Compare epi2me and MetONTIIME results of 4 InSilicoPCR classifications

Stephane Plaisance

Report created: 2019-10-23

Aim:

The motivation behind this analysis was to simulate 16S long amplicon sequencing on ONT platform using data from the **mockcommunity** site as a large Promethion fastq archive obtained after sequencing of ONT fastq data obtained after sequencing the ZymoBIOMICS™ Microbial Community Standard **link**.

Table 1: Microbial Composition

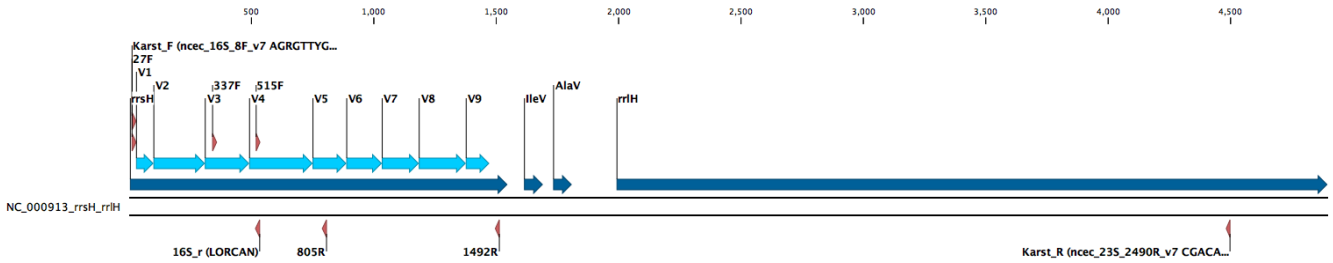
Species	Theoretical Composition (%)				
	Genomic DNA	16S Only[1]	16S & 18S[1]	Genome Copy[2]	Cell Number[3]
Pseudomonas aeruginosa	12	4.2	3.6	6.1	6.1
Escherichia coli	12	10.1	8.9	8.5	8.5
Salmonella enterica	12	10.4	9.1	8.7	8.8
Lactobacillus fermentum	12	18.4	16.1	21.6	21.9
Enterococcus faecalis	12	9.9	8.7	14.6	14.6
Staphylococcus aureus	12	15.5	13.6	15.2	15.3
Listeria monocytogenes	12	14.1	12.4	13.9	13.9
Bacillus subtilis	12	17.4	15.3	10.3	10.3
Saccharomyces cerevisiae	2	NA	9.3	0.57	0.29
Cryptococcus neoformans	2	NA	3.3	0.37	0.18

[1] The theoretical composition in terms of 16S (or 16S & 18S) rRNA gene abundance was calculated from theoretical genomic DNA composition with the following formula: 16S/18S copy number = total genomic DNA (g) × unit conversion constant (bp/g) / genome size (bp) × 16S/18S copy number per genome. Use this as reference when performing 16S targeted sequencing.

[2] The theoretical composition in terms of genome copy number was calculated from theoretical genomic DNA composition with the following formula: genome copy number = total genomic DNA (g) × unit conversion constant (bp/g) / genome size (bp). Use this as reference when inferring microbial abundance from shotgun sequencing data based on read depth/coverage.

[3] The theoretical composition in terms of cell number was calculated from theoretical genomic DNA composition with the following formula: cell number = total genomic DNA (g) × unit conversion constant (bp/g) / genome size (bp)/ploidy.

Four different primer sets were selected in order to extract different regions of the bacterial genomes. The obtained reads were submitted to the online **ONT epi2me** classification tool to identify metagenomic signatures and the final result file was downloaded from the ONT virtual machine.



27F-U1492R in-silico amplicon (~1.4kb)

- 27F: "AGAGTTTGATCMTGGCTCAG"
- 1492Rw: "CGGTWACCTTGTTACGACTT"

337F-805R in-silico amplicon (~400bps)

- 337F: "GACTCCTACGGGAGGCWGCAG"
- 805R: "GACTACHVGGGTATCTAATCC"

515FB-U1492Rw in-silico amplicon (~850bps)

- 515FB: "GTGYCAGCMGCCGCGGTAA"
- U1492Rw: "CGGTWACCTTGTTACGACTT"

16S_8F_23S_2490R_v7 in-silico amplicon (~4400bps)

- 16S_8F:AGRGTTYGATYMTGGCTCAG
- 23S_2490R:CGACATCGAGGTGCCAAAC

The four classification files were further analysed in **R** to count metagenomes at the level of family, genus and or species and to create summary tables and plots.

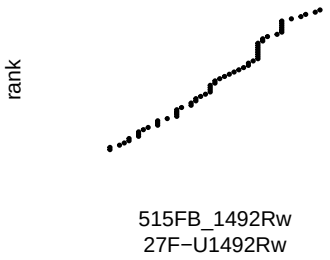
Finally, the summary tables were sorted by decreasing feature counts at each level and compared to correlate the different amplicon classifications.

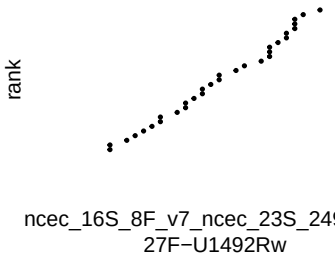
Results

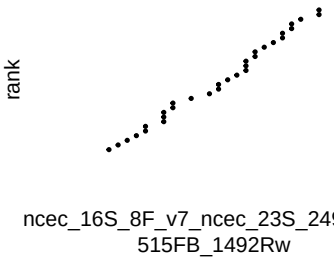
Epi2me analysis of the four datasets

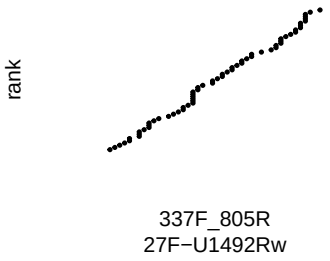
epi2me classification comparison at genus level

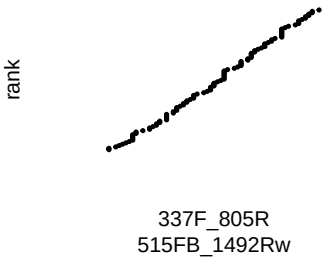
The list of genus reported by epi2me and sorted by decreasing abundance are compared in pairs to count how many of the same hits are reported in the two runs at the same level of each list. The comparison stops at the end of the shortest list of each pair. A perfect diagonal indicates identical order between two files (eg comparing a file with itself). As seen, most classifications return about the same genus, sometimes close counts rank few results in slightly different orders but all genus are finally returned in all pairs.

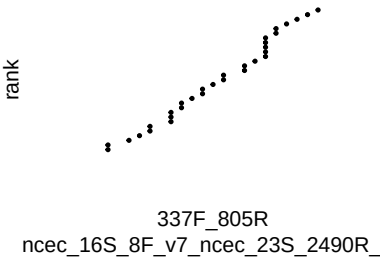








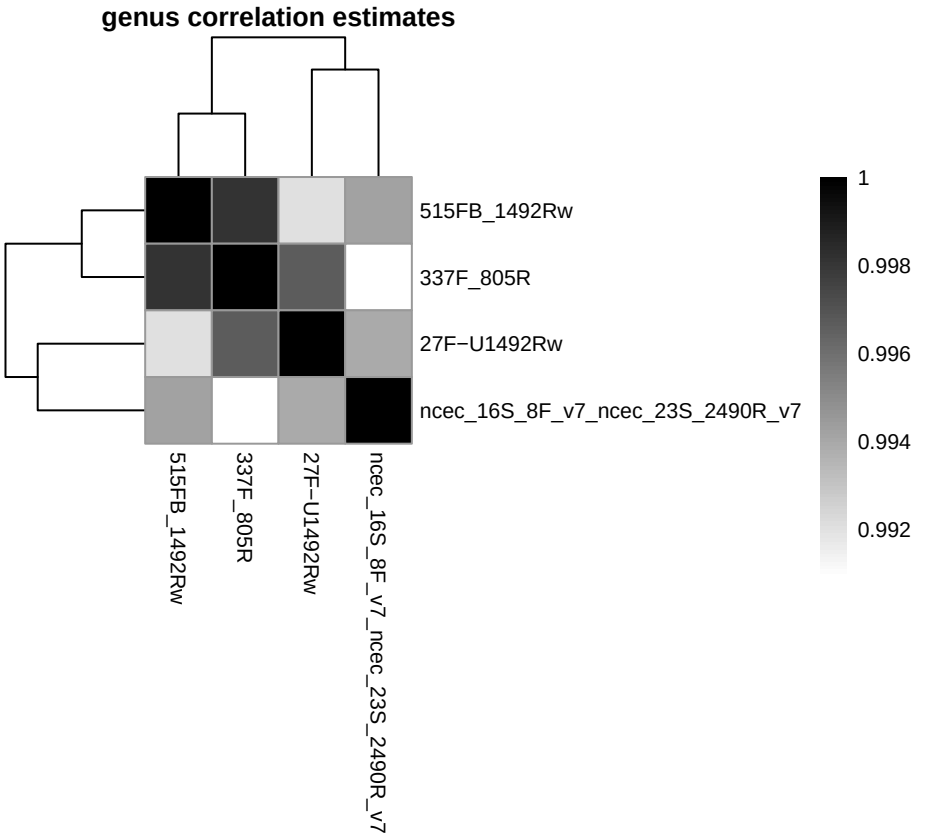




Pairwise genus correlation estimate matrix

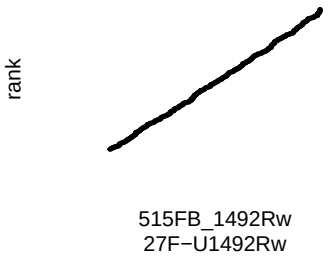
The correlation between pairs is computed and the returned estimates are used to plot a heatmap illustrating the similarity between the four amplicon results at genus level.

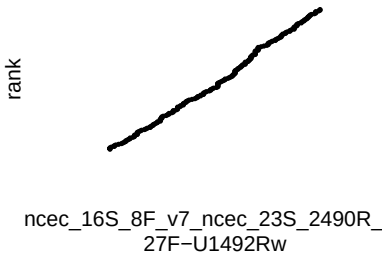
	27F-U1492Rw	515FB_1492Rw	ncec_16S_8F_v7_ncec_23S_2490R_v7	337F_805R
27F-U1492Rw	1.000	0.992	0.994	0.997
515FB_1492Rw	0.992	1.000	0.994	0.998
ncec_16S_8F_v7_ncec_23S_2490R_v7	0.994	0.994	1.000	0.991
337F_805R	0.997	0.998	0.991	1.000

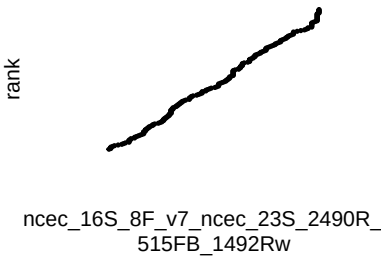


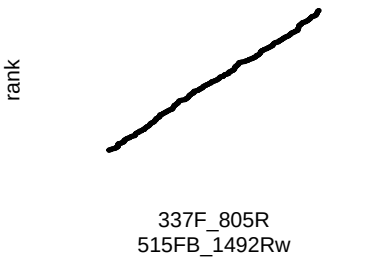
epi2me classification comparison at species level

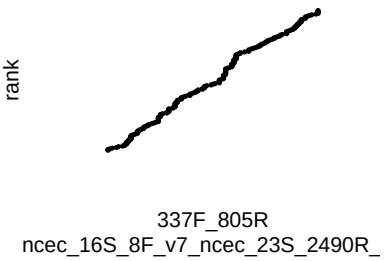
The list of species reported by epi2me and sorted by decreasing abundance are compared in pairs to count how many of the same hits are reported in the two runs at the same level of each list. The comparison stops at the end of the shortest list of each pair. A perfect diagonal indicates identical order between two files (eg comparing a file with itself). As seen, most classifications return about the same species, sometimes close counts rank few results in slightly different orders but all top species are finally returned in all pairs.







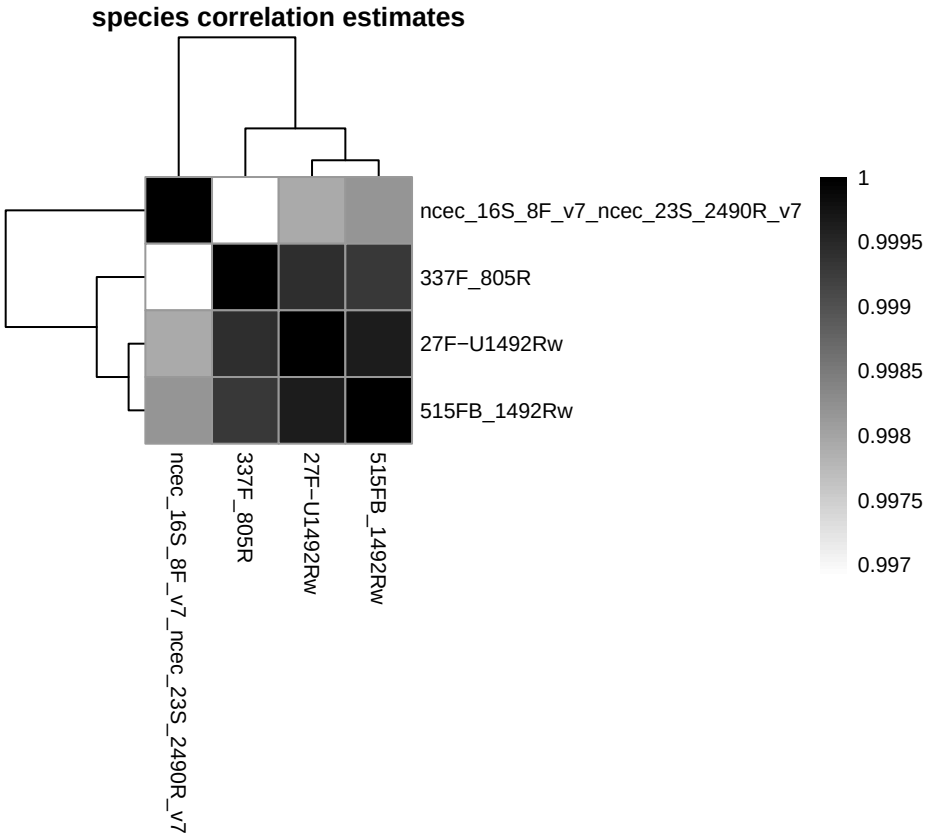




Pairwise species correlation estimate matrix

The correlation between pairs is computed and the returned estimates are used to plot a heatmap illustrating the similarity between the four amplicon results at species level.

	27F-U1492Rw	515FB_1492Rw	ncec_16S_8F_v7_ncec_23S_2490R_v7	337F_805R
27F-U1492Rw	1.000	1.000	0.998	0.999
515FB_1492Rw	1.000	1.000	0.998	0.999
ncec_16S_8F_v7_ncec_23S_2490R_v7	0.998	0.998	1.000	0.997
337F_805R	0.999	0.999	0.997	1.000



MetONTIIME analysis of the four datasets

MetONTIIME was recently developed to allow users analyze their data outside of the ONT black-box. The tool is using **Qiime2** and is rather slow, reason why we only analyzed the first 10-15k reads of each sample.

MetONTIIME generate results with long labels parsing all 7 taxonomy levels which lead to hard to read plots. We therefore did some reformatting of the data in order to limit the labels to "genus" or "genus;species" as shown below. The code used to simplify the taxonomy plot files is available on our **GIT repo**

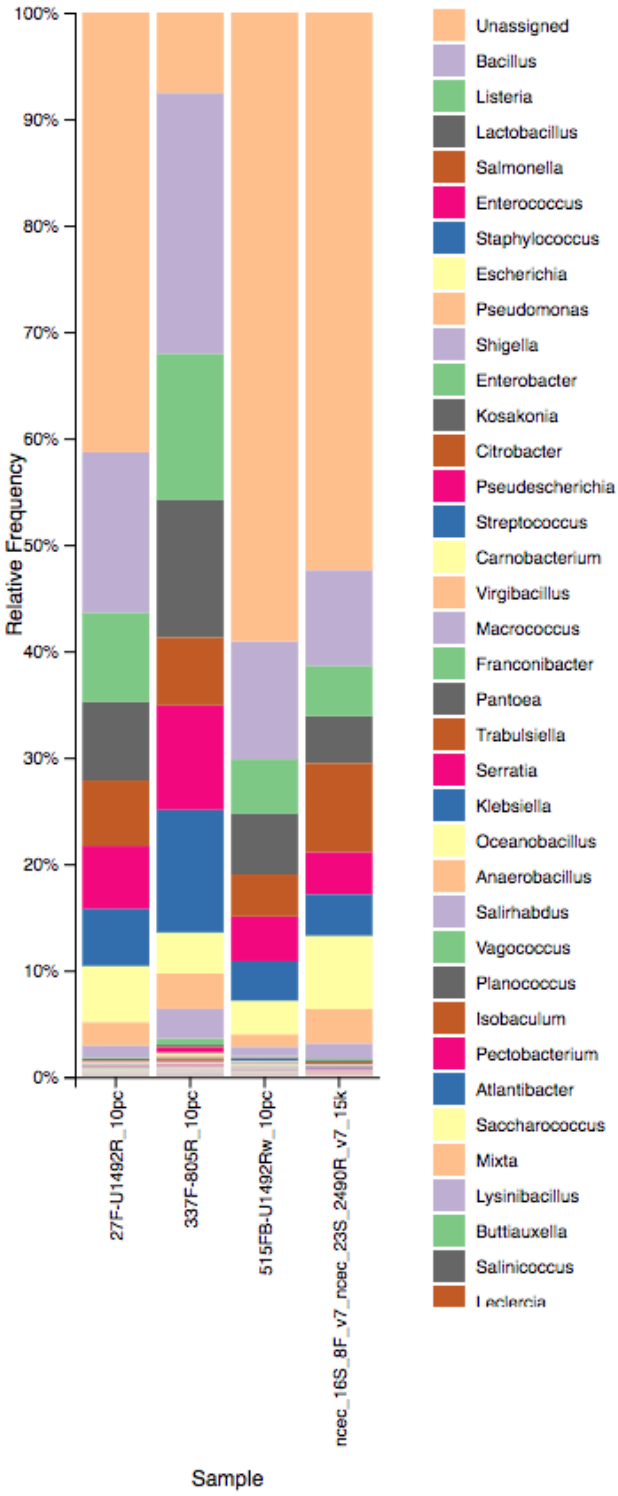
The final results saved as a *qsv* file were used to produce plots with the online **Qiime2 viewer**

Analysis against the NCBI database

The analysis was done using the NCBI database (*PRJNA33175*) as reference as described in the MetONTIIME documentation, similarly to what ONT epi2me does.

```
# NCBI PRJNA33175 run
./MetONTIIME.sh /data2/analyses/MetONTIIME \
  /data2/analyses/MetONTIIME/sample-metadata.tsv \
  /opt/biotools/MetONTIIME/PRJNA33175_Bacterial_sequences_sequence.qza \
  /opt/biotools/MetONTIIME/PRJNA33175_Bacterial_sequences_taxonomy.qza \
  84
```

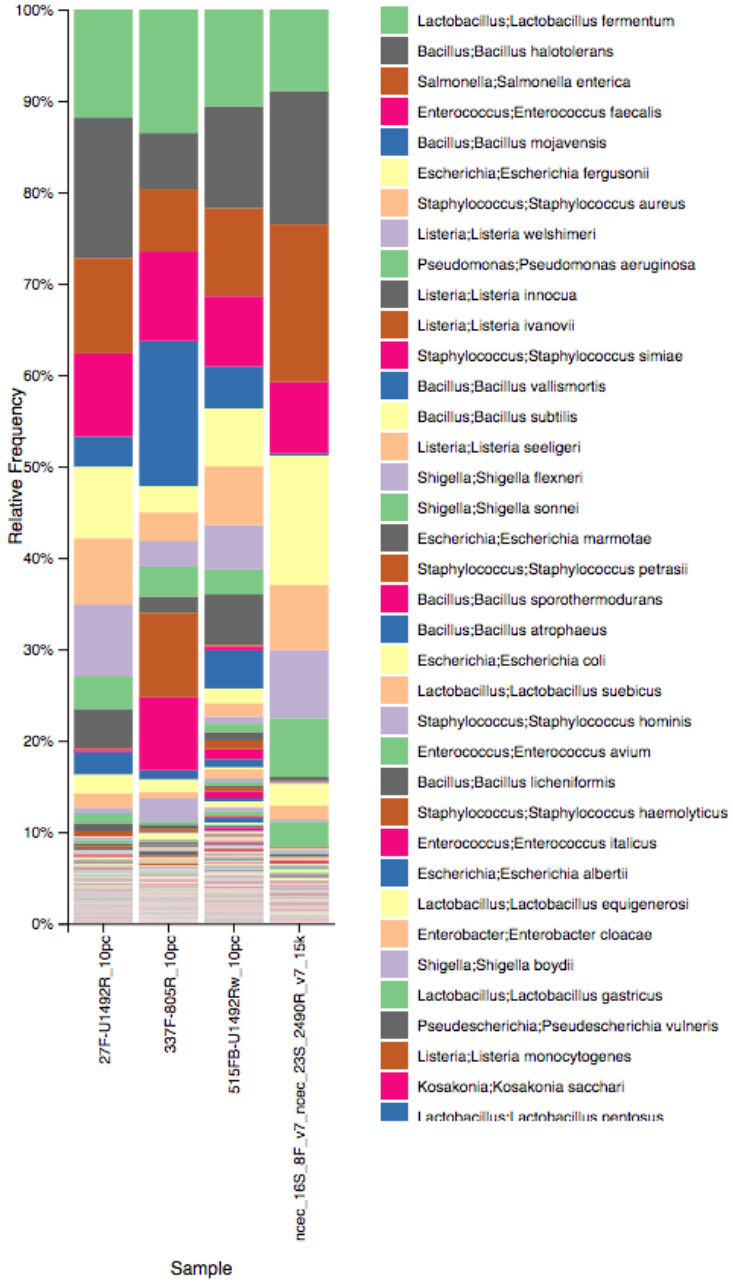
The first figure reports the classification of the four samples at **genus** level with unassigned reads.



The proportion of unassigned reads varies largely between amplicons, suggesting that other loci are extracted from the genomic data due to sequence similarity with one or both primers as already seen in the epi2me analysis. We did not investigate what the unassigned sequences really are. and whether they cluster in specific regions of genomes as this falls beyond the topic of this study.

The next figure reports successful classification for the four samples at **species** level (after filtering out

unassigned reads).



The order and relative abundance of the top species is quite close between amplicons but different enough to support not mixing up different amplicons in a comparison analysis of metagenomes when a quantitative analysis is required (not unexpected).

Analysis against the SILVA database

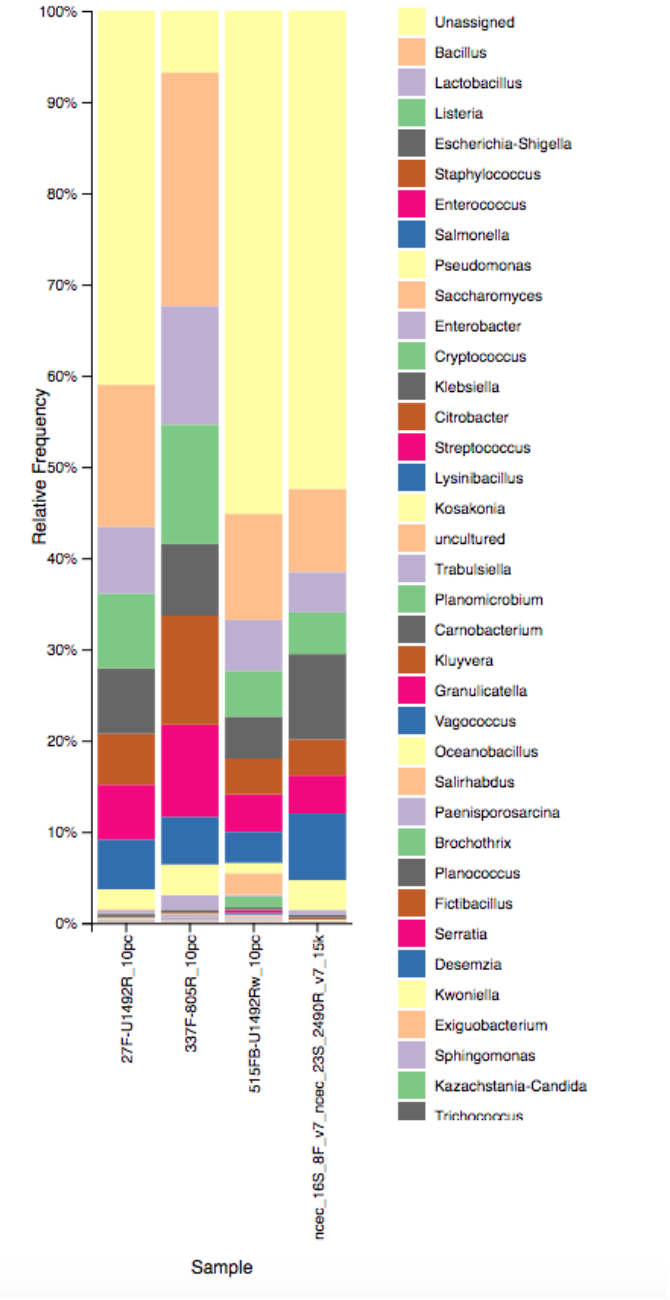
The analysis was done here using the **SILVA build 132 database (*PRJNA33175*) with commands described in the MetONTIIME documentation.

```
# from : https://forum.qiime2.org/t/nanopore-reads-analysis-using-qiime2/11364/2

wget https://www.arb-silva.de/fileadmin/silva_databases/qiime/Silva_132_release.zip
unzip Silva_132_release.zip

identity=99
input="all"
type=""

qiime tools import \
    --type FeatureData[Sequence] \
    --input-path SILVA_132_QIIME_release/rep_set/rep_set_${input}/${identity}/silva132_${identity}${type}.fna \
    --output-path silva_132_${identity}${type}_sequence.qza

qiime tools import \
    --type FeatureData[Taxonomy] \
    --input-path  SILVA_132_QIIME_release/taxonomy/taxonomy_${input}/${identity}/taxonomy_7_levels.txt \
    --input-format HeaderlessTSVTaxonomyFormat \
    --output-path silva_132_${identity}${type}_taxonomy.qza
```

The resulting artifacts were used during the MetONTIIME run to classify the same read subsets as above.

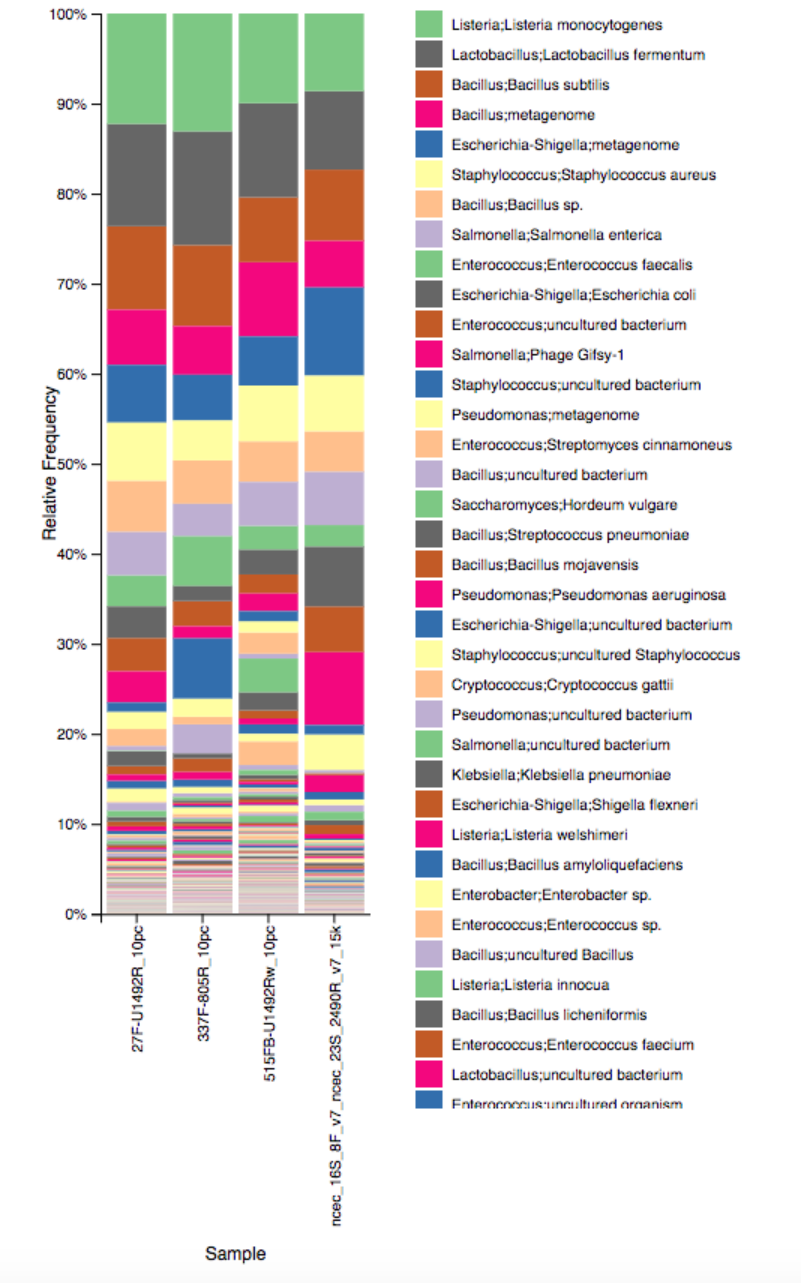
```
# silva_132_99 run
./MetONTIIME.sh /data2/analyses/MetONTIIME_4smpl_silva \
  /data2/analyses/MetONTIIME_4smpl_silva/sample-metadata.tsv \
  /data/biodata/MetONTIIME_DB/silva_132_99_sequence.qza \
  /data/biodata/MetONTIIME_DB/silva_132_99_taxonomy.qza \
  84
```

The first figure reports the classification of the four samples at **species** level with unassigned reads.



The proportion of unassigned reads varies largely between amplicons, suggesting that other loci are extracted from the genomic data due to sequence similarity with one or both primers as already seen in the epi2me analysis. We did not investigate what the unassigned sequences really are. and whether they cluster in specific regions of genomes as this falls beyond the topic of this study.

The next figure reports successful classification for the four samples at **species** level (after filtering out unassigned reads).



Note: When using **Silva**, the **Escherichia genus** is detected, in contrast to the results obtained with NCBI references. The Coli species is not reported alone probably because of a high sequence similarity between different Escherichia species in Silva.

Analysis against the rrnDB database

Another public database (**rrnDB**) was derived from the NCBI RefSeq collection (published in 2015) and contains genomic regions corresponding to the full 16S_ITS_18S locus from a large number of bacteria. We obtained this data and corresponding accession numbers from the **github repository**. The data was re-annotated using the MetONTIIME workflow and NCBI taxonomy dump data. The resulting two Qiime2 artifacts were then used to analyze the same data samples as before.

```
wget https://github.com/alfbenpa/rrn_db/blob/master/operon.100.fa.tar.gz
tar -xzvf operon.100.fa.tar.gz

wget https://github.com/alfbenpa/rrn_db/blob/master/species_annotation

# rename fasta headers with rrn2ncbi.R
Rscript rrn2ncbi.R operons.100.fa species_annotation operons.100_gb_names.fa
```

The **rrn2ncbi.R** script next was used to rename the fasta sequences based on the accompanying annotation file

```
# prepare rrnaDB data for MetONTIIME
# source activate MetONTIIME_env
# Rscript rrn2ncbi.R operons.100.fa species_annotation operons.100_gb_names.fa
# script provided by Simone Maestri (https://github.com/MaestSi)
# 2019-10-16

suppressMessages(library("Biostrings"))
args = commandArgs(trailingOnly=TRUE)
db_file_name <- args[1]
annotation_file_name <- args[2]
output_file_name <- args[3]

db <- readDNAStringSet(db_file_name, "fasta")
orig_names <- names(db)
annotation_file <- read.table(file = annotation_file_name, sep = "\t", stringsAsFactors = FALSE)
new_names <- c()

for (i in 1:length(orig_names)) {
  new_names[i] <- annotation_file[which(orig_names[i] == annotation_file[, 2]), 3]
}

new_db <- db
names(new_db) <- new_names

writeXStringSet(x = new_db, filepath = output_file_name, format = "fasta", width = 20000)
```

Classification of the rrnDB with the classical genbank *nucl_gb.accession2taxid* leads to **8012** missing accessions. The same analysis using the second dump file *nucl_wgs.accession2taxid* missed **3755** accessions. This motivated merging both files to recover more taxons.

The 2 Genbank dumps *nucl_gb.accession2taxid* and *nucl_wgs.accession2taxid* were merged to cover most of the rrn DB accessions. Only **283** accessions from the rrnDB out of **11484** were still missed with the merged reference set.

```
wget ftp://ftp.ncbi.nlm.nih.gov/pub/taxonomy/accession2taxid/nucl_gb.accession2taxid.gz
gunzip nucl_gb.accession2taxid.gz

wget ftp://ftp.ncbi.nlm.nih.gov/pub/taxonomy/accession2taxid/nucl_wgs.accession2taxid.gz
gunzip nucl_wgs.accession2taxid.gz

# merge to a unique dataset
cat nucl_gb.accession2taxid nucl_wgs.accession2taxid | sort | uniq > nucl_merged.accession2taxid

# use to create annotation taxonomy as detailed in  Import_database.sh
python2.7 ./entrez_qiime/entrez_qiime.py \
 -i operons.100_gb_names.fa \
 -n ./taxonomy/taxdump \
```

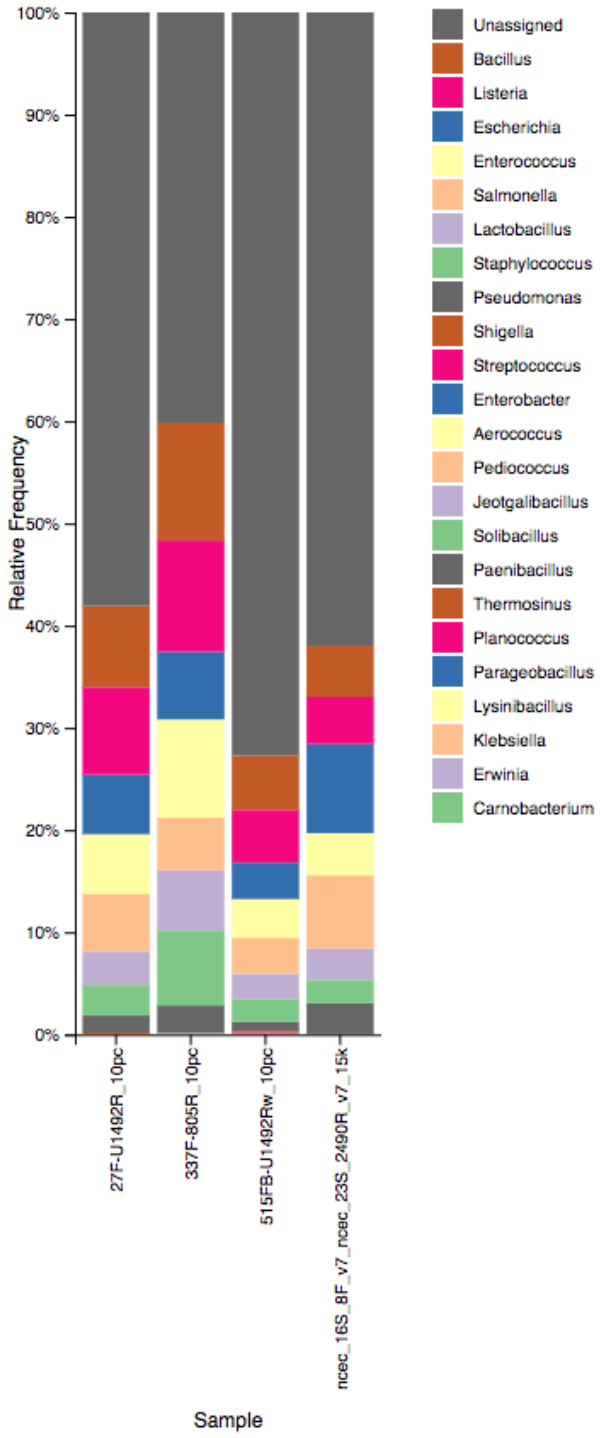
```
 -a ./taxonomy/nucl_merged.accession2taxid

# create Qiime2 artefacts for MetONTIIME
qiime tools import \
  --type FeatureData[Sequence] \
  --input-path rrnDB/operons.100_gb_names.fa\
  --output-path rrnDB_operons_sequence.qza

qiime tools import \
  --type FeatureData[Taxonomy] \
  --input-path  rrnDB/operons.100_merged_accession_taxonomy.txt \
  --input-format HeaderlessTSVTaxonomyFormat \
  --output-path rrnDB_operons_taxonomy.qza
```

MetONTIIME was then run with the reference data and the previously prepared 10% data subsets.

The final command is shown next

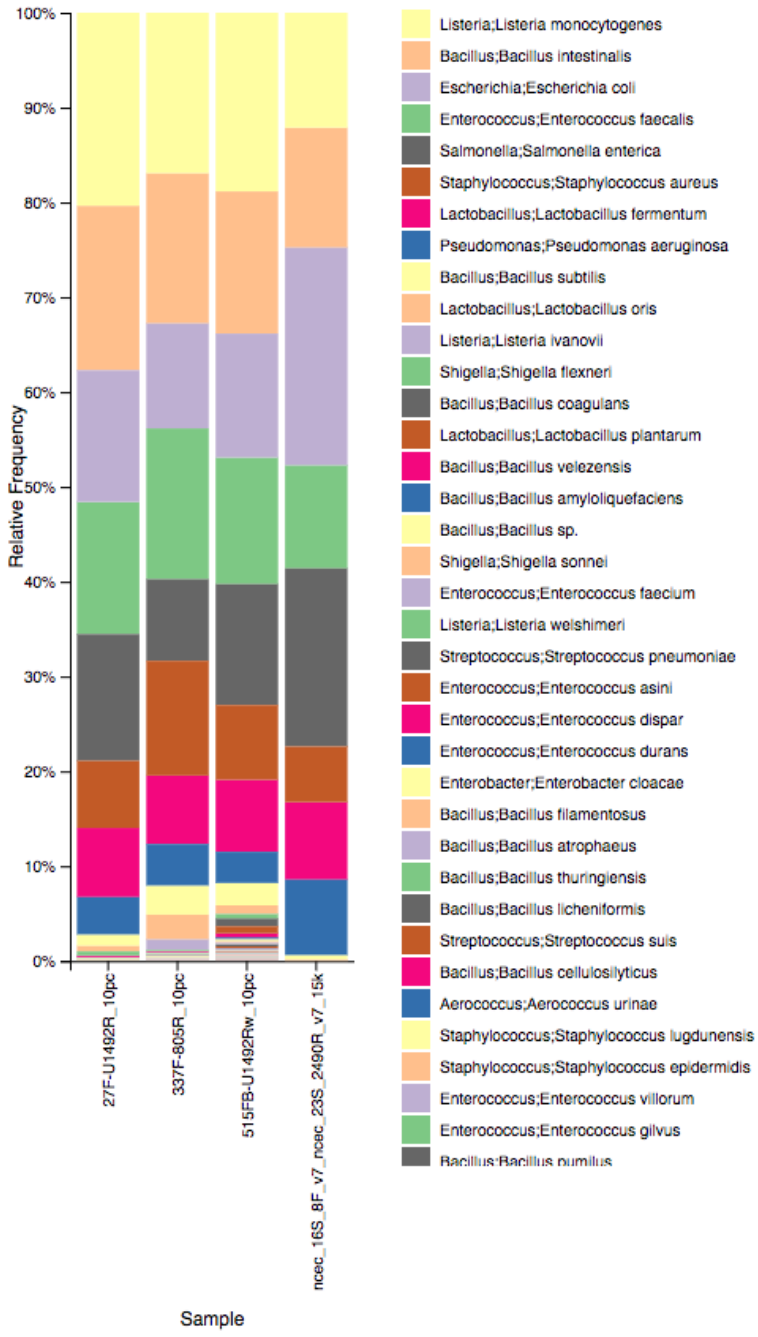
```
# rrnDB run with vsearch instead of blastn
./MetONTIIME.sh /data2/analyses/MetONTIIME_4smpl_rrnDB \
  /data2/analyses/MetONTIIME_4smpl_rrnDB/sample-metadata.tsv \
  /data/biodata/MetONTIIME_DB/rrnDB_operons_sequence.qza \
  /data/biodata/MetONTIIME_DB/rrnDB_operons_taxonomy.qza \
  84
```

The first figure reports the classification of the four samples at **genus** level with unassigned reads.



The proportion of unassigned reads varies largely between amplicons, suggesting that other loci are extracted from the genomic data due to sequence similarity with one or both primers as already seen in the epi2me analysis. We did not investigate what the unassigned sequences really are. and whether they cluster in specific regions of genomes as this falls beyond the topic of this study.

The next figure reports successful classification for the four samples at **species** level (after filtering out unassigned reads). Only the top 20 classes are shown in the legend to keep the plot readable.



Note: When using **rrnDB**, the **Escherichia Coli species is detected**, in contrast to the results obtained with NCBI references. The rrnDB is therefore so far the reference giving results most similar to the expected Zymo community and performs better (with MetONTIIME) than epi2me. As downside, the rrnDB being very large, it takes significantly more time to use it. One possible solution will be using **vscan** instead of **blastn** for the classification but this requires changes in the Qiime code (in progress after discussion with the developers).

Conclusion

Although different at the level of extracted read counts and raw classification results (degree of classification), the four amplicon datasets get very similar classifications using **ONT epi2me** or **MetONTIIME** and several reference databases.

Today, MetONTIIME is the only available 'out of the box' solution for classification with improved databases and ONT refuses to alter the epi2me tool to allow third-party references.

It is not yet clear whether a longer amplicon of 4.4kb which encompasses both the **16S** and **23S** regions could outperform the results obtained here. Such amplicon would include the *Internal Transcribed Spacer* (**ITS**) region which is known to be very variable and could add discriminating power to the classification if present in the reference database (eg: rrnDB). [this is ongoing work]

last edit: Wed Oct 23 14:30:13 2019

```
## R version 3.6.1 (2019-07-05)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 16.04.6 LTS
##
## Matrix products: default
## BLAS:   /usr/lib/openblas-base/libblas.so.3
## LAPACK: /usr/lib/libopenblasp-r0.2.18.so
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_1.0.2         magrittr_1.5       hms_0.5.1          munsell_0.5.0      tidyselect_0.2.5
##  [6] colorspace_1.4-1   R6_2.4.0           rlang_0.4.0        highr_0.8          stringr_1.4.0
## [11] tools_3.6.1        grid_3.6.1         gtable_0.3.0       xfun_0.10          htmltools_0.4.0
## [16] yaml_2.2.0         digest_0.6.21      assertthat_0.2.1   tibble_2.1.3       crayon_1.3.4
## [21] RColorBrewer_1.1-2 purrr_0.3.2        vctrs_0.2.0        zeallot_0.1.0      glue_1.3.1
## [26] evaluate_0.14      rmarkdown_1.16     stringi_1.4.3      compiler_3.6.1     pillar_1.4.2
## [31] scales_1.0.0       backports_1.1.5    pkgconfig_2.0.3
```