De-novo clustering of nanopore reads

Marko Suokas

Preprocess reads

Dorado does not support demultiplexing dual indexes located on both the 5' and 3' ends. Additionally, in ligated libraries, the reads can appear in either orientation. To address this, we use `cutadapt` for demultiplexing. Index pairs are identified using the linked adapters approach in both forward and reverse orientations, after which scripts are applied to reverse complement the reverse reads. Finally, the reads are merged.

Note: Be aware that autocorrect might change double dashes in command-line examples.

Extracting Forward Reads

You can extract forward reads into a FASTQ file using the following command:

```
cutadapt -e 0 -O 12 -g file:~/scripts/barcodes.fasta --trimmed-only \
-m 1200 -o "fdemuxed/{name}.fastq.gz" reads.fastq.gz
```

This command extracts barcodes defined in the `barcodes.fasta` file and outputs matching reads into individual files within the `fdemuxed` subdirectory. In this example, the minimum read length is set to 1200 bp.

Extracting Reverse Reads

To extract reverse reads, use the reverse-complemented barcode file:

```
cutadapt -e 0 -O 12 -g file:~/scripts/rev_barcodes.fasta \
--trimmed-only -m 1200 -o "rdemuxed/{name}.fastq.gz" \
reads.fastq.gz
```

The reads are demultiplexed into a separate directory.

Tip: Parameters `-O`, `-e`, `-m`, and `-M` can help reduce the chances of mismatched alignments.

Reverse Complementing Reverse Reads

Next, we use a bash script to process each reverse read file and reverse complement them using the following command:

```
seqkit seq -rp --seq-type DNA -o reverse_comp.fastq.gz \
reverse_out.fastq.gz
```

Merging Forward and Reverse Reads

Next, forward and reverse reads with the same base name are merged from two directories. Here's a simple bash command for that:

```
cat forward_out.fastq.gz reverse_comp.fastq.gz >merged_reads.fastq.gz
```

Trimming Primers

Finally, `cutadapt` and bash script can be employed to trim forward and reverse PCR primers from the sequence reads.

Import sequence data to Qiime 2

```
#Activate qiime environment
source /opt/miniconda3/etc/profile.d/conda.sh
conda activate qiime2-2024.2
# QIIME 2 command to import sequence data
qiime tools import \
  --type 'SampleData[SequencesWithQuality]' \
  --input-path data/reads/set2/manifest.csv \
  --output-path data/demux.qza \
  --input-format SingleEndFastqManifestPhred33
```

Dereplicate sequences

Dereplication removes unnecessary redundancy from sequence files

```
source /opt/miniconda3/etc/profile.d/conda.sh
conda activate qiime2-2024.2
# Dereplicate sequences with vsearch plugin
qiime vsearch dereplicate-sequences \
    --p-min-seq-length 1200 \
    --o-dereplicated-table data/derep_table.qza \
    --o-dereplicated-sequences data/derep_sequences.qza \
    --i-sequences data/demux.qza
```

Pick de-novo features

Step executed at CSC.

```
#!/bin/bash
#SBATCH --job-name=cluster
#SBATCH --account=project_2010620
#SBATCH --time=48:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=1
#SBATCH --cpus-per-task=32
#SBATCH --mem=48G
#SBATCH --partition=small
#SBATCH --gres=nvme:100

#set up qiime
module load qiime2/2024.2-amplicon
# run task. Don't use srun in submission as it resets TMPDIR
qiime vsearch cluster-features-de-novo \
   --i-sequences data/derep_sequences.qza \
   --i-table data/derep_table.qza \
   --p-strand plus --p-threads 32 --p-perc-identity 0.97 \
   --output-dir data/de_novo
```

Filter rare features

Rare otus are removed from results before chimera detection

```
#Activate qiime environment
source /opt/miniconda3/etc/profile.d/conda.sh
conda activate qiime2-2024.2
# QIIME 2 command to filter rare features
qiime feature-table filter-features \
    --i-table data/de_novo/clustered_table.qza \
    --p-min-frequency 10 \
    --o-filtered-table data/de_novo/f1_table.qza
qiime feature-table filter-seqs \
    --i-data data/de_novo/clustered_sequences.qza \
    --i-table data/de_novo/f1_table.qza \
    --o-filtered-data data/de_novo/f1_sequences.qza
```

Detect chimeric features

```
#Activate qiime environment
source /opt/miniconda3/etc/profile.d/conda.sh
conda activate qiime2-2024.2

qiime vsearch uchime-denovo --i-sequences data/de_novo/f1_sequences.qza \
    --i-table data/de_novo/f1_table.qza  \
    --o-chimeras data/de_novo/chimeras.qza --o-stats data/de_novo/stats.qza \
    --o-nonchimeras data/de_novo/nonchimeras.qza
```

Filter chimeras from the table file

```
#Activate qiime environment
source /opt/miniconda3/etc/profile.d/conda.sh
conda activate qiime2-2024.2
# QIIME 2 command to keep nonchimeric features
 qiime feature-table filter-features \
    --i-table data/de_novo/f1_table.qza \
    --m-metadata-file data/de_novo/nonchimeras.qza \
    --o-filtered-table data/de_novo/otu_table.qza
```

Filter chimeras from the sequence file

```
#Activate qiime environment
source /opt/miniconda3/etc/profile.d/conda.sh
conda activate qiime2-2024.2
# QIIME 2 command to keep nonchimeric sequences
qiime feature-table filter-seqs \
   --i-data data/de_novo/f1_sequences.qza \
   --i-table data/de_novo/otu_table.qza \
   --o-filtered-data data/de_novo/otu_sequences.qza
```

R libraries

```
library(dada2)
library(mia)
library(scater)
library(vegan)
library(Biostrings)
library(tidyverse)
library(kableExtra)
library(ggthemes)
library(ggpubr)
```

Import qiime otu table and project metadata

```r
#sequence file
tse <- importQIIME2(featureTableFile = "data/de_novo/otu_table.qza")
tse <- tse[, sort(colnames(tse))]
#add metadata
metadata <- data.frame(read_tsv("data/set2_meta.tsv",
                                show_col_types = F))
metadata <- column_to_rownames(metadata, "Sampleid")
colData(tse) <- DataFrame(metadata)
tse
```

```
class: TreeSummarizedExperiment
dim: 1469 6
metadata(0):
assays(1): counts
rownames(1469): 780015658d5994b3e7649355028cdb0ede4aa40e
  92901274c2b00cc23f7a06885764bc41e8da85ec ...
  0ba42ae60d68d1bea239c158de0013e7e169b879
  586e676bc2eba711e9befb7156796a22819d5f91
rowData names(0):
colnames(6): barcode007 barcode008 ... barcode011 barcode012
colData names(2): Name Media
reducedDimNames(0):
mainExpName: NULL
altExpNames(0):
rowLinks: NULL
rowTree: NULL
colLinks: NULL
colTree: NULL
```

Qiime arranges sequence file alphabetically, while TSE expects sequences in same order as rownames. Thus we need to rearrange sequences

```r
ref_sequences <- importQZA("data/de_novo/otu_sequences.qza")
ref_ids <- names(ref_sequences)
tse_ids <- rownames(tse)
# Check if all rownames are present in the reference IDs
if (!all(tse_ids %in% ref_ids)) {
  stop("Not all rownames from tse are present in the reference sequences.")
}
# Reorder `ref_sequences` to match the order of `tse` rownames
ref_sequences_ordered <- ref_sequences[match(tse_ids, ref_ids)]
all(names(ref_sequences_ordered) == rownames(tse))
```

```
[1] TRUE
```

```
referenceSeq(tse) <- ref_sequences_ordered
```

Assign taxonomy

```
taxa <- assignTaxonomy(referenceSeq(tse), minBoot=90, multithread=2,
        refFasta="~/feature_classifiers/silva_nr99_v138.1_train_set.fa.gz")
saveRDS(taxa, "data/de_novo/taxa.rds" )
```

Add taxonomy results to rowData and rename identifiers

```
taxa <- readRDS("data/de_novo/taxa.rds")
#Add taxonomy
rownames(taxa) <- NULL
rowData(tse) <- DataFrame(taxa)
#Rename rows (alternative to Silva ID)
rownames(tse) <- paste0("OTU_", seq_len(nrow(tse)))
tse
```

```
class: TreeSummarizedExperiment
dim: 1469 6
metadata(0):
assays(1): counts
rownames(1469): OTU_1 OTU_2 ... OTU_1468 OTU_1469
rowData names(6): Kingdom Phylum ... Family Genus
colnames(6): barcode007 barcode008 ... barcode011 barcode012
colData names(2): Name Media
reducedDimNames(0):
mainExpName: NULL
altExpNames(0):
rowLinks: NULL
rowTree: NULL
colLinks: NULL
colTree: NULL
referenceSeq: a DNAStringSet (1469 sequences)
```

Write object data to files

Write RDS. The object can be easily reloaded in R

```r
saveRDS(tse, "results/denovo/tse.rds")
```

Write an abundance table

```r
#FeatureID will be rowname
abd <- data.frame(FeatureID = rownames(tse),assays(tse)$counts)
#Write
write_tsv(abd, "results/denovo/feature_table.tsv")
```

Write a taxonomy table

```r
#FeatureID will be rowname
taxt <- data.frame(FeatureID = rownames(tse), rowData(tse))
#Write
write_tsv(taxt, "results/denovo/taxonomy.tsv")
```

Write variant sequences to fasta file

```r
writeXStringSet(referenceSeq(tse), "results/denovo/repseq.fasta",
                                      append = F, compress = F,
                                      format = "fasta")
```

Write a metadata file

```r
metadf <- data.frame(colData(tse)) %>% rownames_to_column(var="Sampleid")
#write
write_tsv(metadf, "results/denovo/metadata.tsv")
```

Microbial data analysis

Agglomerate taxonomy to genus rank and count relative abundance

```
altExp(tse, "Genus") <- agglomerateByRank(tse, rank="Genus",
                             onRankOnly=T, na.rm=F)
#relabundance
altExp(tse, "Genus") <- transformAssay(altExp(tse, "Genus"),
                                       assay.type="counts",
                                       method="relabundance")
```

Pick ten most abundant features

```
#top10 features
top10 <- getTopFeatures(altExp(tse, "Genus"), top=10,
                        method = "mean",
                        assay.type="relabundance")
#create and filter table
table <- data.frame(assays(altExp(tse, "Genus"))$relabundance)
table <- table %>%
  rownames_to_column(var = "Genus") %>%
  filter(Genus %in% top10) %>% bind_rows(
  summarise(., Genus = "Others", across(where(is.numeric), ~ 1 - sum(.))))
```

Print abundance table

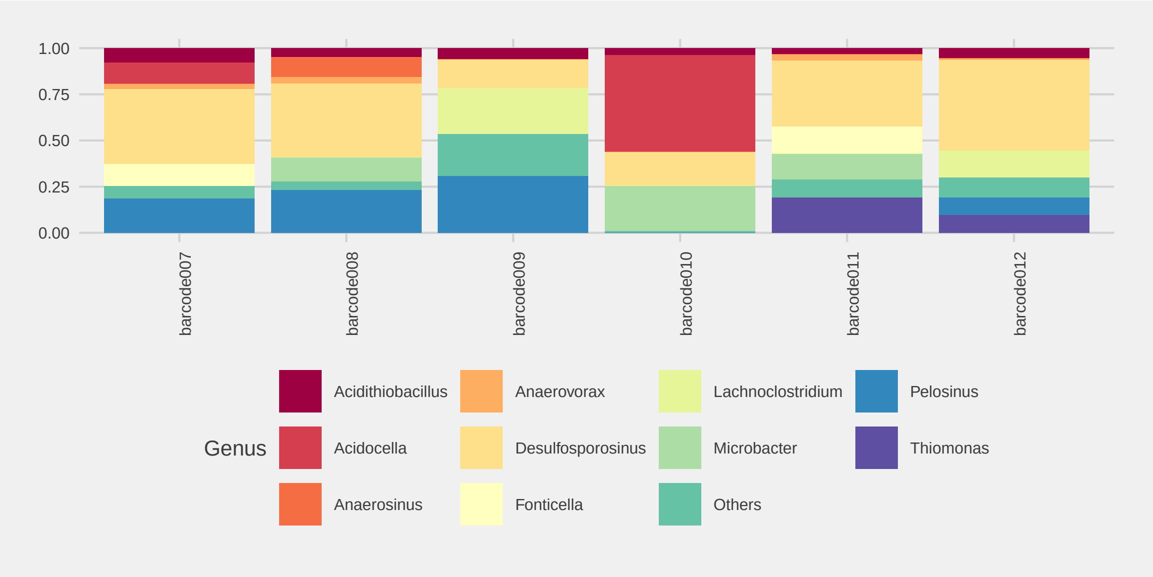
```
kable(table, digits=2) %>%
  kable_styling(latex_options = c("HOLD_position", "striped"),
                font_size = 10) %>%
  row_spec(0, background = "teal", color = "white")
```

Genus	barcode007	barcode008	barcode009	barcode010	barcode011	barcode012
Acidocella	0.12	0.00	0.00	0.53	0.00	0.00
Microbacter	0.00	0.13	0.00	0.25	0.14	0.00
Acidithiobacillus	0.08	0.05	0.06	0.04	0.03	0.05
Fonticella	0.12	0.00	0.00	0.00	0.15	0.00
Thiomonas	0.00	0.00	0.00	0.00	0.19	0.10
Pelosinus	0.19	0.23	0.31	0.00	0.00	0.10
Desulfosporosinus	0.41	0.40	0.15	0.19	0.35	0.49
Lachnoclostridium	0.00	0.00	0.25	0.00	0.00	0.14
Anaerovorax	0.03	0.04	0.00	0.00	0.03	0.01
Anaerosinus	0.00	0.11	0.00	0.00	0.00	0.00
Others	0.07	0.04	0.23	0.01	0.10	0.11

Composition plot

Transform data to long table format

```
df_long <- table %>% pivot_longer(cols = starts_with("barcode"),
names_to = "Sample", values_to = "Abundance")
#Plot stacked barplot
ggplot(df_long, aes(x=Sample, y=Abundance, fill=Genus)) +
geom_bar(stat = "identity") +
   theme_fivethirtyeight(base_size=8) +
   scale_fill_brewer(palette = "Spectral") +
   theme(axis.text.x=element_text(angle=90))
```



Rarefaction

Prior diveristy calculations, we rarefy data to minimize the effect of varying sample sizes.

```
totalcounts <- colSums(assays(tse)$counts)
totalcounts
```

```
barcode007 barcode008 barcode009 barcode010 barcode011 barcode012
    345406     212800     361049     367188     320530     273280
```

```
set.seed(456)
tse <- subsampleCounts(tse, name="subsampled", assay.type = "counts",
                       min_size = 210000)
```

Alpha diversity

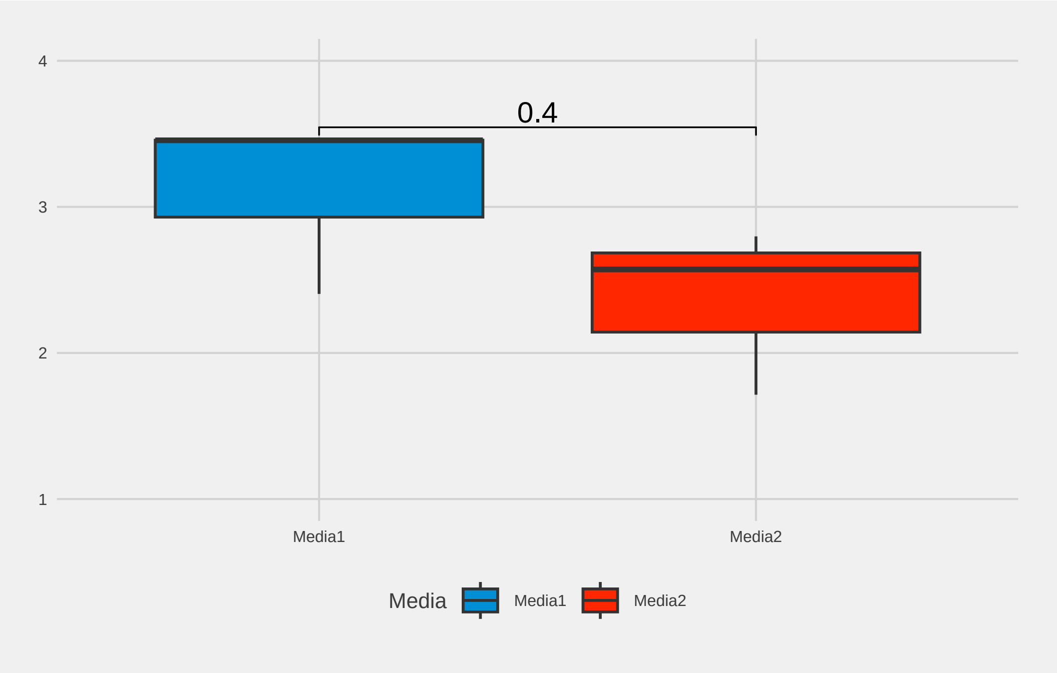
```r
tse <- estimateDiversity(tse, assay.type="counts", index="shannon")
shannon <- data.frame(Samples = colnames(tse),
                      Shannon_index = colData(tse)$shannon)
rownames(shannon) <- NULL
#colnames(shannon) <- c("Sample", "Shannon index")
#table
kable(shannon, digits=2, caption = "Shannon index") %>%
kable_styling(latex_options = c("HOLD_position", "striped"),
              font_size = 11) %>% row_spec(0, background = "teal",
                                           color = "white")
```

Table 1: Shannon index

Samples	Shannon_index
barcode007	3.46
barcode008	3.46
barcode009	2.40
barcode010	1.71
barcode011	2.80
barcode012	2.57

Shannon boxplot between two media groups

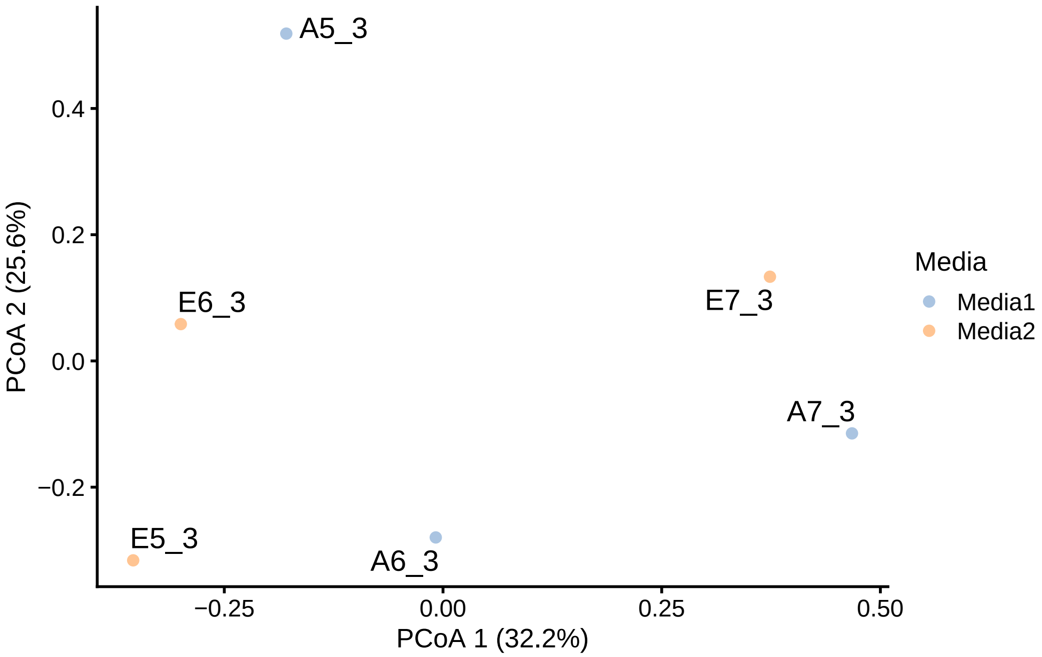
```r
comparisons <- list(c("Media1", "Media2"))
media <- ggplot(colData(tse), aes(x=Media, y=shannon, fill=Media)) +
  geom_boxplot(ylim=c(0,4)) + stat_compare_means(comparisons = comparisons,
                                      method = "wilcox.test",
                                      label = "p.format") +
  theme_fivethirtyeight( base_size=8) + scale_fill_fivethirtyeight() +
    ylim(1, 4)
media
```



Beta diversity measured by Bray-Curtis dissimilarity

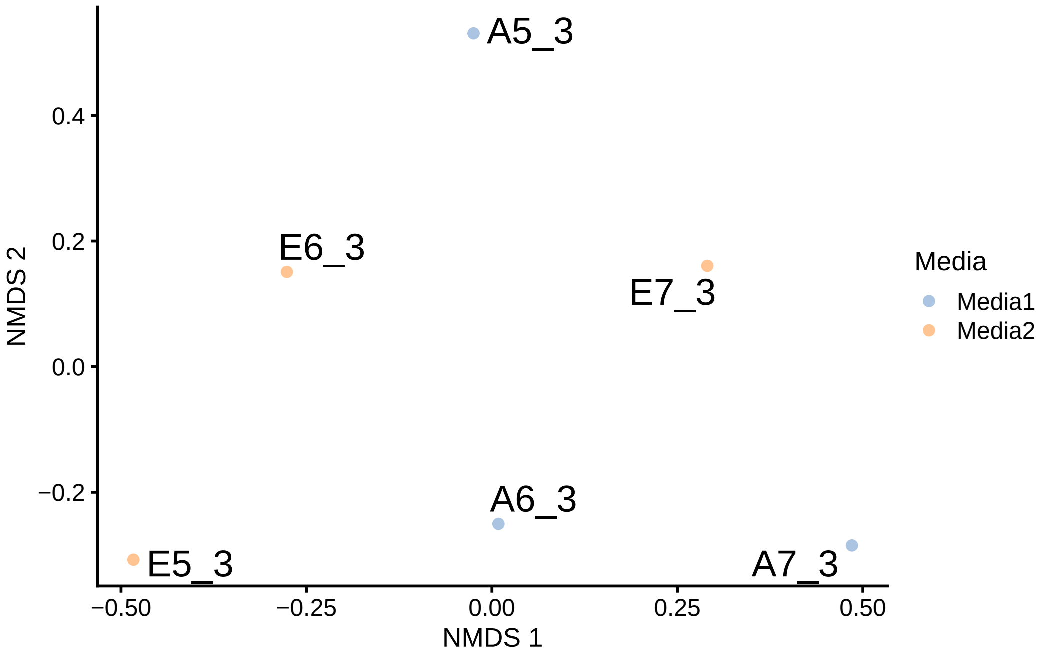
PCoA plot illustrating community composition differences using Bray-Curtis dissimilarity distances

```
tse <- runMDS(tse, FUN = vegan::vegdist, method = "bray",
              name="PCoA_BC", exprs_values = "subsampled")
#Explained variance
e <- attr(reducedDim(tse, "PCoA_BC"), "eig")
rel_eig <- e / sum(e[e > 0])
#Plot
plotReducedDim(tse, "PCoA_BC", colour_by = "Media",
               text_by = "Name", text_size = 4) +
  labs(x = paste("PCoA 1 (", round(100 * rel_eig[[1]], 1),
                 "%", ")", sep = ""), y = paste("PCoA 2 (",
                 round(100 * rel_eig[[2]], 1),
                 "%", ")", sep = ""))
```



NMDS plot illustrating community composition differences using Bray-Curtis dissimilarity distances

```
data <- t(assay(tse, "counts"))
bray_curtis <- vegdist(data, method = "bray")
nmds <- metaMDS(bray_curtis, k = 2, trymax = 100, trace=0)
nmds_coords <- as.data.frame(scores(nmds, display = "sites"))
reducedDim(tse, "NMDS") <- nmds_coords
plotReducedDim(tse, dimred = "NMDS", colour_by = "Media", text_by="Name")
```

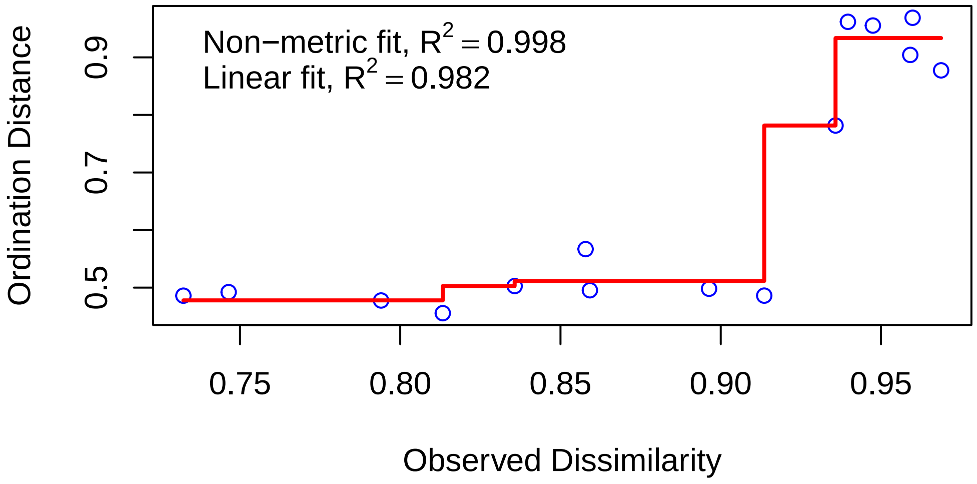


Stress values of NMDS plot

```
# Stress value
print(paste0("NMDS stress value ",nmds$stress))
```

```
[1] "NMDS stress value 0.0398873869824664"
```

```
# Stress plot
stressplot(nmds)
```



Results

The analysis method was switched to de novo clustering at a 97% sequence identity level due to a lack of studies on the performance of new Nanopore sequencing in complex microbial communities. We do have evidence that the sequencer operates accurately when analyzing mock community standards composed of a few microbial species with known quantities.

Each OTU picking strategy introduces minor variations, but the overall patterns remain consistent, as highlighted in the comparison document. De novo clustering appears to be the best compromise among the available methods, followed closely by open-reference clustering, as both methods retain variants that do not match perfectly with the reference database. In contrast, closed-reference OTU picking heavily depends on how well the studied communities are represented in the database.

The Shannon index values suggest that the samples contain enriched microbial communities. While there appears to be some variation between samples grown in different media, this difference is not statistically significant. Although beta diversity analyses did not reveal significant similarities between communities, the plots clearly illustrate the bray-curtis distances between samples.