DADA_pipeline

Alisa Kabalina

2023-11-07

```
main_dir <- dirname(rstudioapi::getSourceEditorContext()$path)
setwd(main_dir)

path <- "/home/alisa/metagenomics_SPBU/Task3/trimmed/"
list.files(path)
```

```
##  [1] "analysis"                             "errors_forvard.RData"
##  [3] "errors_reverse.RData"                 "filtered"
##  [5] "group1_1_R1_paired.fastq"             "group1_1_R2_paired.fastq"
##  [7] "group1_2_R1_paired.fastq"             "group1_2_R2_paired.fastq"
##  [9] "group1_3_R1_paired.fastq"             "group1_3_R2_paired.fastq"
## [11] "group1_4_R1_paired.fastq"             "group1_4_R2_paired.fastq"
## [13] "group1_5_R1_paired.fastq"             "group1_5_R2_paired.fastq"
## [15] "group1_6_R1_paired.fastq"             "group1_6_R2_paired.fastq"
## [17] "group2_1_R1_paired.fastq"             "group2_1_R2_paired.fastq"
## [19] "group2_2_R1_paired.fastq"             "group2_2_R2_paired.fastq"
## [21] "group2_3_R1_paired.fastq"             "group2_3_R2_paired.fastq"
## [23] "group2_4_R1_paired.fastq"             "group2_4_R2_paired.fastq"
## [25] "group2_5_R1_paired.fastq"             "group2_5_R2_paired.fastq"
## [27] "group2_6_R1_paired.fastq"             "group2_6_R2_paired.fastq"
## [29] "R_script_metagenomics.html"           "R_script_metagenomics.R"
## [31] "R_script_metagenomics.Rmd"            "results"
## [33] "seqtab_nochim.csv"                    "seqtab_nochim.RData"
## [35] "silva_nr_v138_train_set.fa.gz"        "silva_species_assignment_v138.fa.gz"
## [37] "Snakefile"                            "taxa.RData"
## [39] "taxa_df.csv"                          "taxa_print.csv"
## [41] "taxa_print.RData"                     "unpair_trim"
```

Let’s install and activate libraries

```
#install.packages("BiocManager")
```

```
#BiocManager::install("dada2", version = "3.18")
#BiocManager::install("phyloseq")
library (ggplot2)
library(dplyr)
library(dada2)
library(phyloseq)
library(vegan)
```

First we read in the names of the fastq files, and perform some string manipulation to get lists of the forward and reverse fastq files in matched order

```
# Forward and reverse fastq filenames have format: group1_1_R1_paired.fastq and group1_1_R2_paired.fastq
fnFs <- sort(list.files(path, pattern="_R1_paired.fastq", full.names = FALSE))
fnRs <- sort(list.files(path, pattern="_R2_paired.fastq", full.names = FALSE))

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

```
##  [1] "group1_1" "group1_2" "group1_3" "group1_4" "group1_5" "group1_6"
##  [7] "group2_1" "group2_2" "group2_3" "group2_4" "group2_5" "group2_6"
```

As DADA required no N bases in the sequences, we have to remove it

```
filtFs <- file.path(path, "filtered", paste0(sample.names, "_F_filt.fastq"))
filtRs <- file.path(path, "filtered", paste0(sample.names, "_R_filt.fastq"))
out <- filterAndTrim(fnFs, filtFs, fnRs, filtRs, maxN=0, compress=TRUE, multithread=FALSE)
out
```

```
##                           reads.in reads.out
## group1_1_R1_paired.fastq    68317     68317
## group1_2_R1_paired.fastq    39229     39229
## group1_3_R1_paired.fastq    48972     48972
## group1_4_R1_paired.fastq    42448     42448
## group1_5_R1_paired.fastq    44773     44773
## group1_6_R1_paired.fastq    46630     46630
## group2_1_R1_paired.fastq    55815     55815
## group2_2_R1_paired.fastq    59660     59660
## group2_3_R1_paired.fastq    68186     68186
## group2_4_R1_paired.fastq    47808     47808
## group2_5_R1_paired.fastq    51964     51964
## group2_6_R1_paired.fastq    38732     38732
```

Learn the Error Rates

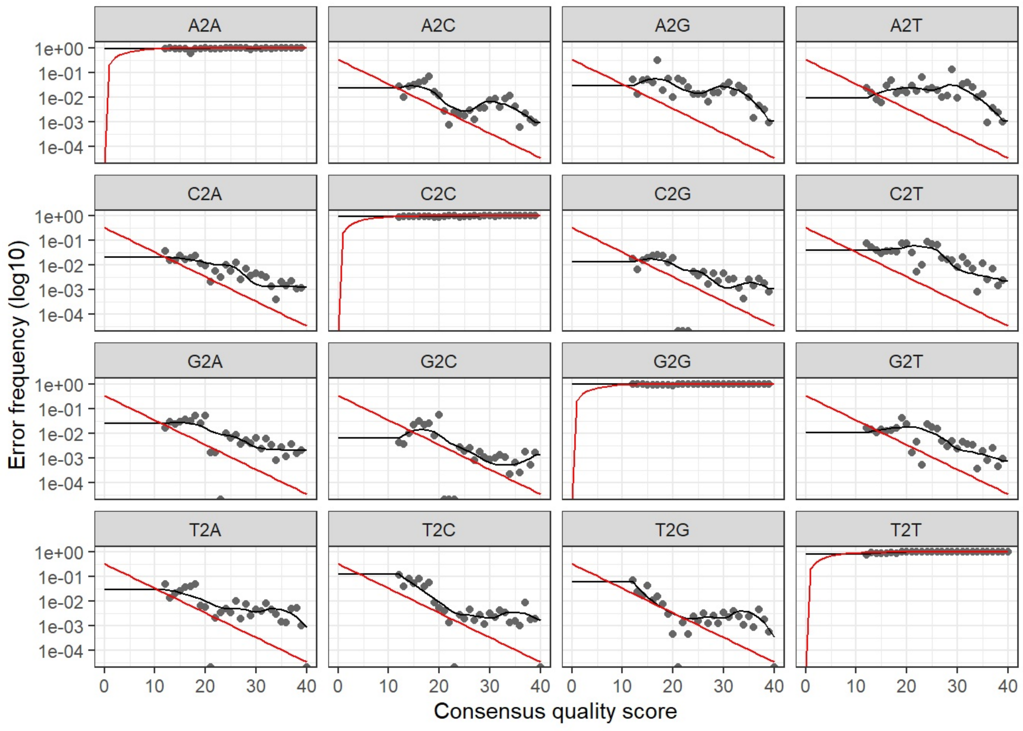
The learnErrors method learns this error model from the data, by alternating estimation of the error rates and inference of sample composition until they converge on a jointly consistent solution.

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

# or import te data that was calculated before
errF <- readRDS("errors_forvard.RData")
```

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

```
## Warning: Transformation introduced infinite values in continuous y-axis
## Transformation introduced infinite values in continuous y-axis
```



Save results of learnErrors for later usage

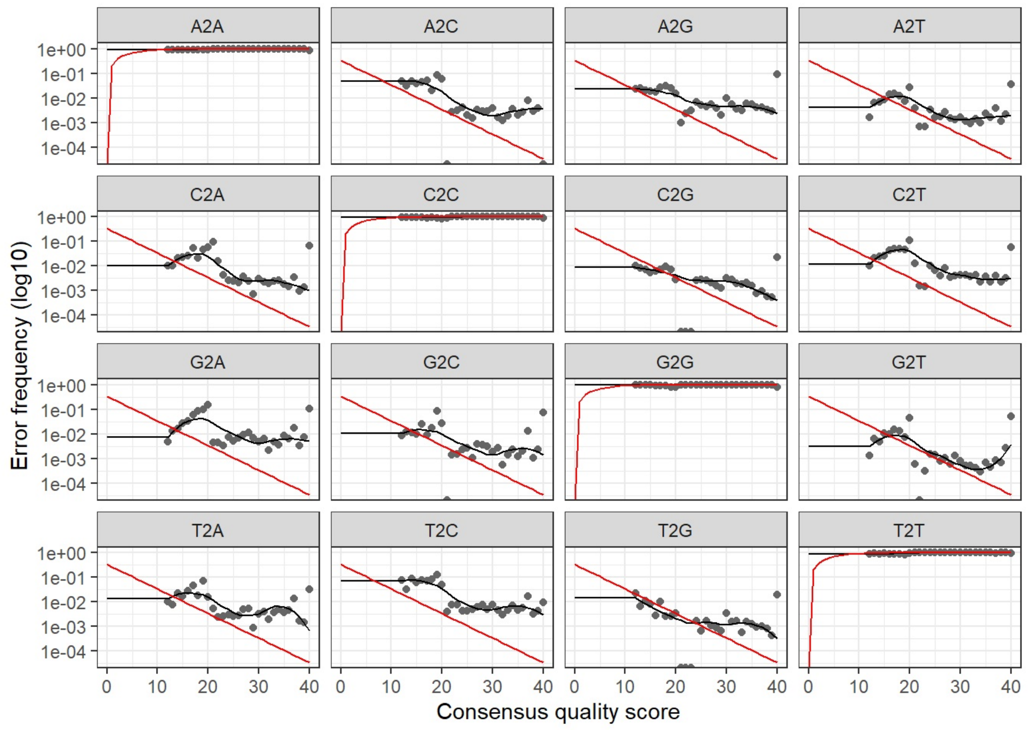
```
#saveRDS(errF, file="errors_forvard.RData")
```

Then we have to do the same for the reverse samples

```
# errR <- learnErrors(filtRs, multithread=TRUE)
# saveRDS(errR, file="errors_reverse.RData")
errR <- readRDS("errors_reverse.RData")
```

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

```
## Warning: Transformation introduced infinite values in continuous y-axis
## Transformation introduced infinite values in continuous y-axis
```



Dereplication

Next step is to do dereplication of the sequences. Dereplication combines all identical sequencing reads into into “unique sequences” with a corresponding “abundance” equal to the number of reads with that unique sequence.

```
derepFs <- derepFastq(filtFs, verbose=TRUE)
derepRs <- derepFastq(filtRs, verbose=TRUE)

# Name the derep-class objects by the sample names
names(derepFs) <- sample.names
names(derepRs) <- sample.names
```

Then we use the core sample inference algorithm to the dereplicated data.

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

```
## Sample 1 - 68317 reads in 26983 unique sequences.
## Sample 2 - 39229 reads in 21272 unique sequences.
## Sample 3 - 48972 reads in 24565 unique sequences.
## Sample 4 - 42448 reads in 16608 unique sequences.
## Sample 5 - 44773 reads in 20113 unique sequences.
## Sample 6 - 46630 reads in 21437 unique sequences.
## Sample 7 - 55815 reads in 21185 unique sequences.
## Sample 8 - 59660 reads in 21760 unique sequences.
## Sample 9 - 68186 reads in 24698 unique sequences.
## Sample 10 - 47808 reads in 22978 unique sequences.
## Sample 11 - 51964 reads in 16043 unique sequences.
## Sample 12 - 38732 reads in 20635 unique sequences.
```

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

```
## Sample 1 - 68317 reads in 30913 unique sequences.
## Sample 2 - 39229 reads in 24663 unique sequences.
## Sample 3 - 48972 reads in 29335 unique sequences.
## Sample 4 - 42448 reads in 21624 unique sequences.
## Sample 5 - 44773 reads in 26678 unique sequences.
## Sample 6 - 46630 reads in 25864 unique sequences.
## Sample 7 - 55815 reads in 26064 unique sequences.
## Sample 8 - 59660 reads in 26500 unique sequences.
## Sample 9 - 68186 reads in 28107 unique sequences.
## Sample 10 - 47808 reads in 27351 unique sequences.
## Sample 11 - 51964 reads in 17609 unique sequences.
## Sample 12 - 38732 reads in 22104 unique sequences.
```

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

```
## dada-class: object describing DADA2 denoising results
## 343 sequence variants were inferred from 26983 input unique sequences.
## Key parameters: OMEGA_A = 1e-40, OMEGA_C = 1e-40, BAND_SIZE = 16
```

The next step is to merge paired reads to obtain the full sequences.

```
mergers <- mergePairs(dadaFs, derepFs, dadaRs, derepRs)
```

```
## Duplicate sequences in merged output.
```

```
# mergers[[1]]
```

Construct an amplicon sequence variant (ASV) table

```
seqtab <- makeSequenceTable(mergers)

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

```
##
##   93  112  125  135  143  147  148  166  167  168  218  286  289  316  329  424
##    1    1    1    1    1    3    1   33    1    1    1    2    1    1    1    3
##  438  439  440  441  442  443  444  445  446  447  448  449  457  458  459  460
##    1  177 1248  501  215  283   17   74   56   78   11  114    4    7  107 3781
##  461  462  463  464  465  466
##    5    1    3  392 1233  156
```

Remove chimeras

Or just imoport the data that was calculated before

```
#seqtab.nochim <- removeBimeraDenovo(seqtab, method="consensus", multithread=TRUE, verbose=TRUE)
seqtab.nochim <- readRDS('seqtab_nochim.RData')
asv_tab <- t(seqtab.nochim)
dim(seqtab.nochim)
```

```
## [1]   12 1637
```

```
#saveRDS(seqtab.nochim, 'seqtab_nochim.RData')
#write.csv(seqtab.nochim, 'seqtab_nochim.csv', quote=FALSE)
#write.csv(asv_tab, 'analysis/asv_tab.csv', quote=FALSE)


str(seqtab.nochim)
```

```
##  int [1:12, 1:1637] 359 0 734 0 828 0 1345 1418 518 202 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : chr [1:12] "group1_1" "group1_2" "group1_3" "group1_4" ...
##   ..$ : chr [1:1637] "CCTACGGGAGGCTGCAGTGGGGAATCTTGCGCAATGGGGGGAACCCTGACGCAGCGACGCCGCGTGCGGGACGGAGGCCTTCGGGTCG
TAAACCGCTTTCAGCAGGGAAGA"| __truncated__ "CCTACGGGAGGCAGCAGTGGGGAATCTTGCGCAATGGGGGGAACCCTGACGCAGCGACGCCGCGTGCGGGAC
GGAGGCCTTCGGGTCGTAAACCGCTTTCAGCAGGGAAGA"| __truncated__ "CCTACGGGAGGCAGCAGTGGGGAATATTGCACAATGGGCGCAAGCCTGATGCAGCG
ACGCCGCGTGCGGGATGACGGCCTTCGGGTTGTAAACCGCTTTTGACTGGGAGCA"| __truncated__ "CCTACGGGAGGCTGCAGTGGGGAATATTGCACAATGGGCG
CAAGCCTGATGCAGCGACGCCGCGTGCGGGATGACGGCCTTCGGGTTGTAAACCGCTTTTGACTGGGAGCA"| __truncated__ ...
```

```
row.names(seqtab.nochim)
```

```
##  [1] "group1_1" "group1_2" "group1_3" "group1_4" "group1_5" "group1_6"
##  [7] "group2_1" "group2_2" "group2_3" "group2_4" "group2_5" "group2_6"
```

Let’s take a look how much data don’t contain chimeras

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

```
## [1] 0.566857
```

The final step before assigning taxonomy is to look at the number of reads that passed at every step of the pipeline

```
getN <- function(x) sum(getUniques(x))
track <- cbind(out, sapply(dadaFs, getN), sapply(dadaRs, getN), sapply(mergers, getN), rowSums(seqtab.nochim))
```

```
## Duplicate sequences detected and merged.
```

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

```
##          input filtered denoisedF denoisedR merged nonchim
## group1_1 68317    68317     66992     67299  51030   26846
## group1_2 39229    39229     37491     37930  25061   16536
## group1_3 48972    48972     46871     47480  32564   19326
## group1_4 42448    42448     41285     41884  31910   15380
## group1_5 44773    44773     43154     43931  32858   23867
## group1_6 46630    46630     45053     45727  31419   15308
## group2_1 55815    55815     55019     55083  46001   25346
## group2_2 59660    59660     58558     59202  49287   25300
## group2_3 68186    68186     66924     67115  55880   30829
## group2_4 47808    47808     45734     46879  30212   24010
## group2_5 51964    51964     50856     50833  46884   18957
## group2_6 38732    38732     36621     37303  24662   17784
```

Assign taxonomy

```
# taxa <- assignTaxonomy(seqtab.nochim, "silva_nr_v138_train_set.fa.gz", multithread=TRUE)

# bad_taxa = as.character(rownames(taxa))[(taxa[, 1] == 'Eukaryota' &
#                                           is.na(taxa[, 3])) | is.na(taxa[, 2])]
# print(length(bad_taxa))
# print(ncol(seqtab.nochim))
# seqtab.nochim = seqtab.nochim[, sapply(as.character(colnames(seqtab.nochim)),
#                                       function(x) !(x %in% bad_taxa))]
# print(nrow(seqtab.nochim))

# taxa <- assignTaxonomy(seqtab.nochim, "silva_nr_v138_train_set.fa.gz", multithread=TRUE)

# taxa <- addSpecies(taxa, "silva_species_assignment_v138.fa.gz", verbose=TRUE, allowMultiple=T)
```

Or import data that was calculated before

```
taxa <- readRDS("taxa.RData")
taxa_df <- read.csv("analysis/taxa_df.csv")
taxa_print_df <- read.csv("taxa_print.csv")
```

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

head(taxa_print_df)
```

```
##    Kingdom           Phylum          Class             Order             Family
## 1 Bacteria Actinobacteriota Coriobacteriia  Coriobacteriales  Coriobacteriaceae
## 2 Bacteria Actinobacteriota Coriobacteriia  Coriobacteriales  Coriobacteriaceae
## 3 Bacteria Actinobacteriota Actinobacteria Bifidobacteriales Bifidobacteriaceae
## 4 Bacteria Actinobacteriota Actinobacteria Bifidobacteriales Bifidobacteriaceae
## 5 Bacteria       Firmicutes     Clostridia   Oscillospirales    Ruminococcaceae
## 6 Bacteria Actinobacteriota Actinobacteria Bifidobacteriales Bifidobacteriaceae
##             Genus Species
## 1      Collinsella    <NA>
## 2      Collinsella    <NA>
## 3  Bifidobacterium    <NA>
## 4  Bifidobacterium    <NA>
## 5 Faecalibacterium    <NA>
## 6  Bifidobacterium    <NA>
```

Save the data for later usage

```
#saveRDS(taxa, file="taxa.RData")
#write.csv(taxa, "analysis/taxa_df.csv", quote=FALSE)
#write.csv(taxa_print_df, "taxa_print.csv", row.names=FALSE, quote=FALSE)
```

Alpha-diversity

Calculate the biodiversity index and compare the presence of statistical differences

```
colnames(seqtab.nochim) <-  as.character(sapply(colnames(seqtab.nochim), function(x) gsub('NNNNNNNNNN', '', x)))

samples_data <- data.frame(SampleID=sample.names)
rownames(samples_data) <- sample.names

ps <- phyloseq(otu_table(seqtab.nochim, taxa_are_rows=FALSE),
               sample_data(samples_data),
               tax_table(taxa))
```

Estimate richness

```
rich = estimate_richness(ps)
colnames(rich)[0] <- 'Sample'
rich <- rich %>% mutate_if(is.numeric, round, digits = 3)
rich
```

```
##          Observed Chao1 se.chao1     ACE se.ACE Shannon Simpson InvSimpson
## group1_1      225 225.0    0.000 225.000  2.211   5.193   0.993    151.674
## group1_2      147 147.0    0.249 147.359  2.872   4.597   0.987     74.657
## group1_3      216 216.0    0.000 216.000  4.260   4.901   0.989     90.829
## group1_4      139 139.0    0.000 139.000  2.746   4.648   0.989     87.723
## group1_5      187 187.0    0.000 187.000  1.978   4.909   0.990     98.228
## group1_6      152 152.5    1.298 152.482  4.196   4.586   0.987     75.925
## group2_1      191 191.0    0.000 191.000  2.411   4.863   0.989     88.321
## group2_2      190 205.0   13.961 198.971  3.417   4.793   0.988     80.835
## group2_3      252 252.0    0.000 252.000  1.722   5.217   0.993    142.974
## group2_4      181 181.0    0.249 181.668  2.184   4.942   0.991    115.455
## group2_5      118 118.0    0.000 118.000  1.710   4.272   0.980     49.178
## group2_6      198 198.0    0.499 198.262  2.534   4.994   0.991    115.484
##          Fisher
## group1_1 33.686
## group1_2 22.242
## group1_3 34.053
## group1_4 21.080
## group1_5 27.656
## group1_6 23.446
## group2_1 28.059
## group2_2 27.895
## group2_3 37.565
## group2_4 26.597
## group2_5 16.797
## group2_6 31.193
```

Shannon index varies between 4.272 and 5.217. That means that we have more than one dominant species Chao1 is an indicator of species richness. There we can see that it varies between 118 and 252. So we can conclude that there is quite high richness in our samples

```
wilcox.test(rich$Chao1[1:6], rich$Chao1[7:12])
```

```
##
##  Wilcoxon rank sum exact test
##
## data:  rich$Chao1[1:6] and rich$Chao1[7:12]
## W = 15, p-value = 0.6991
## alternative hypothesis: true location shift is not equal to 0
```

```
wilcox.test(rich$Shannon[1:6], rich$Shannon[7:12])
```

```
##
##  Wilcoxon rank sum exact test
##
## data:  rich$Shannon[1:6] and rich$Shannon[7:12]
## W = 14, p-value = 0.5887
## alternative hypothesis: true location shift is not equal to 0
```

```
wilcox.test(rich$Simpson[1:6], rich$Simpson[7:12])
```

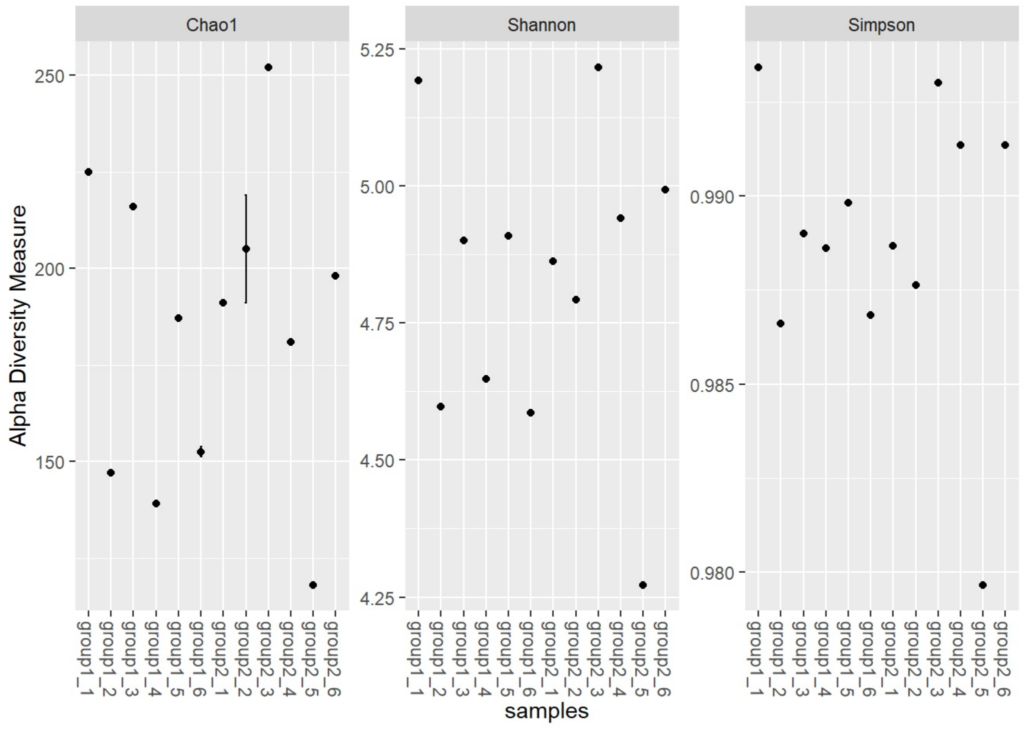
```
## Warning in wilcox.test.default(rich$Simpson[1:6], rich$Simpson[7:12]): не могу
## подсчитать точное p-значение при наличии повторяющихся наблюдений
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  rich$Simpson[1:6] and rich$Simpson[7:12]
## W = 15.5, p-value = 0.7457
## alternative hypothesis: true location shift is not equal to 0
```

Based on test results we can conclude that our observations belong to the same general population (p-value > 0.05)

```
#dir.create('results')
write.csv(rich, "results/alpha_diversity.csv", row.names=FALSE, quote=FALSE)
```

```
plot_richness(ps, measures=c("Chao1", "Shannon", "Simpson"))
```



Eveness

Transform data to proportions as appropriate for Bray-Curtis distances

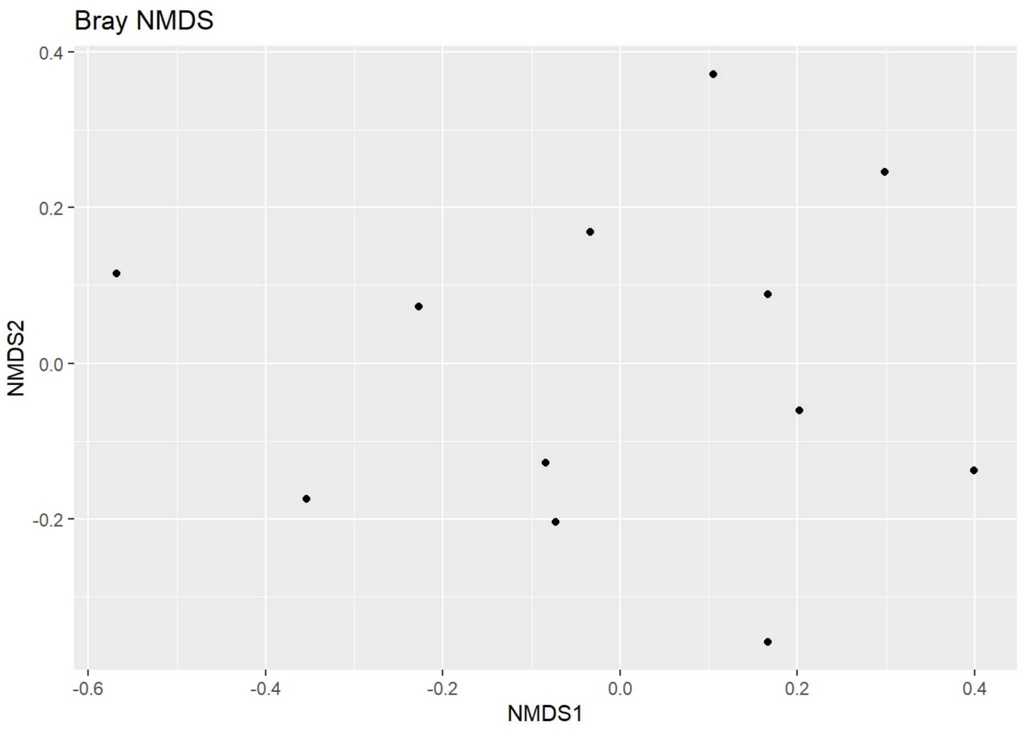
```
ps.prop <- transform_sample_counts(ps, function(otu) otu/sum(otu))
ord.nmds.bray <- ordinate(ps.prop, method="NMDS", distance="bray")
```

```
## Run 0 stress 0.171756
## Run 1 stress 0.1786174
## Run 2 stress 0.2156631
## Run 3 stress 0.2013323
## Run 4 stress 0.2156634
## Run 5 stress 0.1637822
## ... New best solution
## ... Procrustes: rmse 0.1922346  max resid 0.5174212
## Run 6 stress 0.1637822
## ... New best solution
## ... Procrustes: rmse 1.517764e-05  max resid 3.803984e-05
## ... Similar to previous best
## Run 7 stress 0.1717557
## Run 8 stress 0.2011703
## Run 9 stress 0.1945771
## Run 10 stress 0.1637822
## ... Procrustes: rmse 3.99423e-05  max resid 8.286952e-05
## ... Similar to previous best
## Run 11 stress 0.172685
## Run 12 stress 0.1773316
## Run 13 stress 0.1637822
## ... Procrustes: rmse 6.311237e-06  max resid 1.048157e-05
## ... Similar to previous best
## Run 14 stress 0.1956821
## Run 15 stress 0.186658
## Run 16 stress 0.1995539
## Run 17 stress 0.1956821
## Run 18 stress 0.2011703
## Run 19 stress 0.200963
## Run 20 stress 0.2148076
## *** Best solution repeated 3 times
```

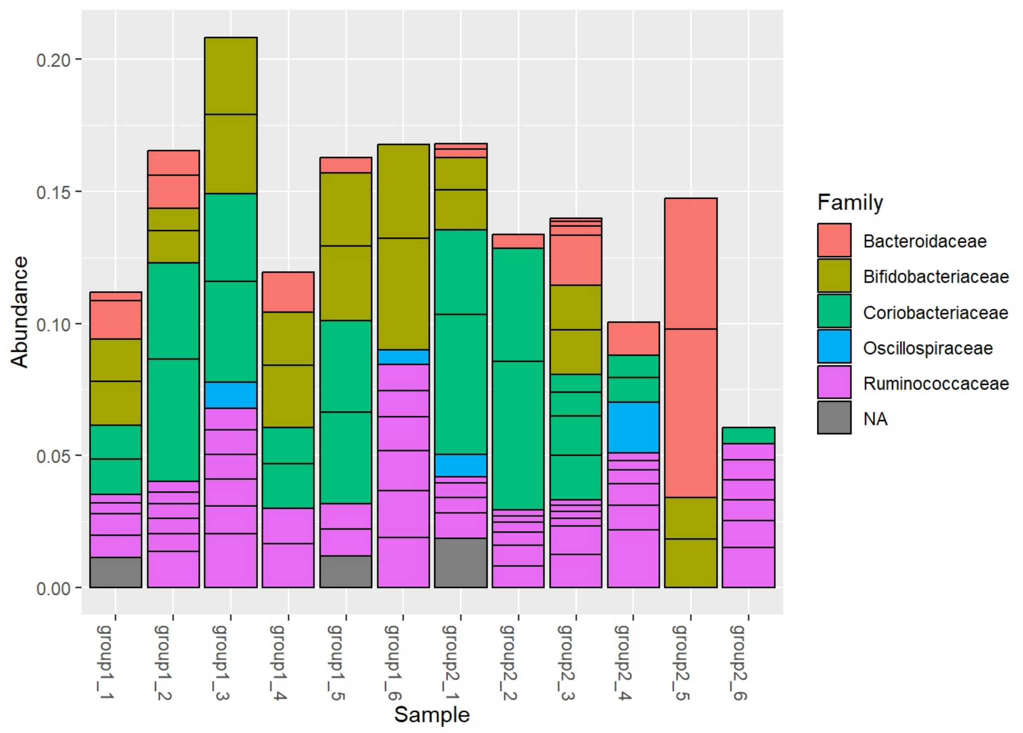
```
## Warning in postMDS(out$points, dis, plot = max(0, plot - 1), ...): skipping
## half-change scaling: too few points below threshold
```

```
plot_ordination(ps.prop, ord.nmds.bray, title="Bray NMDS")
```

```
## No available covariate data to map on the points for this plot `type`
```



```
top20 <- names(sort(taxa_sums(ps), decreasing=TRUE))[1:20]
ps.top20 <- transform_sample_counts(ps, function(OTU) OTU/sum(OTU))
ps.top20 <- prune_taxa(top20, ps.top20)
plot_bar(ps.top20, fill="Family")
```



```
abundances_table <- function(ps, seqtab, rank){
    rank_ps <- tax_glom(ps, rank)
    otu_df <- as.data.frame(t(otu_table(rank_ps)))
    taxa_table <- tax_table(rank_ps)
    taxa_table <- taxa_table[,colSums(is.na(taxa_table))<nrow(taxa_table)] # all rows are saved here
    # rownames(otu_df) <- apply(taxa_table, 1, paste, collapse=";")
    if (rank=='Species'){
      otu_df["taxa"] <- apply(taxa_table[, c('Genus', 'Species')], 1, paste, collapse="_")
    }
    else {
      otu_df["taxa"] <- taxa_table[, rank]
    }
    otu_df <- otu_df %>% group_by(taxa) %>% summarise_all(funs(sum))
    otu_df <- as.data.frame(otu_df)
    rownames(otu_df) <- otu_df$taxa
    otu_df <- otu_df[, !(names(otu_df) %in% c("taxa"))]
    otu_df["Unclassified", ] <-  rowSums(seqtab) - colSums(otu_df)
    otu_df <- (t(apply(otu_df, 1, function(x) round(x/colSums(otu_df), digits=8)*100)))

    return(otu_df[order(rowMeans(otu_df), decreasing = TRUE), ])
}
```

```
species_table <- abundances_table(ps, seqtab.nochim, "Species")
genus_table <- abundances_table(ps, seqtab.nochim, "Genus")
family_table <- abundances_table(ps, seqtab.nochim, "Family")
class_table <- abundances_table(ps, seqtab.nochim, "Class")
phylum_table <- abundances_table(ps, seqtab.nochim, "Phylum")
```

```
write.csv(species_table, file=paste0("results/", 'Species.csv'), quote=FALSE)
write.csv(species_table, file=paste0("results/", 'Genus.csv'), quote=FALSE)
write.csv(species_table, file=paste0("results/", 'Family.csv'), quote=FALSE)
write.csv(species_table, file=paste0("results/", 'Class.csv'), quote=FALSE)
write.csv(species_table, file=paste0("results/", 'Phylum.csv'), quote=FALSE)
```