 1    27  0.00200   60 <    38  0.00016   71 G
 6  0.25119   39 '    17  0.01995   50 2    28  0.00158   61 =    39  0.00013   72 H
 7  0.19953   40 (    18  0.01585   51 3    29  0.00126   62 >    40  0.00010   73 I
 8  0.15849   41 )    19  0.01259   52 4    30  0.00100   63 ?    41  0.00008   74 J
 9  0.12589   42 *    20  0.01000   53 5    31  0.00079   64 @    42  0.00006   75 K
10  0.10000   43 +    21  0.00794   54 6    32  0.00063   65 A
```

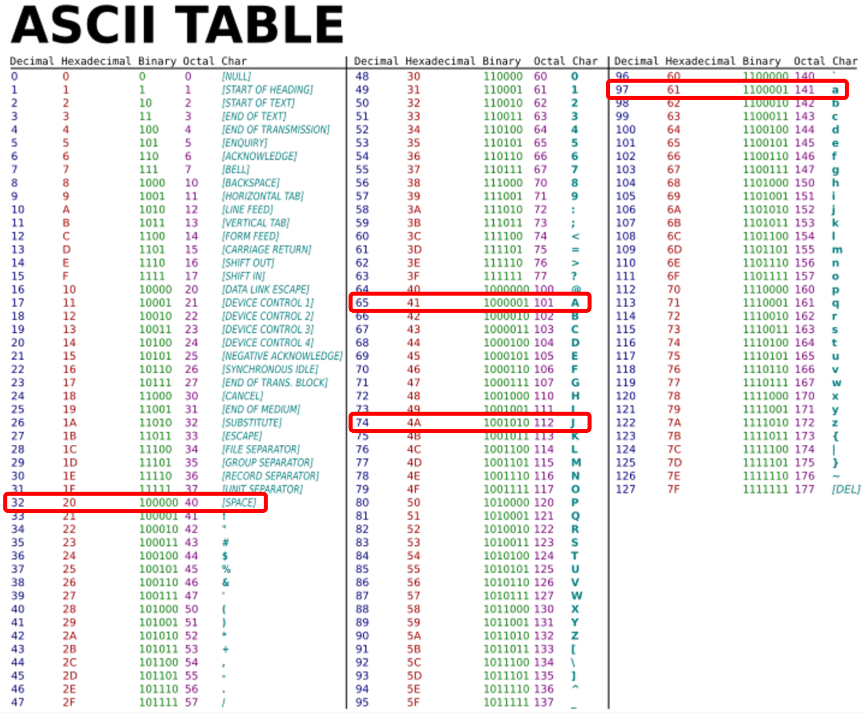
Fichier FASTQ

Identifieur	@MM123:002:FC123AB:3:2208:3330:9840 2:Y:18:ATCACG
FASTA	AGGATACTAGCATAGATACCCTAGATAGTCATAGATCATGATAGGGAGATCTA
Séparateur	+
Scores de Qualité	IJJJJJJIIIIIJIIIIIFFFEEEEEEDDDDDDCABBBBB@@00))))*(*&%!

Fichier FASTQ contient donc :

- **Identifieur** : Informations spécifiques permettant d’identifier la sequence (RunID + barcode)
- **FASTA**
- **Séparateur** : Marque la fin de la sequence
- **Scores de qualité** (Phred quality score):
  +33 encoded, using ASCII characters to represent the numerical quality scores.

29



ASCII TABLE

Decimal	Hexadecimal	Binary	Octal	Char
0	0	0	0	[NULL]
1	1	1	1	[START OF HEADING]
2	2	10	2	[START OF TEXT]
3	3	11	3	[END OF TEXT]
4	4	100	4	[END OF TRANSMISSION]
5	5	101	5	[ENQUIRY]
6	6	110	6	[ACKNOWLEDGE]
7	7	111	7	[BELL]
8	8	1000	10	[BACKSPACE]
9	9	1001	11	[HORIZONTAL TAB]
10	A	1010	12	[LINE FEED]
11	B	1011	13	[VERTICAL TAB]
12	C	1100	14	[FORM FEED]
13	D	1101	15	[CARRIAGE RETURN]
14	E	1110	16	[SHIFT OUT]
15	F	1111	17	[SHIFT IN]
16	10	10000	20	[DATA LINK ESCAPE]
17	11	10001	21	[DEVICE CONTROL 1]
18	12	10010	22	[DEVICE CONTROL 2]
19	13	10011	23	[DEVICE CONTROL 3]
20	14	10100	24	[DEVICE CONTROL 4]
21	15	10101	25	[NEGATIVE ACKNOWLEDGE]
22	16	10110	26	[SYNCHRONOUS IDLE]
23	17	10111	27	[END OF TRANS. BLOCK]
24	18	11000	30	[CANCEL]
25	19	11001	31	[END OF MEDIUM]
26	1A	11010	32	[SUBSTITUTE]
27	1B	11011	33	[ESCAPE]
28	1C	11100	34	[FILE SEPARATOR]
29	1D	11101	35	[GROUP SEPARATOR]
30	1E	11110	36	[RECORD SEPARATOR]
31	1F	11111	37	[UNIT SEPARATOR]
32	20	100000	40	[SPACE]
33	21	100001	41	!
34	22	100010	42	"
35	23	100011	43	#
36	24	100100	44	$
37	25	100101	45	%
38	26	100110	46	&
39	27	100111	47	'
40	28	101000	50	(
41	29	101001	51	)
42	2A	101010	52	*
43	2B	101011	53	+
44	2C	101100	54	,
45	2D	101101	55	-
46	2E	101110	56	.
47	2F	101111	57	/
48	30	110000	60	0
49	31	110001	61	1
50	32	110010	62	2
51	33	110011	63	3
52	34	110100	64	4
53	35	110101	65	5
54	36	110110	66	6
55	37	110111	67	7
56	38	111000	70	8
57	39	111001	71	9
58	3A	111010	72	:
59	3B	111011	73	;
60	3C	111100	74	<
61	3D	111101	75	=
62	3E	111110	76	>
63	3F	111111	77	?
64	40	1000000	100	@
65	41	1000001	101	A
66	42	1000010	102	B
67	43	1000011	103	C
68	44	1000100	104	D
69	45	1000101	105	E
70	46	1000110	106	F
71	47	1000111	107	G
72	48	1001000	110	H
73	49	1001001	111	I
74	4A	1001010	112	J
75	4B	1001011	113	K
76	4C	1001100	114	L
77	4D	1001101	115	M
78	4E	1001110	116	N
79	4F	1001111	117	O
80	50	1010000	120	P
81	51	1010001	121	Q
82	52	1010010	122	R
83	53	1010011	123	S
84	54	1010100	124	T
85	55	1010101	125	U
86	56	1010110	126	V
87	57	1010111	127	W
88	58	1011000	130	X
89	59	1011001	131	Y
90	5A	1011010	132	Z
91	5B	1011011	133	[
92	5C	1011100	134	\
93	5D	1011101	135	]
94	5E	1011110	136	^
95	5F	1011111	137	_
96	60	1100000	140	`
97	61	1100001	141	a
98	62	1100010	142	b
99	63	1100011	143	c
100	64	1100100	144	d
101	65	1100101	145	e
102	66	1100110	146	f
103	67	1100111	147	g
104	68	1101000	150	h
105	69	1101001	151	i
106	6A	1101010	152	j
107	6B	1101011	153	k
108	6C	1101100	154	l
109	6D	1101101	155	m
110	6E	1101110	156	n
111	6F	1101111	157	o
112	70	1110000	160	p
113	71	1110001	161	q
114	72	1110010	162	r
115	73	1110011	163	s
116	74	1110100	164	t
117	75	1110101	165	u
118	76	1110110	166	v
119	77	1110111	167	w
120	78	1111000	170	x
121	79	1111001	171	y
122	7A	1111010	172	z
123	7B	1111011	173	{
124	7C	1111100	174	|
125	7D	1111101	175	}
126	7E	1111110	176	~
127	7F	1111111	177	[DEL]

R ASCII conversion

# Character to Hexadecimal:

```
> charToRaw(" ")
[1] 20
> charToRaw("A")
[1] 41
> charToRaw("J")
[1] 4a (Hexadecimal)
> charToRaw("ab")
[1] 61 62
```

# Hexadecimal to Integer:

```
> strtoi("20", base = 16)
[1] 32
> strtoi("4a", base = 16)
[1] 74
```

# Integer to Hexadecimal:

```
> as.hexmode(74)
[1] ”4a”
```

# Integer to Char

```
> rawToChar(as.raw(as.hexmode(74)))
[1] ”J”
```

# Char to Integer

```
> strtoi(as.character(charToRaw("J")), base=16)
[1] 74
```

30

Fichier FASTQ

Identifieur	@MM123:002:FC123AB:3:2208:3330:9840 2:Y:18:ATCACG
FASTA	AGGATACTAGCATAGATACCCTAGATAGTCATAGATCATGATAGGGAGATCTA
Séparateur	+
Scores de Qualité	IJJJJJJIIIIIJIIIIIFFFEEEEEEDDDDDDCABBBBB@@00))))*(*&%!

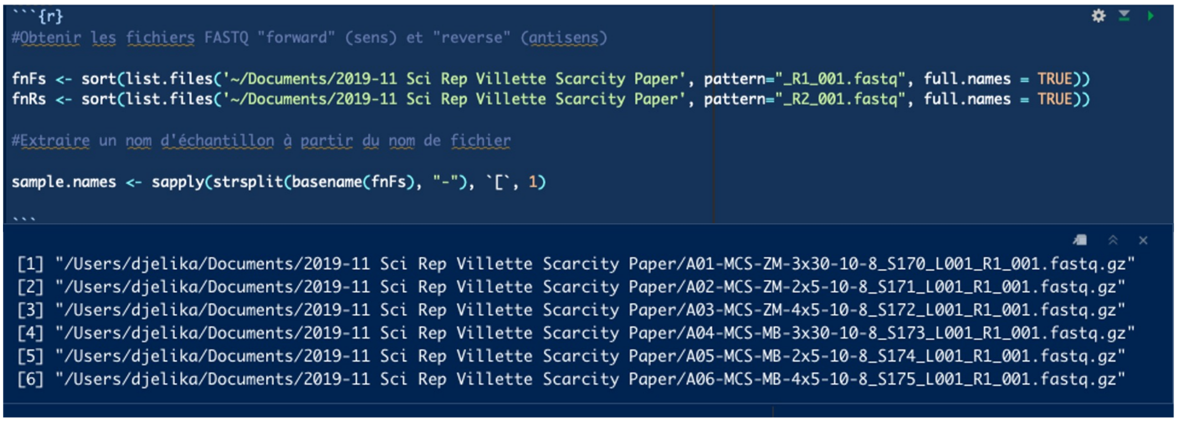
- Scores de qualité (Phred quality score):

+33 encoded, using ASCII characters to represent the numerical quality scores.

```
> strtoi(as.character(charToRaw("IJJJJJJIIIIIJIIIIIFFFEEEEEEDDDDDDCABBBBB@@00))))*(*&%!")), base=16) - 33
 [1] 40 41 41 41 41 41 41 40 40 40 40 40 41 40 40 40 40 40 37 37
[21] 37 36 36 36 36 36 35 35 35 35 35 35 34 32 33 33 33 33 33 31
[41] 31 15 15  8  8  8  8  9  7  9  5  4  0
```

31

1. Importer les séquences FASTQ



~~~
```{r}
#Obtenir les fichiers FASTQ "forward" (sens) et "reverse" (antisens)

fnFs <- sort(list.files('~/Documents/2019-11 Sci Rep Villette Scarcity Paper', pattern="_R1_001.fastq", full.names = TRUE))
fnRs <- sort(list.files('~/Documents/2019-11 Sci Rep Villette Scarcity Paper', pattern="_R2_001.fastq", full.names = TRUE))

#Extraire un nom d'échantillon à partir du nom de fichier

sample.names <- sapply(strsplit(basename(fnFs), "-"), `[`, 1)

```
~~~

```
[1] "/Users/djelika/Documents/2019-11 Sci Rep Villette Scarcity Paper/A01-MCS-ZM-3x30-10-8_S170_L001_R1_001.fastq.gz"
[2] "/Users/djelika/Documents/2019-11 Sci Rep Villette Scarcity Paper/A02-MCS-ZM-2x5-10-8_S171_L001_R1_001.fastq.gz"
[3] "/Users/djelika/Documents/2019-11 Sci Rep Villette Scarcity Paper/A03-MCS-ZM-4x5-10-8_S172_L001_R1_001.fastq.gz"
[4] "/Users/djelika/Documents/2019-11 Sci Rep Villette Scarcity Paper/A04-MCS-MB-3x30-10-8_S173_L001_R1_001.fastq.gz"
[5] "/Users/djelika/Documents/2019-11 Sci Rep Villette Scarcity Paper/A05-MCS-MB-2x5-10-8_S174_L001_R1_001.fastq.gz"
[6] "/Users/djelika/Documents/2019-11 Sci Rep Villette Scarcity Paper/A06-MCS-MB-4x5-10-8_S175_L001_R1_001.fastq.gz"
```

32

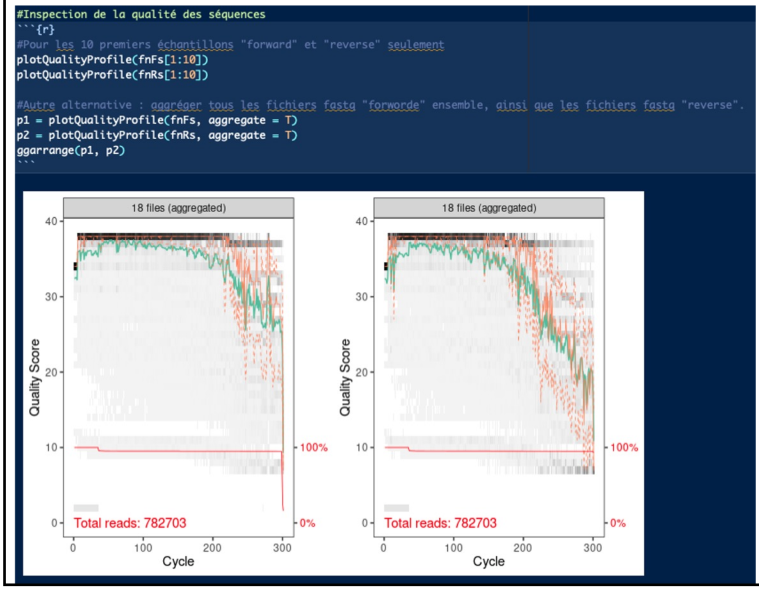
2. Inspection de la qualité

Visualiser la qualité des séquences grâce au Q score associé à chaque nucléotide

33

2. Inspection de la qualité

Visualiser la qualité des séquences grâce au Q score associé à chaque nucléotide



~~~
#Inspection de la qualité des séquences
```{r}
#Pour les 10 premiers échantillons "forward" et "reverse" seulement
plotQualityProfile(fnFs[1:10])
plotQualityProfile(fnRs[1:10])

#Autre alternative : aggréger tous les fichiers fastq "forworde" ensemble, ainsi que les fichiers fastq "reverse".
p1 = plotQualityProfile(fnFs, aggregate = T)
p2 = plotQualityProfile(fnRs, aggregate = T)
ggarrange(p1, p2)
```
~~~

- ○ Ligne verte : médiane
- ○ Lignes oranges pointillés : Quartiles
- ○ L'indice Q :
  - Indique la precision du séquençage
  - Permet de choisir les paramètres de filtrage et tronquage (pour la prochaine étape)

Q score	Precision
10	90 %
20	99 %
30	99.9 %
40	99.99 %

34

Why reverse reads are lower quality than forward reads

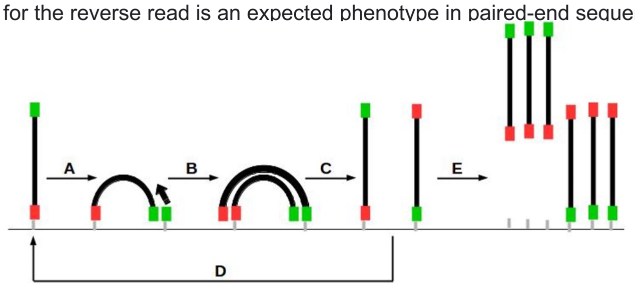
The amplification problem

- The clusters size decreases during bridge amplification at the paired-end turnaround stage.
- Illumina MiSeq does 12 cycles of bridge amplification in order to regenerate the clusters.

- **Result:**
  - A cluster with a smaller amount of molecules and
  - A higher number of errors within these molecules due to more amplification steps, lead to the effect that the per base quality of the read 2 cluster decreases much earlier than for read 1.

- **Consequence:**
  - Low quality: The increased percentage error rate within the (smaller) cluster is now added to the normal 'phasing errors'.

- **Conclusion:** Lower quality for the reverse read is an expected phenotype in paired-end sequencing runs.



35

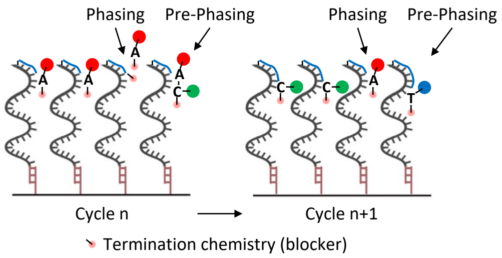
What is phasing error?

The amplification problem

Error types:

- PCR (classic DNA polymerase errors).
  - Deletion, insertion and substitution

- Chemistry:
  - Incomplete deblocking chemistry. (Phasing)
  - Contamination of unblocked bases (Pre-phasing)



- **Phasing:** blocker of a nucleotide is not correctly removed after signal detection =>. no new nucleotide can bind In the next cycle.
  - This DNA fragment will be 1 cycle behind the rest of the fragments in a given cluster (out of phase).

- **Pre-phasing:** a nucleotide lacks the terminator cap => two nucleotides can bind in one cycle.
  - This DNA fragment will be 1 cycle before the rest of the fragments in a given cluster (out of phase).

- All these errors occur with low probability, but **accumulate** for each sequence cycle => **pollute the light signal.**
  - Light signal is used to calculate quality scores.

- Phasing is the main cause of decreasing sequence quality for late cycle base calls.

36

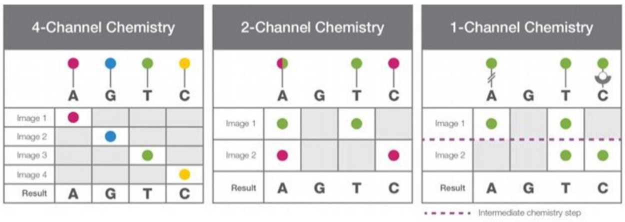
Sequencing by synthesis (SBS) & Base calling

- 4-channel chemistry:
  - Each of the four nucleotides emits a unique wavelength and four images are taken per cycle.
  - The Real-Time Analysis (RTA) software empirically determines the color normalization matrix and calculates phasing/pre-phasing rates, both of which are used in base calling and assigning quality scores.
  - MiSeq, HiSeq 2500, HiSeq 3000/4000 and HiSeq X platforms

- 2-channel chemistry:
  - Two fluorescent dyes and two images determine the incorporation of all four nucleotides per cycle (2 colours combine to 2^2=4).
  - G is represented by absence of color (Dangerous: G's added if amplicons shorter than theoretical sequence read length).
  - Enables faster sequencing and more efficient data processing.
  - MiniSeq, NextSeq 550/550, NextSeq 1000/2000, and the NovaSeq 6000 platforms

- 1-channel chemistry:
  - Each sequencing cycle uses a single fluorescent dye
  - Two chemistry steps
  - Two images (taken after each chemistry step)

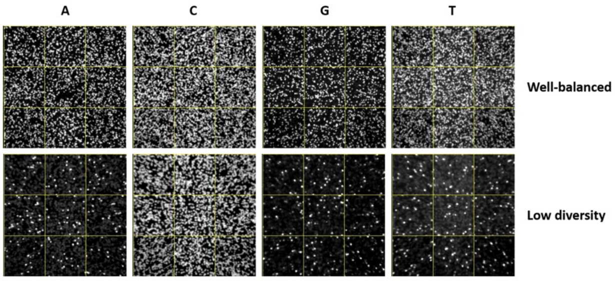
Base calling



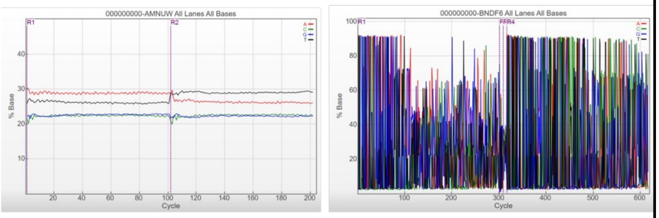
https://emea.illumina.com/science/technology/next-generation-sequencing/sequencing-technology/2-channel-sbs.html

37

Why nucleotide diversity is important?



Well-balanced **Low diversity**



- Nucleotide diversity is particularly important in the first 25 cycles of a sequencing run for calculation of the :
  - clusters passing filter (image recognition of individual sequencing clusters)
  - phasing/pre-phasing (% of molecules within a cluster for which sequencing falls behind (phasing) or jumps ahead (pre-phasing) the current cycle.)
  - colour matrix corrections

- These metrics are then used in base calling and quality score calculations for all cycles in the run.

- Balanced fluorescent signal provides accurate empirical models and improve data quality.

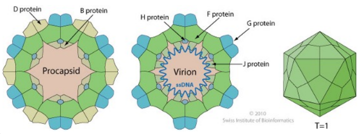
- Template generation (On non-patterned flow cells, the number and location of clusters is empirically determined in the first 4 to 7 cycles).

- Some Illumina platforms with non-patterned flow cells: MiniSeq, MiSeq, NextSeq 500/550, and HiSeq 1000/2500.

https://knowledge.illumina.com/instrumentation/general/instrumentation-general-reference_material-list/000001543

39

How to ensure nucleotide diversity.



- **Phage phiX174 :**
  - Lytic bacteriophage (*Microviridae Sinsheimervirus*)
  - Host: E. coli
  - Isolated from Paris Sewer system by Nicolas Boulgakov (Pasteur institute, 1932)
  - Single stranded genome: 5386 nucleotides (+strand)
    - First DNA genome ever sequenced (Fred Sanger *et al.* Nature 1977 -> Nobel prize 1980)
  - Artificial biology: First virus to be synthesized *in vitro* (Genome: Smith et al PNAS 2003, Complete virus: Cherwa *et al.* JMB 2011)

- **PhiX Control v3 library:**
  - Derived from the PhiX bacteriophage genome
  - Average size of 500 bp
  - Balanced base composition at ~45% GC and ~55% AT.

- **Run Quality Monitor:**
  - Due to its balanced nucleotide composition, the PhiX Control v3 Library is also an ideal sequencing control (typically with ≥ 1% spike-in) for run quality monitoring; e.g., cluster generation, sequencing, and alignment.

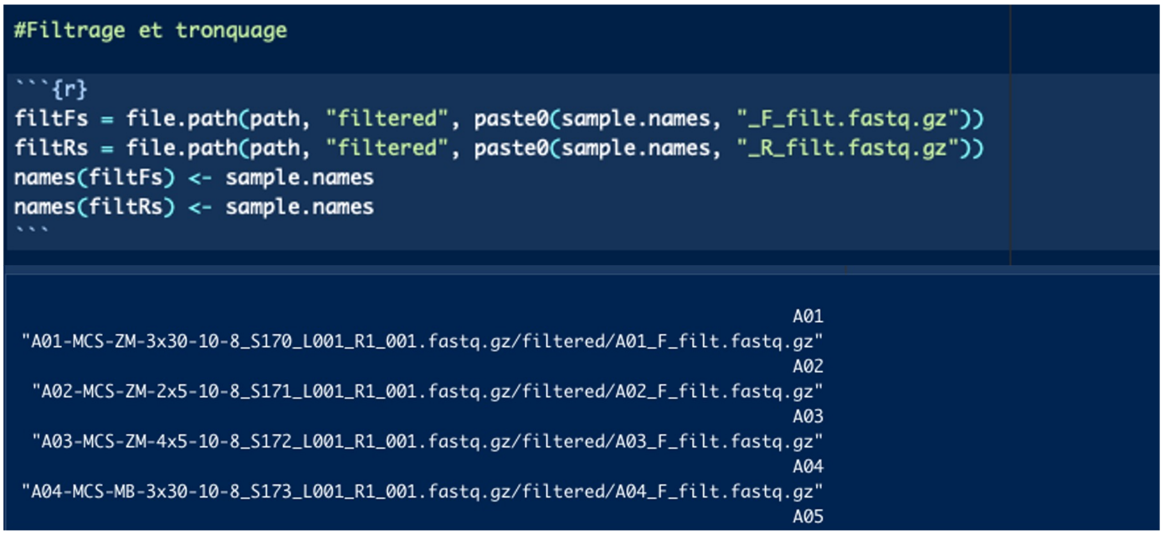
- **Colour Balancing:**
  - For low diversity libraries, the PhiX Control v3 Library provides balanced fluorescent signals at each cycle to improve the overall run quality.

- **16S rRNA gene sequencing:**
  - Considered low diversity
  - Recommendation: Spike in >5% PhiX Control v3 library. (Price: Loss of sequence depth)

https://knowledge.illumina.com/instrumentation/general/instrumentation-general-reference_material-list/000001543

3. Filtrage et tronquage

A. Créer un dossier qui stockera les séquences filtrés

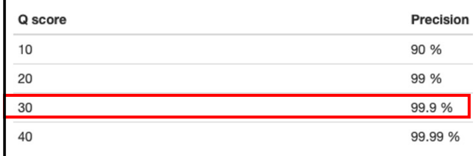


~~~
#Filtrage et tronquage

```{r}
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
```
~~~

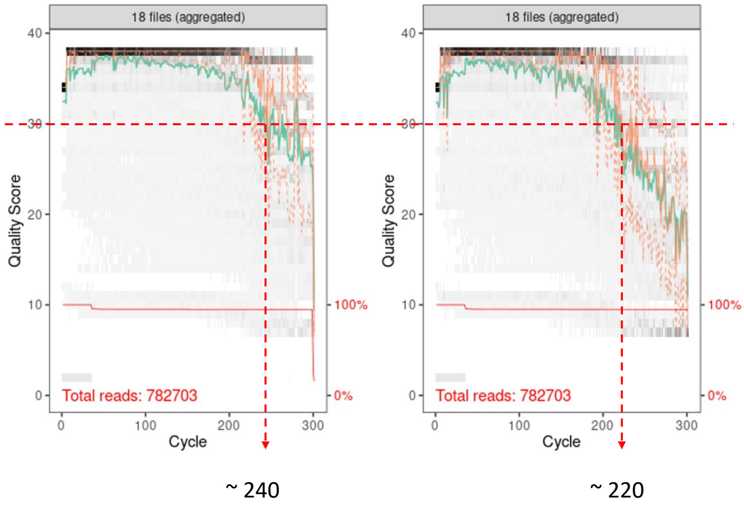
```
                                                                          A01
 "A01-MCS-ZM-3x30-10-8_S170_L001_R1_001.fastq.gz/filtered/A01_F_filt.fastq.gz"
                                                                          A02
  "A02-MCS-ZM-2x5-10-8_S171_L001_R1_001.fastq.gz/filtered/A02_F_filt.fastq.gz"
                                                                          A03
  "A03-MCS-ZM-4x5-10-8_S172_L001_R1_001.fastq.gz/filtered/A03_F_filt.fastq.gz"
                                                                          A04
 "A04-MCS-MB-3x30-10-8_S173_L001_R1_001.fastq.gz/filtered/A04_F_filt.fastq.gz"
                                                                          A05
```

3. Filtrage et tronquage

B. Trouver les meilleurs paramètres de filtrage et tronquage



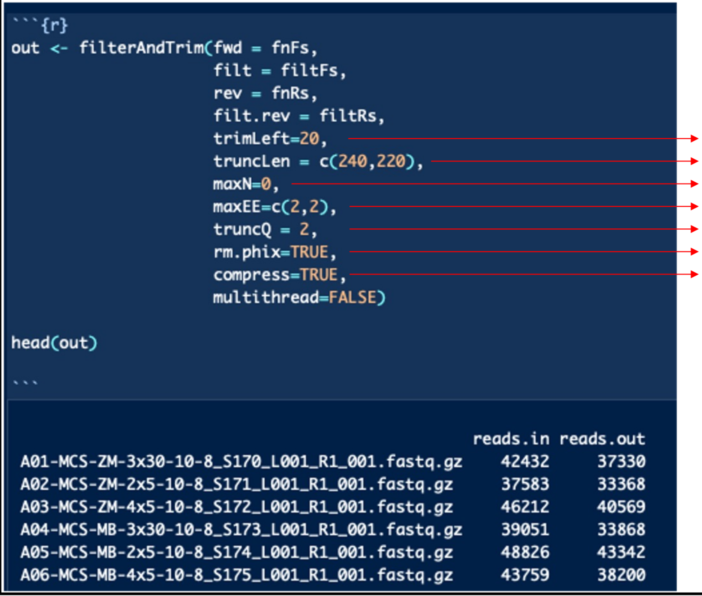
Q score	Precision
10	90 %
20	99 %
30	99.9 %
40	99.99 %



43

3. Filtrage et tronquage

C. Appliquer les paramètres de filtrage et tronquage



~~~
```{r}
out <- filterAndTrim(fwd = fnFs,
                     filt = filtFs,
                     rev = fnRs,
                     filt.rev = filtRs,
                     trimLeft=20,
                     truncLen = c(240,220),
                     maxN=0,
                     maxEE=c(2,2),
                     truncQ = 2,
                     rm.phix=TRUE,
                     compress=TRUE,
                     multithread=FALSE)

head(out)

```
~~~

```
                                              reads.in reads.out
 A01-MCS-ZM-3x30-10-8_S170_L001_R1_001.fastq.gz    42432     37330
 A02-MCS-ZM-2x5-10-8_S171_L001_R1_001.fastq.gz     37583     33368
 A03-MCS-ZM-4x5-10-8_S172_L001_R1_001.fastq.gz     46212     40569
 A04-MCS-MB-3x30-10-8_S173_L001_R1_001.fastq.gz    39051     33868
 A05-MCS-MB-2x5-10-8_S174_L001_R1_001.fastq.gz     48826     43342
 A06-MCS-MB-4x5-10-8_S175_L001_R1_001.fastq.gz     43759     38200
```

Permet d'enlever les amorces

Tronquage de la séquence (élimination des séquences plus courtes)

Nombre maximum de nucléotides « ambigus »

Nombre maximum d' "erreurs attendues" autorisées dans une lecture

Tronque la lecture au premier nucléotide avec un score qualité défini

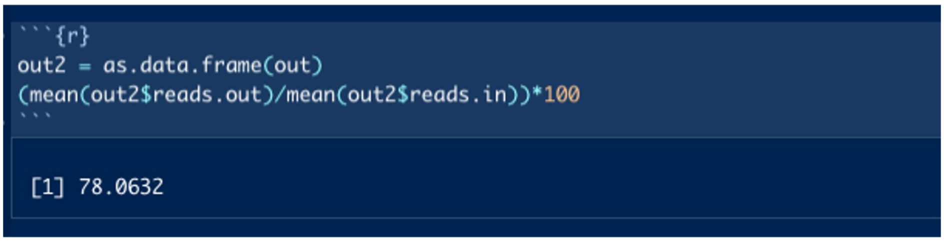
Enlève les séquences non référencées dans la librairie de contrôle (phiX)

Décompression des fichiers

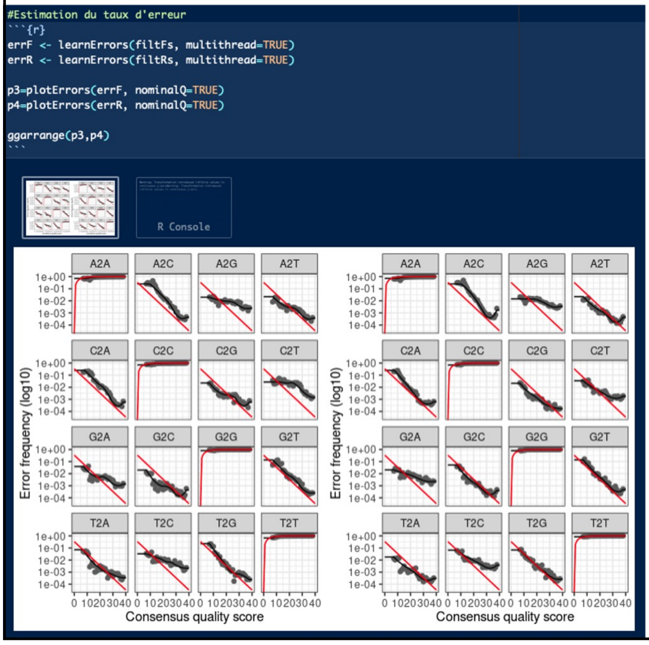
44

3. Filtrage et tronquage

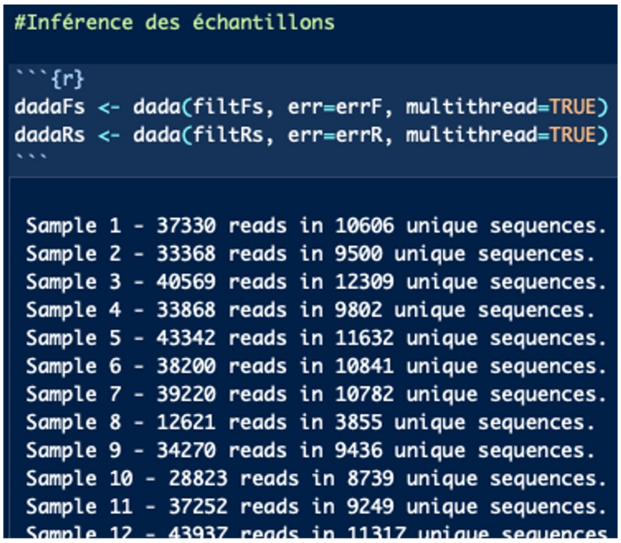
D. Déterminer la pourcentage de sequences ayant passées les étapes de filtrage et tronquage



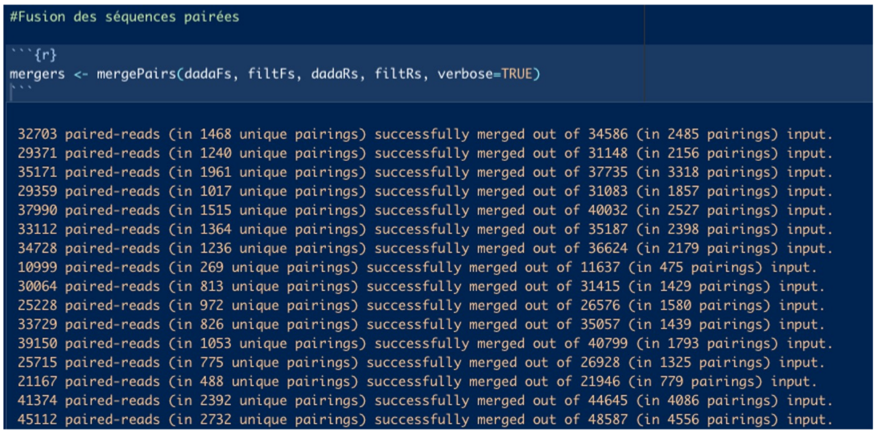
~~~
```{r}
out2 = as.data.frame(out)
(mean(out2$reads.out)/mean(out2$reads.in))*100
~~~

[1] 78.0632
```

~ 17 % des séquences n'ont pas passé les paramètres de filtrage.

45

4. Estimation du taux d'erreur



```
#Estimation du taux d'erreur
```{r}
errF <- learnErrors(filtFs, multithread=TRUE)
errR <- learnErrors(filtRs, multithread=TRUE)

p3=plotErrors(errF, nominalQ=TRUE)
p4=plotErrors(errR, nominalQ=TRUE)

ggarrange(p3,p4)
```
```

- Les taux d'erreur pour chaque transition (A->C, A->G,…) sont affichés.
- Chaque point est un taux d'erreur observé pour chaque score de qualité consensuel.
- La **ligne noire** montre l'erreur après convergence.
- La **ligne rouge** montre l'erreur sous la définition nominale de la valeur Q.
- **Le score de qualité augmente lorsque le taux d'erreur diminue**

47
```

## 5. Inférence des échantillons



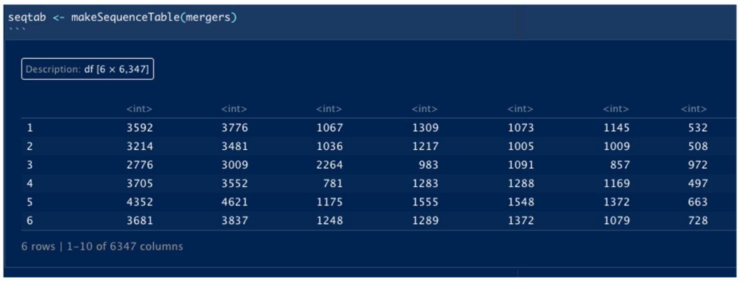
~~~
#Inférence des échantillons

```{r}
dadaFs <- dada(filtFs, err=errF, multithread=TRUE)
dadaRs <- dada(filtRs, err=errR, multithread=TRUE)
~~~

 Sample 1 - 37330 reads in 10606 unique sequences.
 Sample 2 - 33368 reads in 9500 unique sequences.
 Sample 3 - 40569 reads in 12309 unique sequences.
 Sample 4 - 33868 reads in 9802 unique sequences.
 Sample 5 - 43342 reads in 11632 unique sequences.
 Sample 6 - 38200 reads in 10841 unique sequences.
 Sample 7 - 39220 reads in 10782 unique sequences.
 Sample 8 - 12621 reads in 3855 unique sequences.
 Sample 9 - 34270 reads in 9436 unique sequences.
 Sample 10 - 28823 reads in 8739 unique sequences.
 Sample 11 - 37252 reads in 9249 unique sequences.
 Sample 12 - 43937 reads in 11317 unique sequences.
```

- ➢ Utilise le taux d’erreur et les séquences filtrées et tronquées, créés précédemment
- ➢ Permet de confirmer qu’une séquence rencontrés plusieurs fois n’a pas été engendré par des erreurs d’amplifications

48

6. Fusion des séquences pairées



```
#Fusion des séquences pairées

```{r}
mergers <- mergePairs(dadaFs, filtFs, dadaRs, filtRs, verbose=TRUE)
```

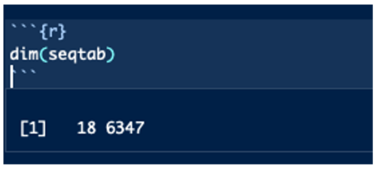
 32703 paired-reads (in 1468 unique pairings) successfully merged out of 34586 (in 2485 pairings) input.
 29371 paired-reads (in 1240 unique pairings) successfully merged out of 31148 (in 2156 pairings) input.
 35171 paired-reads (in 1961 unique pairings) successfully merged out of 37735 (in 3318 pairings) input.
 29359 paired-reads (in 1017 unique pairings) successfully merged out of 31083 (in 1857 pairings) input.
 37990 paired-reads (in 1515 unique pairings) successfully merged out of 40032 (in 2527 pairings) input.
 33112 paired-reads (in 1364 unique pairings) successfully merged out of 35187 (in 2398 pairings) input.
 34728 paired-reads (in 1236 unique pairings) successfully merged out of 36624 (in 2179 pairings) input.
 10999 paired-reads (in 269 unique pairings) successfully merged out of 11637 (in 475 pairings) input.
 30064 paired-reads (in 813 unique pairings) successfully merged out of 31415 (in 1429 pairings) input.
 25228 paired-reads (in 972 unique pairings) successfully merged out of 26576 (in 1580 pairings) input.
 33729 paired-reads (in 826 unique pairings) successfully merged out of 35057 (in 1439 pairings) input.
 39150 paired-reads (in 1053 unique pairings) successfully merged out of 40799 (in 1793 pairings) input.
 25715 paired-reads (in 775 unique pairings) successfully merged out of 26928 (in 1325 pairings) input.
 21167 paired-reads (in 488 unique pairings) successfully merged out of 21946 (in 779 pairings) input.
 41374 paired-reads (in 2392 unique pairings) successfully merged out of 44645 (in 4086 pairings) input.
 45112 paired-reads (in 2732 unique pairings) successfully merged out of 48587 (in 4556 pairings) input.
```

- ➢ Alignement des deux brins uniquement s’ils sont **superposables**
- ➢ Par défaut, les séquences fusionnées ne sont créées que si les lectures sens et antisens se chevauchent d'au moins **12 bases.**
- ➢ Ces bases doivent être identiques les unes aux autres dans la région de chevauchement.

49
```

7. Construction du tableau des Variant de séquence d'amplicon (ASVs)



```
seqtab <- makeSequenceTable(mergers)
```
```
Description: df [6 × 6,347]

         <int>   <int>   <int>   <int>   <int>   <int>   <int>
1         3592    3776    1067    1309    1073    1145     532
2         3214    3481    1036    1217    1005    1009     508
3         2776    3009    2264     983    1091     857     972
4         3705    3552     781    1283    1288    1169     497
5         4352    4621    1175    1555    1548    1372     663
6         3681    3837    1248    1289    1372    1079     728

6 rows | 1-10 of 6347 columns
```

- ➢ Une fois les ASV obtenues, elles sont stockées dans l’objet *seqtab*

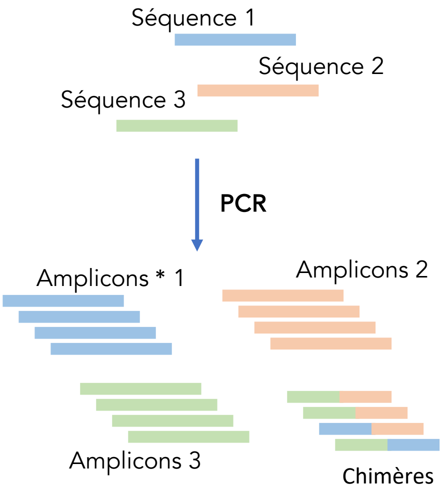


~~~
```{r}
dim(seqtab)
```

 [1]   18 6347
~~~

- ➢ 18 échantillons avec un total de 6347 ASV

50

8. Suppression des chimères

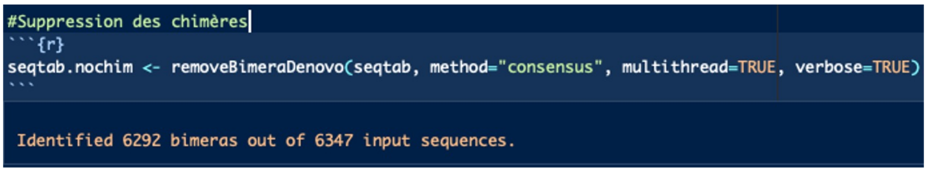


- ➢ Les chimères sont des séquences formées de **deux** ou **plusieurs** séquences réunies :
  - ➢ Des amplicons avec des séquences chimériques peuvent être formés durant la PCR.
  - ➢ Rares durant le séquençage Shotgun mais courantes dans le séquençage d'amplicons – Illumina - (séquences étroitement liées sont amplifiés)

* Morceau d’ADN issu d’une PCR

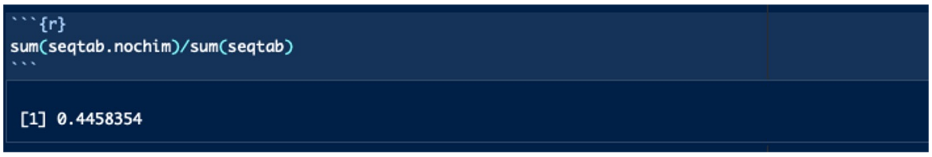
51

8. Suppression des chimères

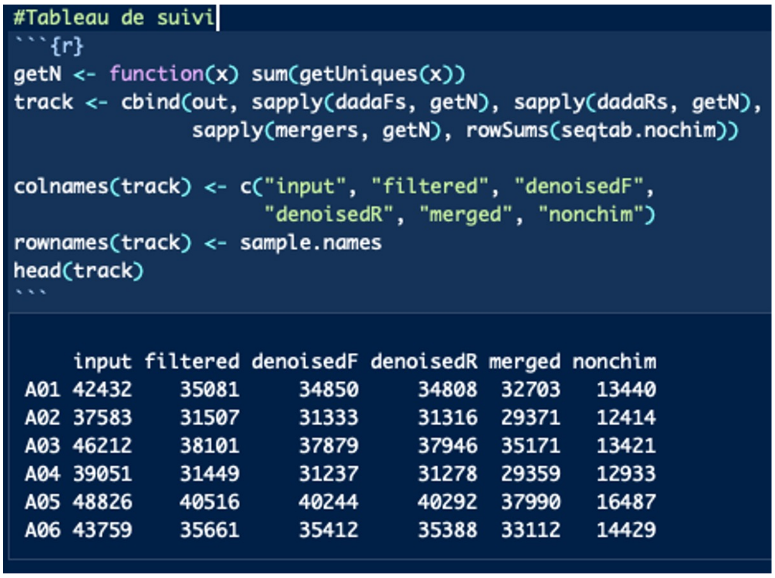


~~~
#Suppression des chimères
```{r}
seqtab.nochim <- removeBimeraDenovo(seqtab, method="consensus", multithread=TRUE, verbose=TRUE)
```

 Identified 6292 bimeras out of 6347 input sequences.
~~~

- 6292 chimères identifiées sur 6347 séquences fusionnées.
  - Pertes importantes de séquences !

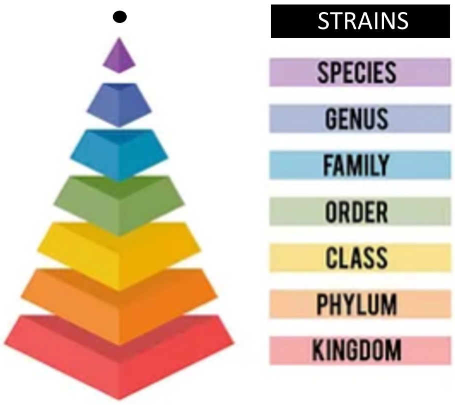


~~~
```{r}
sum(seqtab.nochim)/sum(seqtab)
~~~

 [1] 0.4458354
```

- L’abondance de ces chimères est ~ 56% des lectures de séquences fusionnées

52

9. Suivi des premières étapes



```
#Tableau de suivi
```{r}
getN <- function(x) sum(getUniques(x))
track <- cbind(out, sapply(dadaFs, getN), sapply(dadaRs, getN),
               sapply(mergers, getN), rowSums(seqtab.nochim))

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

    input filtered denoisedF denoisedR merged nonchim
 A01 42432   35081   34850   34808 32703   13440
 A02 37583   31507   31333   31316 29371   12414
 A03 46212   38101   37879   37946 35171   13421
 A04 39051   31449   31237   31278 29359   12933
 A05 48826   40516   40244   40292 37990   16487
 A06 43759   35661   35412   35388 33112   14429
```

Nombre de séquences restantes à chaque étape :

- **Input** : Nombre de séquences totales
- **Filtered** : Nombre de séquences ayant passé l’étape de filtrage et tronquage
- **DenoisedF** : Nombre de séquences sens confirmé
- **DenoisedR** :Nombre de séquences antisens confirmé
- **Merged** : Nombre de séquences mergées
- **Nonchim** : Nombre de séquences après suppression des chimères

53
```

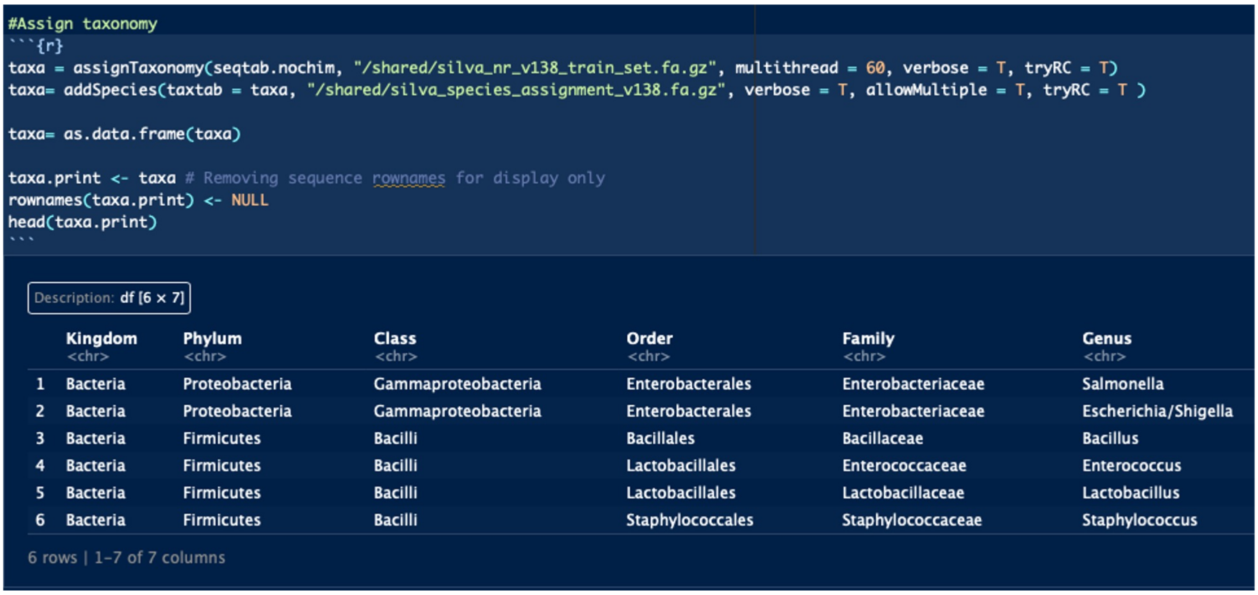
## 10. Assignement taxonomique



- Attributions au niveau **genre** (Genus) et **espèces** (Species) basées sur une correspondance 97%-100% identité entre les ASV et les souches de référence (database Silva).

54

## 10. Assignement taxonomique

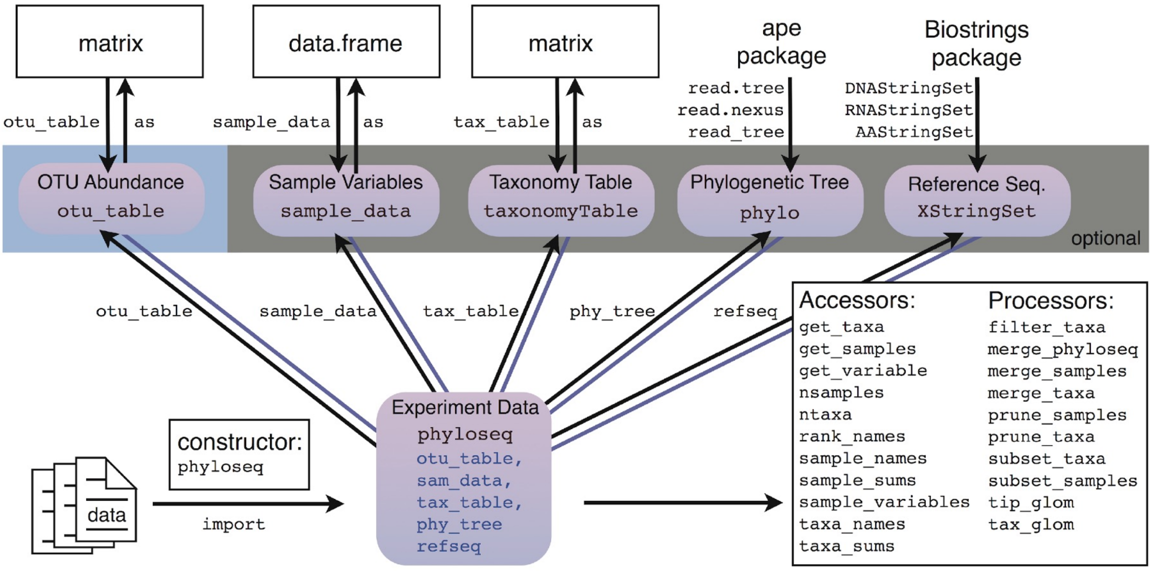


~~~
#Assign taxonomy
```{r}
taxa = assignTaxonomy(seqtab.nochim, "/shared/silva_nr_v138_train_set.fa.gz", multithread = 60, verbose = T, tryRC = T)
taxa= addSpecies(taxtab = taxa, "/shared/silva_species_assignment_v138.fa.gz", verbose = T, allowMultiple = T, tryRC = T )

taxa= as.data.frame(taxa)

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

```
Description: df [6 × 7]

  Kingdom   Phylum          Class                 Order              Family               Genus
  <chr>     <chr>           <chr>                 <chr>              <chr>                <chr>
1 Bacteria  Proteobacteria  Gammaproteobacteria   Enterobacterales   Enterobacteriaceae   Salmonella
2 Bacteria  Proteobacteria  Gammaproteobacteria   Enterobacterales   Enterobacteriaceae   Escherichia/Shigella
3 Bacteria  Firmicutes      Bacilli               Bacillales         Bacillaceae          Bacillus
4 Bacteria  Firmicutes      Bacilli               Lactobacillales    Enterococcaceae      Enterococcus
5 Bacteria  Firmicutes      Bacilli               Lactobacillales    Lactobacillaceae     Lactobacillus
6 Bacteria  Firmicutes      Bacilli               Staphylococcales   Staphylococcaceae    Staphylococcus

6 rows | 1-7 of 7 columns
```

55

11. Création de l’objet Phyloseq



matrix

data.frame

matrix

ape package

Biostrings package

otu_table as

sample_data as

tax_table as

read.tree
read.nexus
read_tree

DNAStringSet
RNAStringSet
AAStringSet

OTU Abundance otu_table

Sample Variables sample_data

Taxonomy Table taxonomyTable

Phylogenetic Tree phylo

Reference Seq. XStringSet

optional

otu_table

sample_data

tax_table

phy_tree

refseq

data

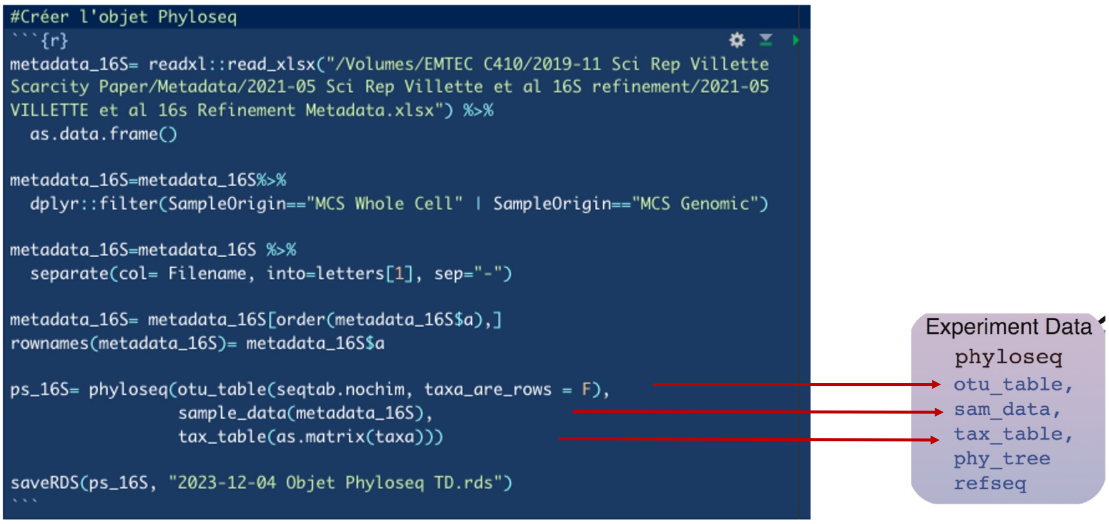
constructor: phyloseq

import

Experiment Data
phyloseq
otu_table,
sam_data,
tax_table,
phy_tree
refseq

Accessors:	Processors:
get_taxa	filter_taxa
get_samples	merge_phyloseq
get_variable	merge_samples
nsamples	merge_taxa
ntaxa	prune_samples
rank_names	prune_taxa
sample_names	subset_taxa
sample_sums	subset_samples
sample_variables	tip_glom
taxa_names	tax_glom
taxa_sums	

11. Création de l’objet Phyloseq



~~~
#Créer l'objet Phyloseq
```{r}
metadata_16S= readxl::read_xlsx("/Volumes/EMTEC C410/2019-11 Sci Rep Villette
Scarcity Paper/Metadata/2021-05 Sci Rep Villette et al 16S refinement/2021-05
VILLETTE et al 16s Refinement Metadata.xlsx") %>%
  as.data.frame()

metadata_16S=metadata_16S%>%
  dplyr::filter(SampleOrigin=="MCS Whole Cell" | SampleOrigin=="MCS Genomic")

metadata_16S=metadata_16S %>%
  separate(col= Filename, into=letters[1], sep="-")

metadata_16S= metadata_16S[order(metadata_16S$a),]
rownames(metadata_16S)= metadata_16S$a

ps_16S= phyloseq(otu_table(seqtab.nochim, taxa_are_rows = F),
                sample_data(metadata_16S),
                tax_table(as.matrix(taxa)))

saveRDS(ps_16S, "2023-12-04 Objet Phyloseq TD.rds")
```
~~~

Experiment Data
phyloseq
otu_table,
sam_data,
tax_table,
phy_tree
refseq