Wastewater AMR in Latvia

Edgars Liepa

26/03/2025

```
library("phyloseq")
library("vegan")
```

```
## Loading required package: permute
```

```
## Loading required package: lattice
```

```
## This is vegan 2.6-8
```

```
library("dplyr")
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
suppressPackageStartupMessages(library(microViz))
library("ggplot2")
library(jsonlite)
library(patchwork)
library(stringr)
library(tidyr)
```

For resistance gene annotations CARD database was used.

```
CARDDB = "~/DB/CARD_02_2024/card.json"

METADATA_PATH =  "../sampleMetadata.csv"
metadata <- read.csv(METADATA_PATH, header = TRUE, colClasses = c(Date = "character", Dairy_farming="cha
```

ARG abundance

Read data resistance data.

AMR genes were classified in metagenome assemblies using Resistance gene finder.

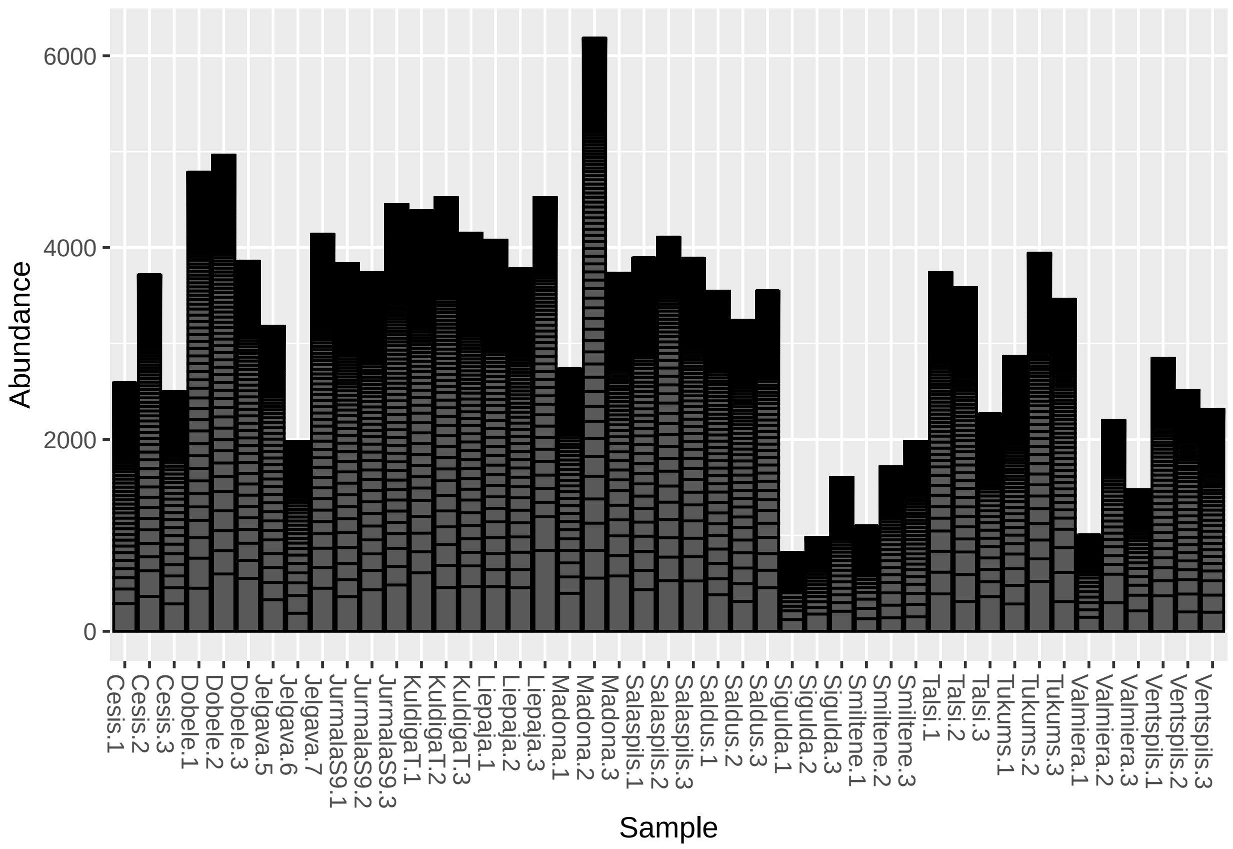
Then genes were quantified by aligning short read sequencing reads to the assemblies and sequenceing read overlap was counted in positions with ht-seq.

```
ARG_SCAFFOLDS_F = "../Resistance_genes/AMR_genes_RGI_scaffolds_filtered.csv"
arg_t <- read.csv(ARG_SCAFFOLDS_F, header = TRUE, row.names = 1)


tax_t <- data.frame(Tax = rownames(arg_t), ARG = rownames(arg_t))
  # tax_t <- sanitizeRowNames(tax_t)
rownames(tax_t) <- tax_t$Tax

ARG_TAX <- tax_table(as.matrix(tax_t[, c('Tax', 'ARG')]))

ps_amr <- phyloseq(otu_table(arg_t, taxa_are_rows = TRUE), sample_data(metadata), ARG_TAX)

# Fix Sample names
# Create a named vector for mapping
name_mapping <- setNames(metadata$Sample, rownames(metadata))

# Use the mapping to rename the columns
sample_names(ps_amr) <- name_mapping[sample_names(ps_amr)]
```

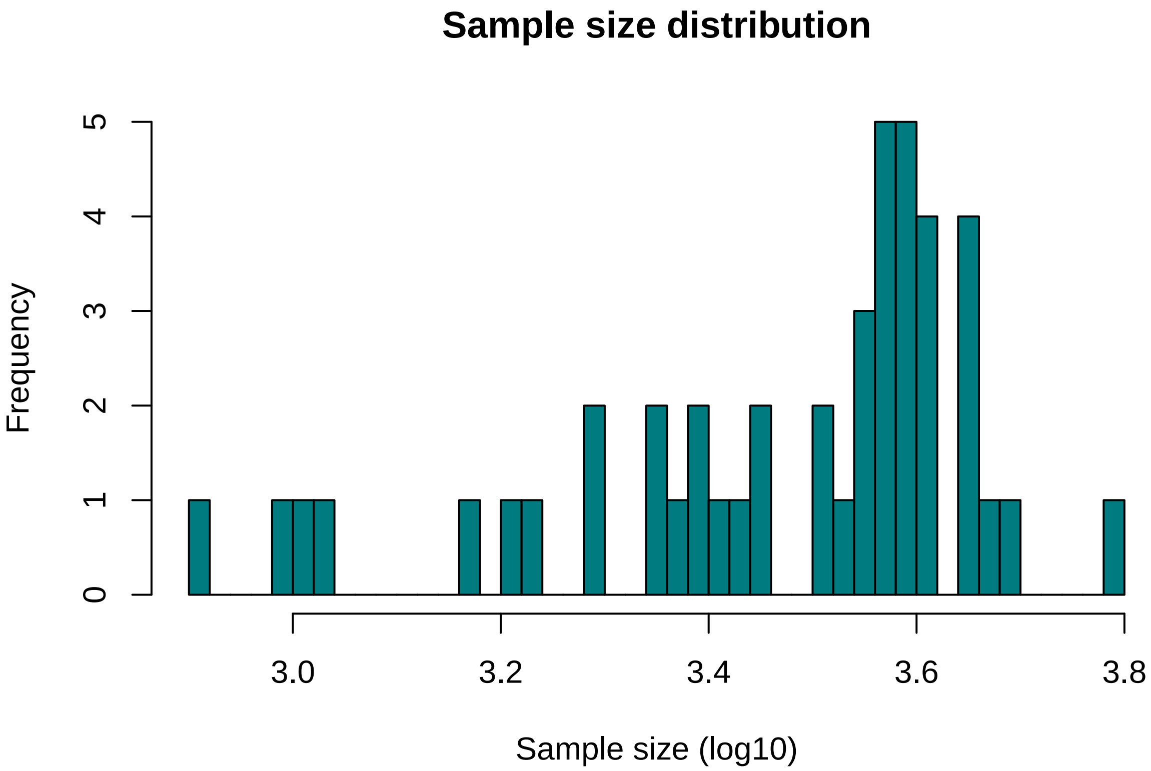
Display amount of sequencing reads alligning to a AMR genes found in contigs.

```
plot_bar(ps_amr)
```

```
## Warning in psmelt(physeq): The sample variables:
## Sample
##  have been renamed to:
## sample_Sample
## to avoid conflicts with special phyloseq plot attribute names.
```



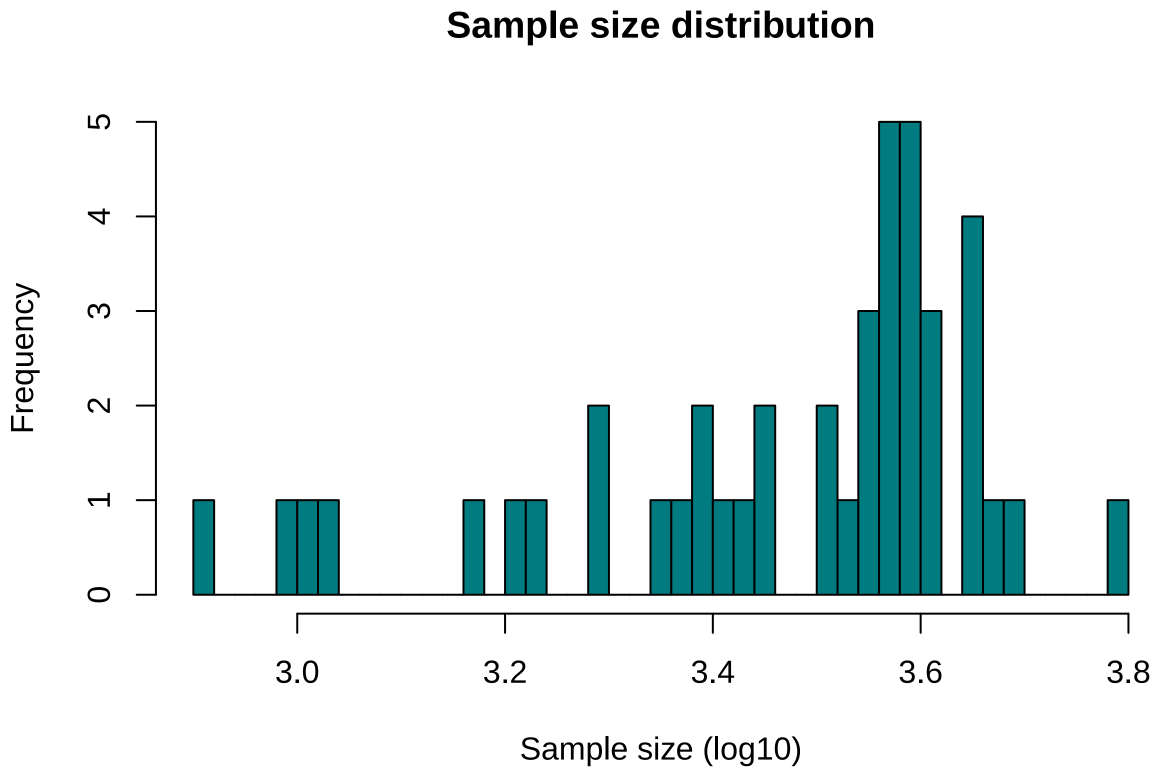
```
options(repr.plot.width=4, repr.plot.height=4)
hist(log10(sample_sums(ps_amr)), breaks=50, main="Sample size distribution", xlab="Sample size (log10)"
```



### Filtering

Since we removed some samples from bacterial anlysis, we have to remove them also here.

```
ps_amr_good = subset_samples(ps_amr, sample_data(ps_amr)$Sample != "Salaspils.2")
ps_amr_good = subset_samples(ps_amr_good, sample_data(ps_amr_good)$Sample != "Talsi.3")

hist(log10(sample_sums(ps_amr_good)), breaks=50, main="Sample size distribution", xlab="Sample size (log
```

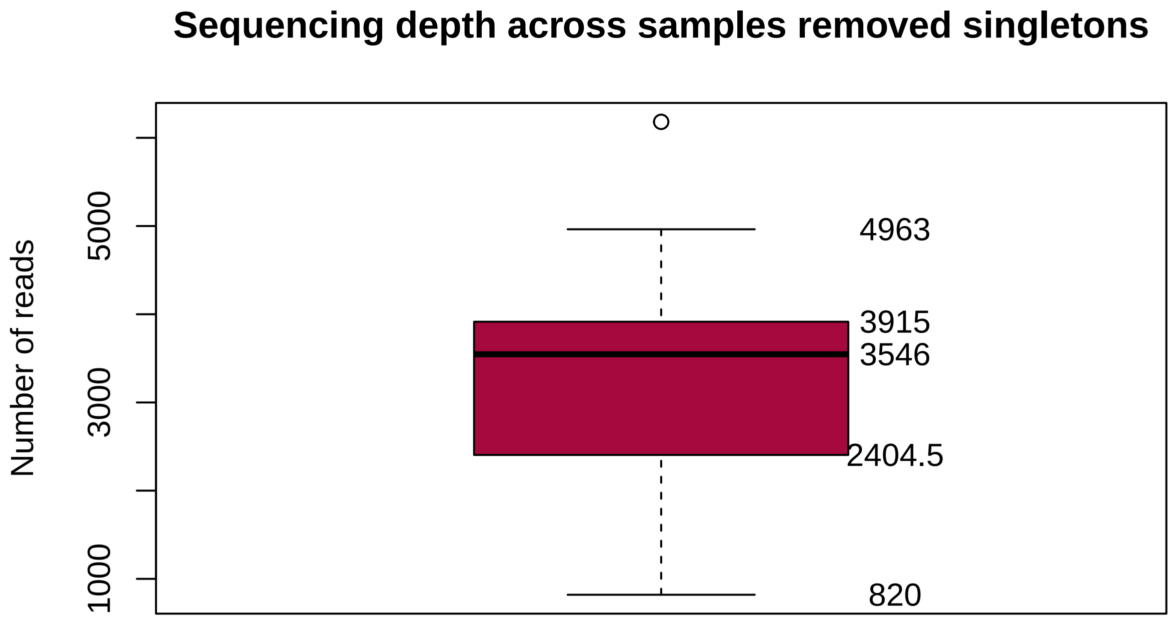


Now we remove singleton reads that are assigned to genes with only one alligned read across all samples.

```
# Less strict filterring option
# remove singletons.
ps_scaffolds_filtered = filter_taxa(ps_amr_good, function(x) sum(x) > 1, prune=TRUE)


max_difference = max(sample_sums(ps_scaffolds_filtered))/min(sample_sums(ps_scaffolds_filtered))

# The max difference in sequencing depth between largest and smallest sample
max_difference
```

```
## [1] 7.537805
```

```
boxplot(sample_sums(ps_scaffolds_filtered), main="Sequencing depth across samples removed singletons", 
text(y=boxplot.stats(sample_sums(ps_scaffolds_filtered))$stats, labels=boxplot.stats(sample_sums(ps_sca
```



Print the dimensions of the filtered data Number of genes found in samples after filtering

```
dim(data.frame(otu_table(ps_scaffolds_filtered)))
```

```
## [1] 417  43
```

Print the total sequence count after filtering Number of sequences in all samples

```
sum(data.frame(otu_table(ps_scaffolds_filtered)))
```

```
## [1] 138009
```

Calculate and print the percentage of sequences dropped from the original dataset

```
original_seq_count = sum(data.frame(otu_table(ps_amr_good)))
filtered_seq_count = sum(data.frame(otu_table(ps_scaffolds_filtered)))
seq_dropped_percentage = ((original_seq_count - filtered_seq_count) / original_seq_count) * 100
```

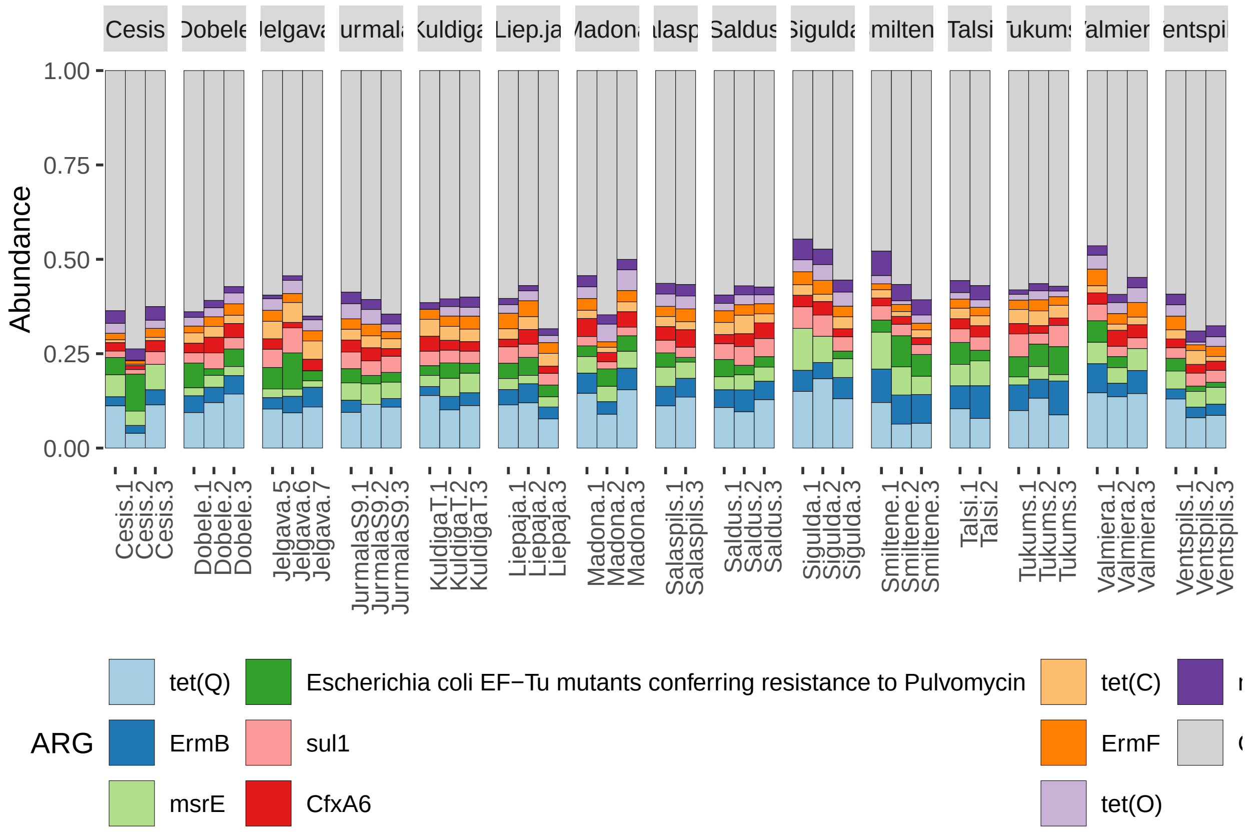
```
# Sequence % dropped from the dataset:
seq_dropped_percentage
```

```
## [1] 0.01376532
```

Show AMR gene distribution across samples

```
ps_scaffolds_filtered %>%
    ps_mutate(Group = factor(sample_data(ps_scaffolds_filtered)$City)) %>%
    tax_agg("ARG") %>%
    ps_seriate(dist = "bray", method = "OLO_ward") %>%
    comp_barplot(tax_level = "ARG",
                 sample_order = rownames(sample_data(ps_scaffolds_filtered)[order(sample_data(ps_scaffol
                 n_taxa = 10,
                 label = "Sample") +
    theme(axis.text.x = element_text(angle = 90, hjust = 1), legend.position = "bottom")+
    facet_grid(~Group, scales = "free", space = "free")
```

```
## Registered S3 method overwritten by 'seriation':
##   method         from 
##   reorder.hclust vegan

## Short values detected in phyloseq tax_table (nchar<4) :
## Consider using tax_fix() to make taxa uniquely identifiable
## Short values detected in phyloseq tax_table (nchar<4) :
## Consider using tax_fix() to make taxa uniquely identifiable
```



CARD DB gene anotations

Using CARD DB we take reconstructed genes to assign for witch AMR group, AMR familie and AMR category they belong.

```r
source("~/ARG_waste_water/src/comperative_metagen_definitions.r")
```

```
## Loading required package: S4Vectors
```

```
## Loading required package: stats4
```

```
## Loading required package: BiocGenerics
```

```
##
## Attaching package: 'BiocGenerics'
```

```
## The following objects are masked from 'package:dplyr':
##
##     combine, intersect, setdiff, union
```

```
## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs
```

```
## The following objects are masked from 'package:base':
##
##     Filter, Find, Map, Position, Reduce, anyDuplicated, aperm, append,
##     as.data.frame, basename, cbind, colnames, dirname, do.call,
##     duplicated, eval, evalq, get, grep, grepl, intersect, is.unsorted,
##     lapply, mapply, match, mget, order, paste, pmax, pmax.int, pmin,
##     pmin.int, rank, rbind, rownames, sapply, saveRDS, setdiff, table,
##     tapply, union, unique, unsplit, which.max, which.min
```

```
##
## Attaching package: 'S4Vectors'
```

```
## The following object is masked from 'package:tidyr':
##
##     expand
```

```
## The following objects are masked from 'package:dplyr':
##
##     first, rename
```

```
## The following object is masked from 'package:utils':
##
##     findMatches
```

```
## The following objects are masked from 'package:base':
##
##     I, expand.grid, unname
```

```
## Loading required package: IRanges
```

```
##
## Attaching package: 'IRanges'

## The following objects are masked from 'package:dplyr':
##
##     collapse, desc, slice

## The following object is masked from 'package:phyloseq':
##
##     distance

## Loading required package: GenomicRanges

## Loading required package: GenomeInfoDb

## Warning: multiple methods tables found for 'sort'

## Loading required package: SummarizedExperiment

## Loading required package: MatrixGenerics

## Loading required package: matrixStats

##
## Attaching package: 'matrixStats'

## The following object is masked from 'package:dplyr':
##
##     count

##
## Attaching package: 'MatrixGenerics'

## The following objects are masked from 'package:matrixStats':
##
##     colAlls, colAnyNAs, colAnys, colAvgsPerRowSet, colCollapse,
##     colCounts, colCummaxs, colCummins, colCumprods, colCumsums,
##     colDiffs, colIQRDiffs, colIQRs, colLogSumExps, colMadDiffs,
##     colMads, colMaxs, colMeans2, colMedians, colMins, colOrderStats,
##     colProds, colQuantiles, colRanges, colRanks, colSdDiffs, colSds,
##     colSums2, colTabulates, colVarDiffs, colVars, colWeightedMads,
##     colWeightedMeans, colWeightedMedians, colWeightedSds,
##     colWeightedVars, rowAlls, rowAnyNAs, rowAnys, rowAvgsPerColSet,
##     rowCollapse, rowCounts, rowCummaxs, rowCummins, rowCumprods,
##     rowCumsums, rowDiffs, rowIQRDiffs, rowIQRs, rowLogSumExps,
##     rowMadDiffs, rowMads, rowMaxs, rowMeans2, rowMedians, rowMins,
##     rowOrderStats, rowProds, rowQuantiles, rowRanges, rowRanks,
##     rowSdDiffs, rowSds, rowSums2, rowTabulates, rowVarDiffs, rowVars,
##     rowWeightedMads, rowWeightedMeans, rowWeightedMedians,
##     rowWeightedSds, rowWeightedVars
## Loading required package: Biobase

## Welcome to Bioconductor
##
##     Vignettes contain introductory material; view with
##     'browseVignettes()'. To cite Bioconductor, see
##     'citation("Biobase")', and for packages 'citation("pkgname")'.

##
## Attaching package: 'Biobase'

## The following object is masked from 'package:MatrixGenerics':
##
##     rowMedians

## The following objects are masked from 'package:matrixStats':
##
##     anyMissing, rowMedians

## The following object is masked from 'package:phyloseq':
##
##     sampleNames

## Warning: multiple methods tables found for 'sort'

## Loading required package: zCompositions

## Loading required package: MASS

##
## Attaching package: 'MASS'

## The following object is masked from 'package:patchwork':
##
##     area

## The following object is masked from 'package:dplyr':
##
##     select

## Loading required package: NADA

## Loading required package: survival

##
## Attaching package: 'NADA'

## The following object is masked from 'package:IRanges':
##
##     cor
## The following object is masked from 'package:S4Vectors':
##
##     cor

## The following object is masked from 'package:stats':
##
##     cor

## Loading required package: truncnorm

## Loading required package: latticeExtra

##
## Attaching package: 'latticeExtra'

## The following object is masked from 'package:ggplot2':
##
##     layer

##
## Attaching package: 'igraph'

## The following object is masked from 'package:GenomicRanges':
##
##     union

## The following object is masked from 'package:IRanges':
##
##     union

## The following object is masked from 'package:S4Vectors':
##
##     union

## The following objects are masked from 'package:BiocGenerics':
##
##     normalize, path, union

## The following object is masked from 'package:tidyr':
##
##     crossing

## The following objects are masked from 'package:dplyr':
##
##     as_data_frame, groups, union

## The following object is masked from 'package:vegan':
##
##     diversity
## The following object is masked from 'package:permute':
##
##     permute

## The following objects are masked from 'package:stats':
##
##     decompose, spectrum

## The following object is masked from 'package:base':
##
##     union

##
## Attaching package: 'SpiecEasi'

## The following object is masked from 'package:igraph':
##
##     make_graph

## The following object is masked from 'package:MASS':
##
##     fitdistr
```

```
card_tax_t <- NA
card_tax_t <- process_card_data(arg_t)

#
# SEPERATE DRUG CLASSES INTO INDIVIDUAL NAMES
#

# Assuming tax_t is your data frame and DrugClass is the column of interest
# unique_drug_classes
unique_drug_classes <- card_tax_t %>%
  # Separate the DrugClass column into rows by splitting on ";"
  separate_rows(DrugClass, sep = ";") %>%
  # Select unique DrugClass names
  distinct(DrugClass) %>%
  # Pull the DrugClass column to get a vector of unique drug class names
  pull(DrugClass)




#unique_amr_families
unique_amr_families <- card_tax_t %>%
  # Separate the DrugClass column into rows by splitting on ";"
  separate_rows(AMRGeneFamily, sep = ";") %>%
  # Select unique DrugClass names
  distinct(AMRGeneFamily) %>%
  # Pull the DrugClass column to get a vector of unique drug class names
  pull(AMRGeneFamily)
```

```r
#unique_amr_categories
unique_amr_categories <- card_tax_t %>%
  # Separate the DrugClass column into rows by splitting on ";"
  separate_rows(AntibioticCategory, sep = ";") %>%
  # Select unique DrugClass names
  distinct(AntibioticCategory) %>%
  # Pull the DrugClass column to get a vector of unique drug class names
  pull(AntibioticCategory)
```

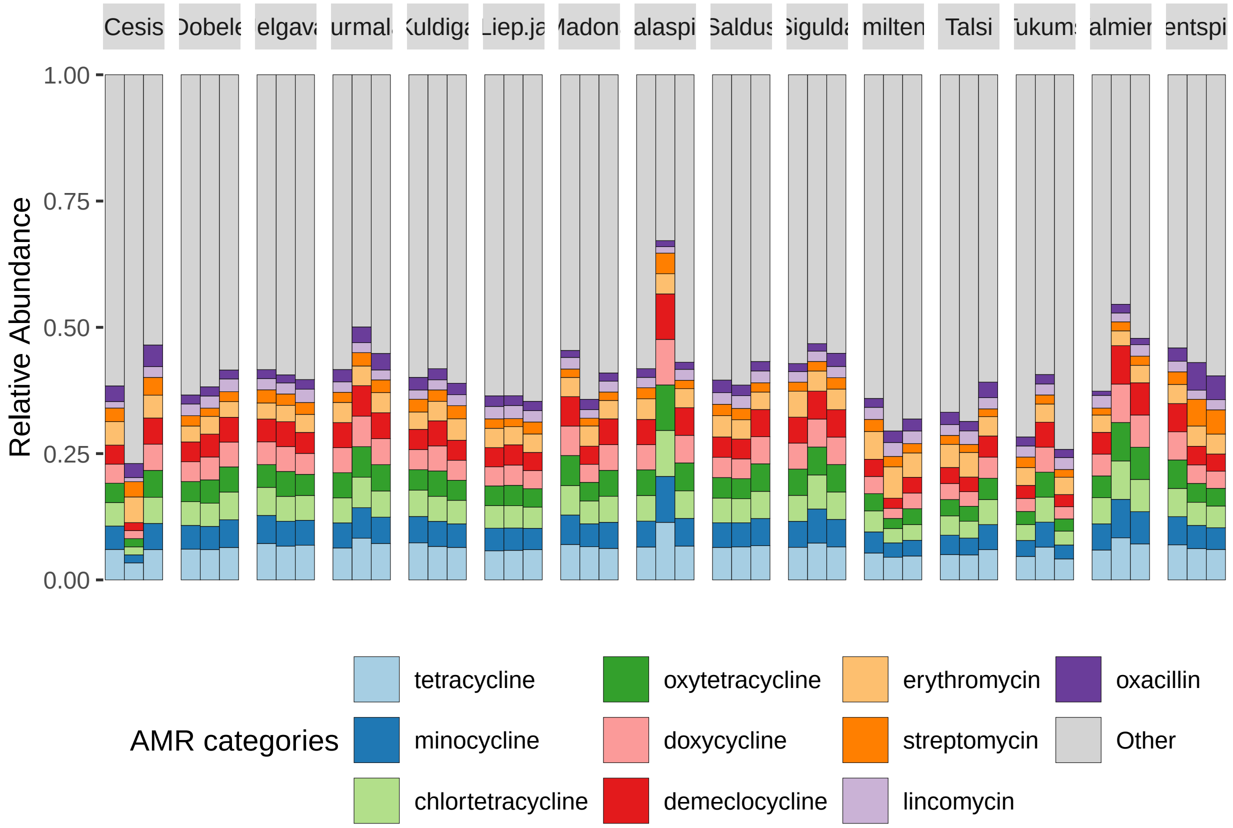
Create Objects for AMR categories.

```r
drug_class_counts <- create_drug_class_counts(arg_t, card_tax_t, unique_drug_classes)

# Create the AMR families counts
amr_families_counts <- create_amr_families_counts(arg_t, card_tax_t, unique_amr_families)

# Create the AMR categories counts
amr_categories_counts <- create_amr_categories_counts(arg_t, card_tax_t, unique_amr_categories)

# Create the taxonomy table
tax_t_drugs <- create_taxonomy_table(card_tax_t)

physeq_drug_class <- createPhyloseqObject(drug_class_counts, METADATA_PATH)
physeq_amr_families <- createPhyloseqObject(amr_families_counts, METADATA_PATH)
physeq_amr_categories <- createPhyloseqObject(amr_categories_counts, METADATA_PATH)
```

Show to what Drug classes resistance genes found in metagenome are resistant to.

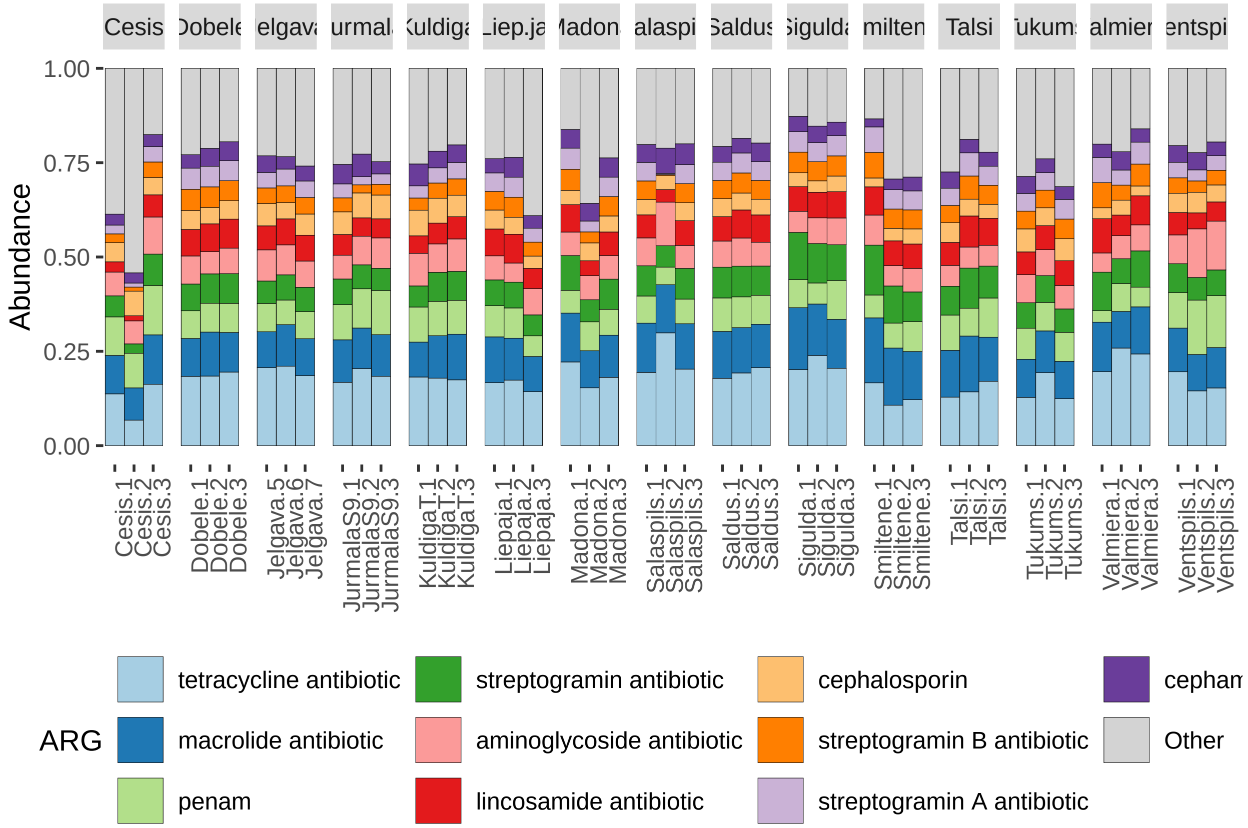
```r
physeq_amr_categories %>%
    ps_mutate(Group = factor(sample_data(physeq_amr_categories)$City)) %>%
    tax_agg("ARG") %>%
    ps_seriate(dist = "bray", method = "OLO_ward") %>%
    comp_barplot(tax_level = "ARG",
                 sample_order = rownames(metadata[order(metadata$Sample), ]),
                 n_taxa = 10,
                 label = "Sample", interactive = TRUE) +
    theme(axis.text.x = element_blank(), axis.ticks.x = element_blank(), legend.position = "bottom")+
    facet_grid(~Group, scales = "free", space = "free")+
    labs(y = "Relative Abundance", fill = "AMR categories")
```

```
## Short values detected in phyloseq tax_table (nchar<4) :
## Consider using tax_fix() to make taxa uniquely identifiable
## Short values detected in phyloseq tax_table (nchar<4) :
## Consider using tax_fix() to make taxa uniquely identifiable
```



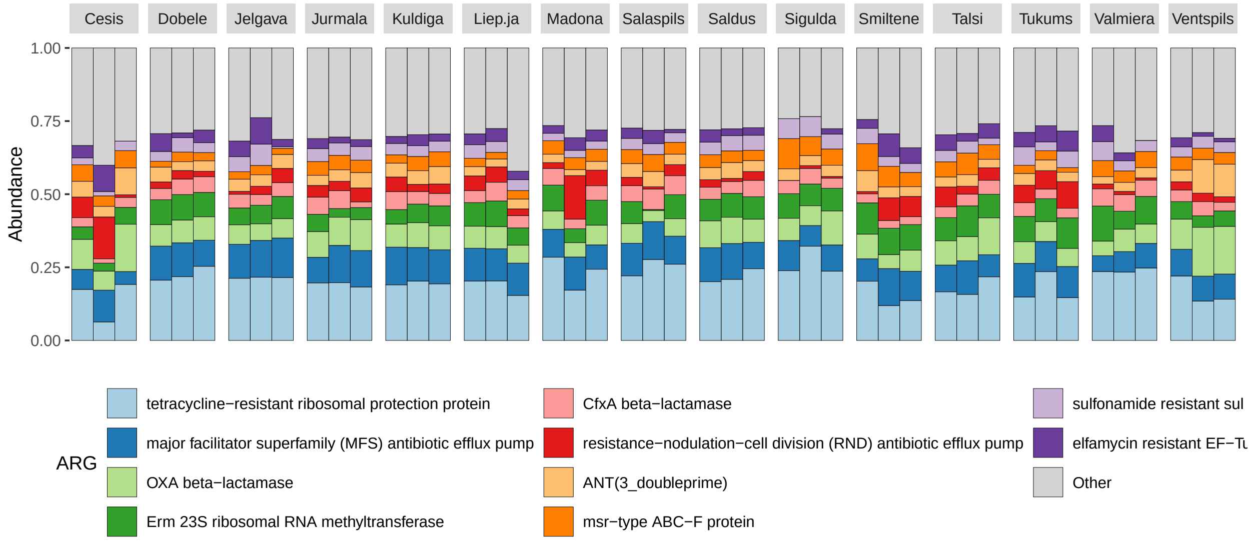
```
physeq_drug_class %>%
    ps_mutate(Group = factor(sample_data(physeq_drug_class)$City)) %>%
    tax_agg("ARG") %>%
    ps_seriate(dist = "bray", method = "OLO_ward") %>%
    comp_barplot(tax_level = "ARG",
                 sample_order = rownames(metadata[order(metadata$Sample), ]),
                 n_taxa = 10,
                 label = "Sample", interactive = TRUE) +
    theme(axis.text.x = element_text(angle = 90, hjust = 1), legend.position = "bottom")+
    facet_grid(~Group, scales = "free", space = "free")
```

```
## Warning in ps_melt(ps): The sample variables:
## Sample
##  have been renamed to:
## sample_Sample
## to avoid conflicts with special phyloseq plot attribute names.
```



```r
physeq_amr_families %>%
    ps_mutate(Group = factor(sample_data(physeq_amr_families)$City)) %>%
    tax_agg("ARG") %>%
    ps_seriate(dist = "bray", method = "OLO_ward") %>%
    comp_barplot(tax_level = "ARG",
                 sample_order = rownames(metadata[order(metadata$Sample), ]),
                 n_taxa = 10,
                 label = "Sample", interactive = TRUE) +
    theme(
      axis.text.x = element_blank(),
      axis.ticks.x = element_blank(),
      legend.position = "bottom"
      )+
    guides(fill = guide_legend(ncol = 3))+
    facet_grid(~Group, scales = "free", space = "free")
```

```
## Warning in ps_melt(ps): The sample variables:
## Sample
##  have been renamed to:
## sample_Sample
## to avoid conflicts with special phyloseq plot attribute names.
```



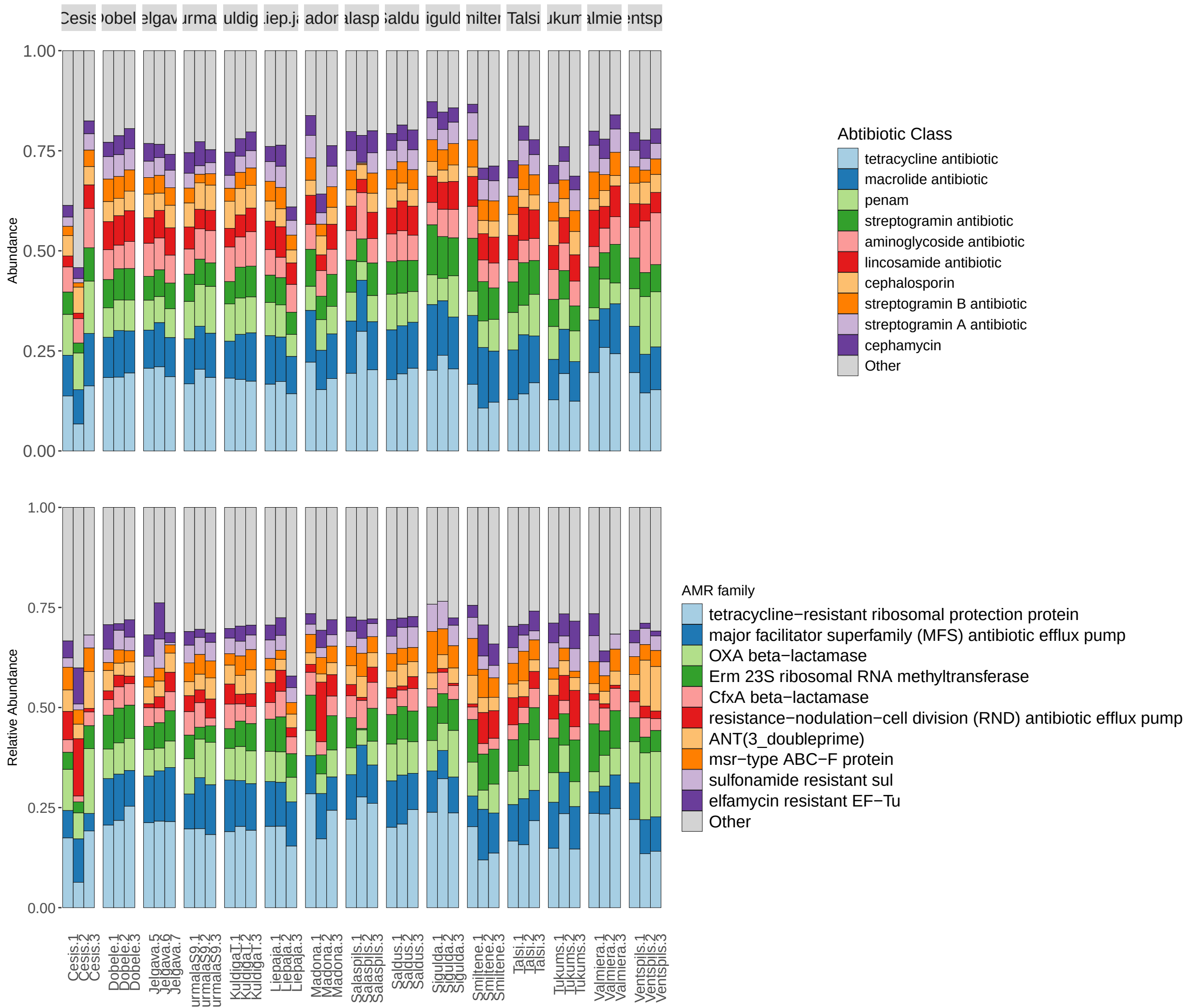
```
# Plot 1 (physeq_drug_class - renamed for clarity)
p1 <- physeq_drug_class %>%
  ps_mutate(Group = factor(sample_data(physeq_drug_class)$City)) %>%
  tax_agg("ARG") %>%
  ps_seriate(dist = "bray", method = "OLO_ward") %>%
  comp_barplot(tax_level = "ARG",
               sample_order = rownames(metadata[order(metadata$Sample), ]),
               n_taxa = 10,
               label = "Sample", interactive = FALSE) + # Turn off interactive for combination
  theme(
        legend.position = "right",
        axis.title.x = element_blank(),
        axis.text.x = element_blank(),
        axis.ticks.x = element_blank(),
        axis.text.y = element_text(size = 14),
        legend.text = element_text(size = 12),
        legend.title = element_text(size = 14),
        strip.text = element_text(size = 14),
  ) +
  facet_grid(~Group, scales = "free", space = "free") +
  labs(fill = "Abtibiotic Class")
```

```
## Warning in ps_melt(ps): The sample variables:
## Sample
##  have been renamed to:
## sample_Sample
## to avoid conflicts with special phyloseq plot attribute names.
```

```
# Plot 2 (physeq_amr_families - renamed for clarity)
p2 <- physeq_amr_families %>%
  ps_mutate(Group = factor(sample_data(physeq_amr_families)$City)) %>%
  tax_agg("ARG") %>%
  ps_seriate(dist = "bray", method = "OLO_ward") %>%
  comp_barplot(tax_level = "ARG",
               sample_order = rownames(metadata[order(metadata$Sample), ]),
               n_taxa = 10,
               label = "Sample", interactive = FALSE) +  # Turn off interactive
  theme(
        axis.text.x = element_text(angle = 90, hjust = 1, size = 12), # Smaller x-axis text
        axis.ticks.x = element_blank(),
        axis.text.y = element_text(size = 12),
        strip.text = element_blank(),
        legend.position = "right",
        axis.title.x = element_blank(),
        legend.text = element_text(size = 14),
        legend.title = element_text(size = 12)

  ) + # Remove x-axis title
  facet_grid(~Group, scales = "free", space = "free") +
  ylab("Relative Abundance") +   # Clear y-axis label
  labs(fill = "AMR family")
```

```
## Warning in ps_melt(ps): The sample variables:
## Sample
##  have been renamed to:
## sample_Sample
## to avoid conflicts with special phyloseq plot attribute names.
```

```
# Combine the plots vertically using patchwork
combined_plot <- p1 / p2

print(combined_plot)
```



Show gene distribution

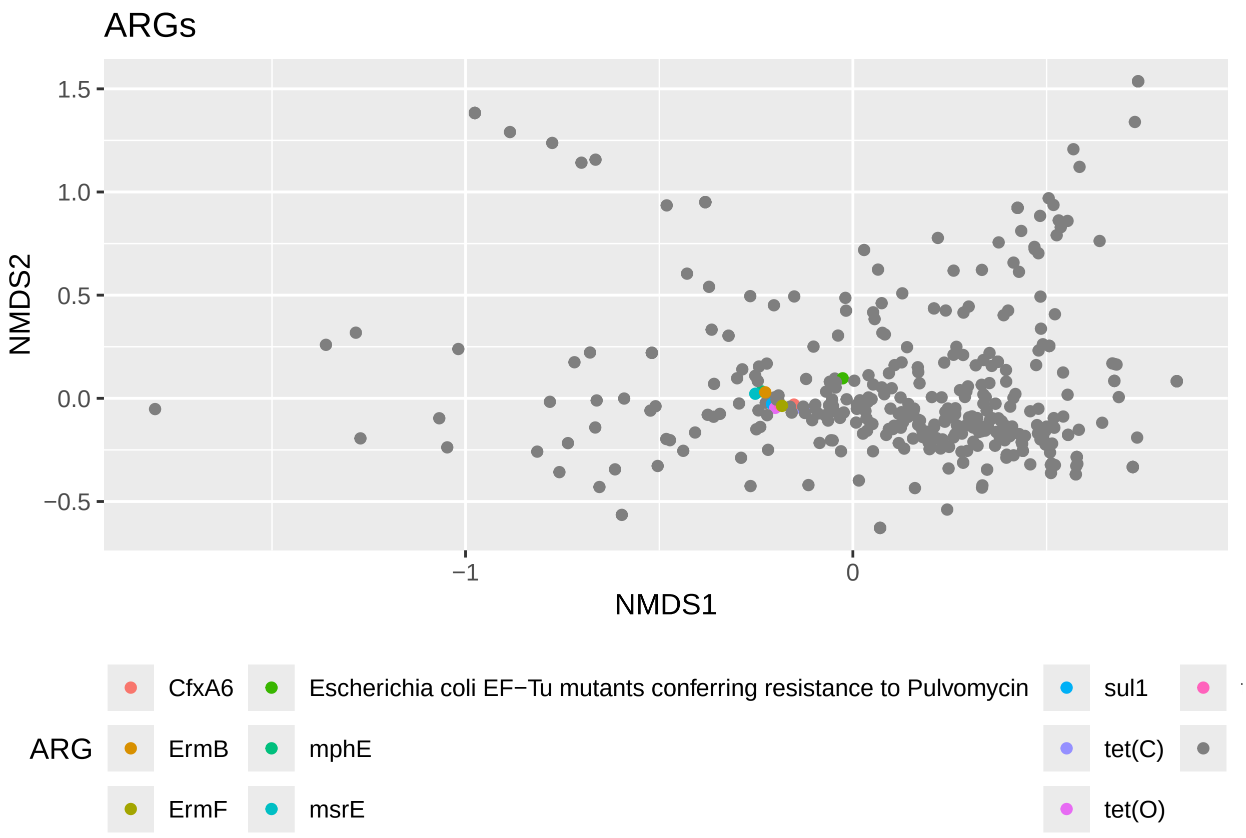
```
GP1 = transform_sample_counts(ps_scaffolds_filtered, function(x) 1E6 * x/sum(x))

arg.sum = tapply(taxa_sums(GP1), tax_table(GP1)[, "ARG"], sum, na.rm=TRUE)
top10args = names(sort(arg.sum, TRUE))[1:10]
GP1 = prune_taxa((tax_table(GP1)[, "ARG"] %in% top10args), GP1)

GP.ord <- ordinate(GP1, "NMDS", "bray")
```

```
## Square root transformation
## Wisconsin double standardization
## Run 0 stress 0.1459272
## Run 1 stress 0.1850934
## Run 2 stress 0.1525538
## Run 3 stress 0.145776
## ... New best solution
## ... Procrustes: rmse 0.01179365  max resid 0.07033017
## Run 4 stress 0.1525538
## Run 5 stress 0.1462751
## ... Procrustes: rmse 0.02390504  max resid 0.1134582
## Run 6 stress 0.1887685
## Run 7 stress 0.1462791
## Run 8 stress 0.1525538
## Run 9 stress 0.1462751
## ... Procrustes: rmse 0.0239098  max resid 0.1134781
## Run 10 stress 0.1887686
## Run 11 stress 0.1468094
## Run 12 stress 0.1612656
## Run 13 stress 0.1459272
## ... Procrustes: rmse 0.01179442  max resid 0.07040723
## Run 14 stress 0.1514297
## Run 15 stress 0.2111069
## Run 16 stress 0.1468094
## Run 17 stress 0.1888131
## Run 18 stress 0.2156544
## Run 19 stress 0.1809571
## Run 20 stress 0.1468094
## *** Best solution was not repeated -- monoMDS stopping criteria:
##     20: stress ratio > sratmax
```

```
GP.ord2 <- ordinate(ps_scaffolds_filtered, "NMDS", "bray")
```

```
## Square root transformation
## Wisconsin double standardization
## Run 0 stress 0.137757
## Run 1 stress 0.1797281
## Run 2 stress 0.1975882
## Run 3 stress 0.1718425
## Run 4 stress 0.1927692
## Run 5 stress 0.186725
## Run 6 stress 0.1854681
## Run 7 stress 0.1381956
## ... Procrustes: rmse 0.02197373  max resid 0.1298108
## Run 8 stress 0.1963749
## Run 9 stress 0.1434782
## Run 10 stress 0.1720912
## Run 11 stress 0.1751035
## Run 12 stress 0.1767703
## Run 13 stress 0.1377036
## ... New best solution
## ... Procrustes: rmse 0.004230481  max resid 0.02448645
## Run 14 stress 0.1748303
## Run 15 stress 0.143372
## Run 16 stress 0.1849371
## Run 17 stress 0.1724141
## Run 18 stress 0.1377036
## ... Procrustes: rmse 6.012247e-05  max resid 0.0003395925
## ... Similar to previous best
## Run 19 stress 0.1803313
## Run 20 stress 0.1809799
## *** Best solution repeated 1 times
```

```r
plot_ordination(GP1, GP.ord2, type="taxa", color = "ARG", title="ARGs") + theme(legend.position = "bott
```



Normalization

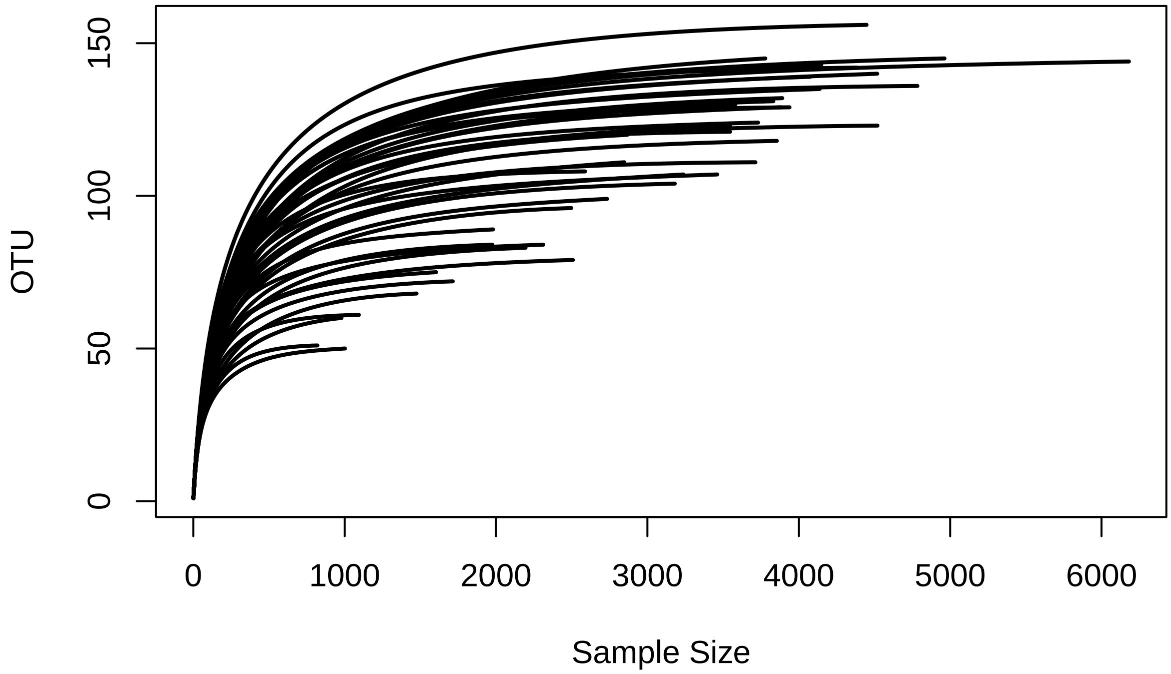
Normalization by subsampling (rarefaction)

Display rarefication curves.

```r
tab <- otu_table(ps_scaffolds_filtered)
class(tab) <- "matrix"
```

```
## Warning in class(tab) <- "matrix": Setting class(x) to "matrix" sets attribute
## to NULL; result will no longer be an S4 object
```

```r
tab <- t(tab)

rarecurve(tab, step=10, lwd=2, ylab="OTU",  label=F)
```



Rarefication if needed.

```
# find minimal deapth to rarefy to
df=as.data.frame(sample_sums(ps_scaffolds_filtered))
head(df[order( df[,1] ),],1)
```

```
## [1] 820
```

```
ps_amr_rare = rarefy_even_depth(ps_scaffolds_filtered, sample.size=820, replace=FALSE, rngseed=123, verb
```

Normalization by cumulative sum scaling (CSS)

```
ps_amr_CSS = microbiomeMarker::normalize(ps_scaffolds_filtered, method="CSS")
```

```
## Registered S3 methods overwritten by 'proxy':
##   method               from
##   print.registry_field registry
##   print.registry_entry registry
```

```
## Registered S3 method overwritten by 'gplots':
##   method         from
##   reorder.factor DescTools
```

```
## Default value being used.
```

Most abundant genes

Show Top 15 most abundant genes and their relative xount

```
genes_transformed  = transform_sample_counts(ps_scaffolds_filtered, function(x) x / sum(x) )

abu_table <- as.data.frame(otu_table(genes_transformed))
tax_table <- as.data.frame(tax_table(genes_transformed))
rownames(abu_table) <- tax_table$ARG

total_counts <- rowSums(abu_table)
top_organisms <- sort(total_counts, decreasing = TRUE)
top_organisms <- head(top_organisms, n = 15)

top_organisms
```

```
##                                                             tet(Q)
##                                                          4.8231801
##                                                               ErmB
##                                                          2.0248165
##                                                               msrE
##                                                          1.9219906
## Escherichia coli EF-Tu mutants conferring resistance to Pulvomycin
##                                                          1.6259683
##                                                               sul1
##                                                          1.5385914
##                                                              CfxA6
##                                                          1.2405375
##                                                             tet(C)
##                                                          1.1826839
##                                                               ErmF
##                                                          1.1664615
##                                                               mphE
##                                                          1.1369348
##                                                             tet(O)
##                                                          1.1287647
##                                                             tet(W)
##                                                          1.0439750
##                                                               aadS
##                                                          0.8678512
##                                                         tet(W/N/W)
##                                                          0.8451646
##                                                             OXA-10
##                                                          0.7624592
##                                                              CfxA2
##                                                          0.7579160
```

```
mean_percentage <- apply(abu_table, 1, mean) * 100
std_dev <- sqrt(apply(abu_table, 1, var)) * 100

top_genes <- sort(mean_percentage, decreasing = TRUE) %>% head(n = 15) %>% as.data.frame %>% rownames

# Format output with both mean percentage and standard deviation
```

```r
formatted_output <- sapply(top_genes, function(gene) {
  sprintf("%s (%.2f%% +/- %.2f%%)",
          gene,
          unname(mean_percentage[gene]),
          unname(std_dev[gene])
          )
})

# Display formatted output
formatted_output
```

```
##                                                                                   tet(Q)
##                                                              "tet(Q) (11.22% +/- 2.79%)"
##                                                                                     ErmB
##                                                                 "ErmB (4.71% +/- 1.97%)"
##                                                                                     msrE
##                                                                 "msrE (4.47% +/- 1.99%)"
##                      Escherichia coli EF-Tu mutants conferring resistance to Pulvomycin
## "Escherichia coli EF-Tu mutants conferring resistance to Pulvomycin (3.78% +/- 2.36%)"
##                                                                                     sul1
##                                                                 "sul1 (3.58% +/- 1.29%)"
##                                                                                    CfxA6
##                                                                "CfxA6 (2.88% +/- 0.88%)"
##                                                                                   tet(C)
##                                                               "tet(C) (2.75% +/- 1.12%)"
##                                                                                     ErmF
##                                                                 "ErmF (2.71% +/- 0.77%)"
##                                                                                     mphE
##                                                                 "mphE (2.64% +/- 1.14%)"
##                                                                                   tet(O)
##                                                               "tet(O) (2.63% +/- 1.10%)"
##                                                                                   tet(W)
##                                                               "tet(W) (2.43% +/- 2.32%)"
##                                                                                     aadS
##                                                                 "aadS (2.02% +/- 0.56%)"
##                                                                               tet(W/N/W)
##                                                           "tet(W/N/W) (1.97% +/- 1.94%)"
##                                                                                   OXA-10
##                                                               "OXA-10 (1.77% +/- 0.67%)"
##                                                                                    CfxA2
##                                                                "CfxA2 (1.76% +/- 0.78%)"
```

Show Drug classes that these genes are affecting

```r
abu_table <- as.data.frame(otu_table(physeq_drug_class))
tax_table <- as.data.frame(tax_table(physeq_drug_class))
rownames(abu_table) <- tax_table$ARG

total_counts <- rowSums(abu_table)
top_organisms <- sort(total_counts, decreasing = TRUE)
top_organisms <- head(top_organisms, n = 15)

top_organisms
```

```
##    tetracycline antibiotic       macrolide antibiotic
##                      43210                      28062
##                      penam   streptogramin antibiotic
##                      20576                      17811
##   aminoglycoside antibiotic     lincosamide antibiotic
##                      17632                      14842
##               cephalosporin streptogramin B antibiotic
##                      12319                      10667
## streptogramin A antibiotic                 cephamycin
##                      10648                      10639
## fluoroquinolone antibiotic                      penem
##                       7925                       6481
##                  carbapenem     sulfonamide antibiotic
##                       6150                       5981
##       elfamycin antibiotic
##                       5704
```

```
# Calculate the percentage of the top 15 organisms
print("Percentage of top 15 organisms:")
```

```
## [1] "Percentage of top 15 organisms:"
```

```
top_organisms/sum(rowSums(abu_table))*100
```

```
##    tetracycline antibiotic       macrolide antibiotic
##                  17.339904                  11.261106
##                      penam   streptogramin antibiotic
##                   8.257021                   7.147443
##   aminoglycoside antibiotic     lincosamide antibiotic
##                   7.075612                   5.956002
##               cephalosporin streptogramin B antibiotic
##                   4.943538                   4.280601
## streptogramin A antibiotic                 cephamycin
##                   4.272976                   4.269364
## fluoroquinolone antibiotic                      penem
##                   3.180253                   2.600785
##                  carbapenem     sulfonamide antibiotic
##                   2.467957                   2.400138
##       elfamycin antibiotic
##                   2.288980
```

Show most common amr_gene families.

```
abu_table <- as.data.frame(otu_table(physeq_amr_families))
tax_table <- as.data.frame(tax_table(physeq_amr_families))
rownames(abu_table) <- tax_table$ARG

total_counts <- rowSums(abu_table)
top_organisms <- sort(total_counts, decreasing = TRUE)
top_organisms <- head(top_organisms, n = 15)

top_organisms
```

```
##              tetracycline-resistant ribosomal protection protein
##                                                            31102
##       major facilitator superfamily (MFS) antibiotic efflux pump
##                                                            16371
##                                               OXA beta-lactamase
##                                                            12523
##                          Erm 23S ribosomal RNA methyltransferase
##                                                            10667
##                                              CfxA beta-lactamase
##                                                             6959
## resistance-nodulation-cell division (RND) antibiotic efflux pump
##                                                             6698
##                                               ANT(3_doubleprime)
##                                                             6676
##                                           msr-type ABC-F protein
##                                                             6365
##                                        sulfonamide resistant sul
##                                                             5981
##                                        elfamycin resistant EF-Tu
##                                                             5704
##                               macrolide phosphotransferase (MPH)
##                                                             4641
##                                                           ANT(6)
##                                                             3203
##                                       OXA-10-like beta-lactamase
##                                                             2799
##                                               APH(3_doubleprime)
##                                                             2281
##                         lincosamide nucleotidyltransferase (LNU)
##                                                             1877
```

Show most commonly affected drug classes.

```r
abu_table <- as.data.frame(otu_table(physeq_amr_categories))
tax_table <- as.data.frame(tax_table(physeq_amr_categories))
rownames(abu_table) <- tax_table$ARG

total_counts <- rowSums(abu_table)
top_organisms <- sort(total_counts, decreasing = TRUE)
top_organisms <- head(top_organisms, n = 15)

top_organisms
```

```
##      tetracycline       minocycline chlortetracycline   oxytetracycline
##             42841             32671             32638             30186
##       doxycycline    demeclocycline      erythromycin      streptomycin
##             29868             29868             26351             14770
##        lincomycin         oxacillin                NA      azithromycin
##             14681             14657             14406             13482
##     telithromycin    clarithromycin     roxithromycin
##             12037             11835             11748
```

Calculate mean relative abundance and variance

```r
genus_transformed_amr  = transform_sample_counts(physeq_amr_families, function(x) x / sum(x) )

abu_table_arg <- as.data.frame(otu_table(genus_transformed_amr))
tax_table_arg <- as.data.frame(tax_table(genus_transformed_amr))
rownames(abu_table_arg) <- tax_table_arg$Tax

variance <- apply(abu_table_arg, 1, var)
mean_percentage <- apply(abu_table_arg, 1, mean) * 100
std_dev <- sqrt(apply(abu_table_arg, 1, var)) * 100


top_arg <- sort(mean_percentage, decreasing = TRUE) %>% head(n = 15) %>% as.data.frame %>% rownames

# Format output with both mean percentage and standard deviation
formatted_output <- sapply(top_arg, function(arg) {
  sprintf("%s (%.2f%% +/- %.2f%%)",
          arg,
          unname(mean_percentage[arg]),
          unname(std_dev[arg])
          )
})

# Display formatted output
formatted_output
```

```
##                                  tetracycline-resistant ribosomal protection protein
##             "tetracycline-resistant ribosomal protection protein (20.29% +/- 4.73%)"
##                           major facilitator superfamily (MFS) antibiotic efflux pump
##      "major facilitator superfamily (MFS) antibiotic efflux pump (10.11% +/- 2.11%)"
##                                                                   OXA beta-lactamase
##                                               "OXA beta-lactamase (8.21% +/- 2.82%)"
##                                              Erm 23S ribosomal RNA methyltransferase
##                          "Erm 23S ribosomal RNA methyltransferase (7.14% +/- 2.27%)"
##                                                               msr-type ABC-F protein
##                                           "msr-type ABC-F protein (4.46% +/- 1.76%)"
##                                                                  CfxA beta-lactamase
##                                              "CfxA beta-lactamase (4.45% +/- 1.26%)"
##                                                                   ANT(3_doubleprime)
##                                               "ANT(3_doubleprime) (4.39% +/- 1.96%)"
##                                                            sulfonamide resistant sul
##                                        "sulfonamide resistant sul (4.05% +/- 1.39%)"
##                     resistance-nodulation-cell division (RND) antibiotic efflux pump
## "resistance-nodulation-cell division (RND) antibiotic efflux pump (3.83% +/- 3.19%)"
##                                                            elfamycin resistant EF-Tu
##                                        "elfamycin resistant EF-Tu (3.57% +/- 2.17%)"
##                                                   macrolide phosphotransferase (MPH)
##                               "macrolide phosphotransferase (MPH) (3.10% +/- 0.97%)"
##                                                                               ANT(6)
##                                                           "ANT(6) (2.05% +/- 0.59%)"
##                                                           OXA-10-like beta-lactamase
##                                      "OXA-10-like beta-lactamase (1.77% +/- 0.68%)"
##                                                                   APH(3_doubleprime)
##                                               "APH(3_doubleprime) (1.51% +/- 0.41%)"
##                                                      tetracycline inactivation enzyme
##                                  "tetracycline inactivation enzyme (1.38% +/- 2.49%)"
```

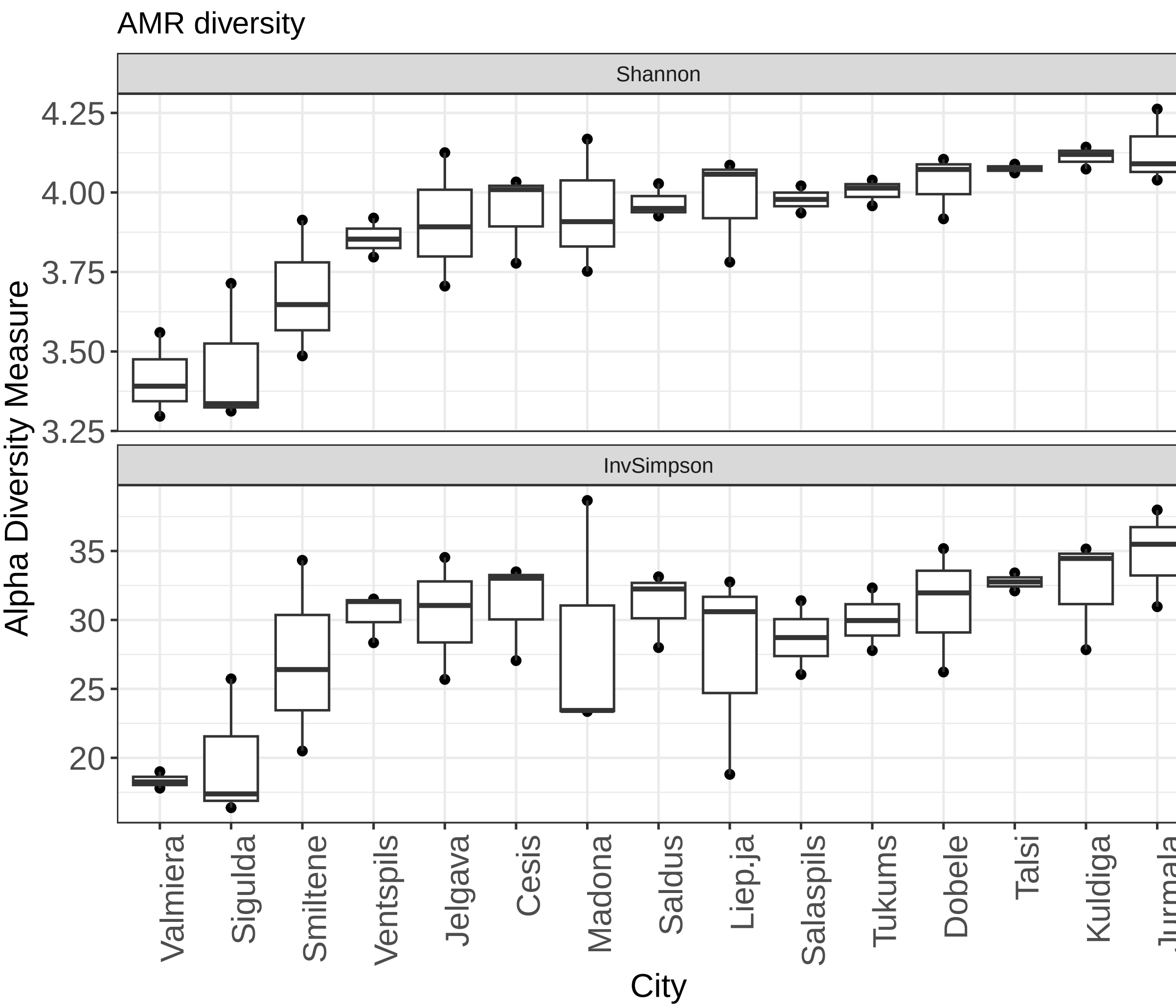
```
variance <- apply(abu_table_arg["MCR phosphoethanolamine transferase",], 1, var)
mean_percentage <- apply(abu_table_arg["MCR phosphoethanolamine transferase",], 1, mean) * 100
std_dev <- sqrt(apply(abu_table_arg["MCR phosphoethanolamine transferase",], 1, var)) * 100
```

City

Exploring differences how AMR gene composition is affected by Municipality factor.

ARG

```


alph_city <- plot_richness(ps_scaffolds_filtered, x="City", title="AMR diversity", measures=c("InvSimpso
    geom_boxplot() +
    theme_bw() +
    theme(legend.position = "none", axis.text.x = element_text(angle = 90, hjust = 1))+
  theme(
    legend.position = "right",
    axis.text.x = element_text(angle = 90, hjust = 1, size = 14),
    axis.text.y = element_text(size = 14),
    axis.title = element_text(size = 14),
    legend.text = element_text(size = 19),
    legend.title = element_text(size = 18)
  )

alph_city
```



Alpha diversity

Calculate distance

```
# Calculate diversity indexes
alpha_indexes_amr <- estimate_richness(ps_scaffolds_filtered, split = TRUE, c("Shannon", "Simpson", "In

kruskal.test(alpha_indexes_amr$Shannon ~ sample_data(ps_scaffolds_filtered)$City)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  alpha_indexes_amr$Shannon by sample_data(ps_scaffolds_filtered)$City
## Kruskal-Wallis chi-squared = 28.415, df = 14, p-value = 0.01253
```

```
kruskal.test(alpha_indexes_amr$InvSimpson ~ sample_data(ps_scaffolds_filtered)$City)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  alpha_indexes_amr$InvSimpson by sample_data(ps_scaffolds_filtered)$City
## Kruskal-Wallis chi-squared = 19.456, df = 14, p-value = 0.1483
```

Kruskal-Wallis rank sum test show significant differences in overall AMR diversity (Shannon) but dont show significant differences in how specific AMR dominate between cities (InvSimpson).

Lets show minimal and maximux diversity values.

```
min(alpha_indexes_amr$InvSimpson)
```

```
## [1] 16.38333
```

```
max(alpha_indexes_amr$InvSimpson)
```

```
## [1] 38.66515
```

```
min(alpha_indexes_amr$Shannon)
```

```
## [1] 3.296257
```

```
max(alpha_indexes_amr$Shannon)
```

```
## [1] 4.262206
```

Dunes Test

pinpoint which specific means are significantlt different from the others

```
library(dunn.test)
dunn_results <- dunn.test(alpha_indexes_amr$Shannon,
          sample_data(ps_scaffolds_filtered)$City,
          method="bh")
```

```
##   Kruskal-Wallis rank sum test
##
## data: x and group
## Kruskal-Wallis chi-squared = 28.4154, df = 14, p-value = 0.01
##
##
##                              Comparison of x by group
##                                (Benjamini-Hochberg)
## Col Mean-|
## Row Mean |      Cesis     Dobele    Jelgava    Jurmala    Kuldiga    Liepāja
## ---------+------------------------------------------------------------------
##   Dobele |  -0.845332
##          |     0.3425
##          |
##  Jelgava |   0.000000   0.845332
##          |     0.5000     0.3482
##          |
##  Jurmala |  -1.528101  -0.682768  -1.528101
##          |     0.1897     0.3711     0.1953
##          |
##  Kuldiga |  -1.658153  -0.812820  -1.658153  -0.130051
##          |     0.1892     0.3469     0.1964     0.4754
##          |
##  Liepāja |  -0.487692   0.357640  -0.487692   1.040409   1.170460
##          |     0.4212     0.4504     0.4267     0.2795     0.2760
##          |
##   Madona |  -0.162564   0.682768  -0.162564   1.365537   1.495589   0.325128
##          |     0.4665     0.3764     0.4713     0.2317     0.1965     0.4548
##          |
## Salaspil |  -0.218102   0.537986  -0.218102   1.148673   1.264994   0.218102
##          |     0.4671     0.4080     0.4721     0.2800     0.2514     0.4773
##          |
##   Saldus |  -0.195076   0.650256  -0.195076   1.333024   1.463076   0.292615
##          |     0.4672     0.3812     0.4721     0.2396     0.2035     0.4593
##          |
##  Sigulda |   1.593127   2.438460   1.593127   3.121229   3.251280   2.080819
##          |     0.1823     0.0774     0.1882    0.0236*    0.0302*     0.1229
##          |
## Smiltene |   1.105435   1.950768   1.105435   2.633537   2.763588   1.593127
##          |     0.2716     0.1490     0.2769     0.0634     0.0600     0.1945
##          |
##    Talsi |  -1.134133  -0.378044  -1.134133   0.232642   0.348964  -0.697928
##          |     0.2696     0.4462     0.2751     0.4760     0.4491     0.3746
##          |
##   Tukums |  -0.455179   0.390153  -0.455179   1.072922   1.202973   0.032512
##          |     0.4206     0.4459     0.4259     0.2806     0.2672     0.4965
##          |
## Valmiera |   1.690665   2.535998   1.690665   3.218767   3.348818   2.178357
##          |     0.1909     0.0654     0.1988    0.0225*    0.0426*     0.1285
##          |
## Ventspil |   0.552717   1.398050   0.552717   2.080819   2.210870   1.040409
##          |     0.4063     0.2240     0.4118     0.1311     0.1291     0.2846
## Col Mean-|
## Row Mean |     Madona   Salaspil     Saldus    Sigulda   Smiltene      Talsi
## ---------+------------------------------------------------------------------
## Salaspil |  -0.072700
##          |     0.4897
##          |
##   Saldus |  -0.032512   0.043620
##          |     0.4917     0.4968
##          |
##  Sigulda |   1.755691   1.643039   1.788204
##          |     0.1889     0.1882     0.1844
##          |
## Smiltene |   1.267999   1.206833   1.300512  -0.487692
##          |     0.2560     0.2714     0.2477     0.4159
##          |
##    Talsi |  -0.988731  -0.836217  -0.959651  -2.559069  -2.122864
##          |     0.2922     0.3413     0.3001     0.0689     0.1266
##          |
##   Tukums |  -0.292615  -0.189022  -0.260102  -2.048306  -1.560614   0.727008
##          |     0.4645     0.4649     0.4688     0.1252     0.1887     0.3774
##          |
## Valmiera |   1.853229   1.730280   1.885742   0.097538   0.585230   2.646310
##          |     0.1676     0.1908     0.1639     0.4842     0.4072     0.0712
##          |
## Ventspil |   0.715281   0.712468   0.747794  -1.040409  -0.552717   1.628498
##          |     0.3774     0.3731     0.3729     0.2899     0.4175     0.1872
## Col Mean-|
## Row Mean |     Tukums   Valmiera
## ---------+----------------------
## Valmiera |   2.145845
##          |     0.1288
##          |
## Ventspil |   1.007896  -1.137948
##          |     0.2888     0.2791
##
## alpha = 0.05
## Reject Ho if p <= alpha/2
```

```r
dunn_results$P.adjusted[dunn_results$P.adjusted < 0.05]
```

```
## [1] 0.02363783 0.03015770 0.02252998 0.04260734
```

```r
dunn_results$chi2
```

```
## [1] 28.41543
```

```r
dunn_results$Z[dunn_results$P.adjusted < 0.05]
```

```
## [1] 3.121229 3.251280 3.218768 3.348819
```

Shows most significantly different pairs between cities.

```r
dunn_results$comparisons[dunn_results$P.adjusted < 0.05]
```

```
## [1] "Jurmala - Sigulda"  "Kuldiga - Sigulda"  "Jurmala - Valmiera"
## [4] "Kuldiga - Valmiera"
```

```r
dunn_results <- dunn.test(alpha_indexes_amr$InvSimpson,
          sample_data(ps_scaffolds_filtered)$City,
          method="bh")
```

```
##   Kruskal-Wallis rank sum test
##
## data: x and group
## Kruskal-Wallis chi-squared = 19.4556, df = 14, p-value = 0.15
##
##
##                            Comparison of x by group
##                              (Benjamini-Hochberg)
## Col Mean-|
## Row Mean |      Cesis     Dobele    Jelgava    Jurmala    Kuldiga    Liepāja
## ---------+------------------------------------------------------------------
##   Dobele |   0.065025
##          |     0.5028
##          |
##  Jelgava |   0.357640   0.292615
##          |     0.4614     0.4491
##          |
##  Jurmala |  -0.747794  -0.812820  -1.105435
##          |     0.4420     0.4751     0.4035
##          |
##  Kuldiga |  -0.357640  -0.422666  -0.715281   0.390153
##          |     0.4671     0.4973     0.4448     0.4875
##          |
##  Liepāja |   0.812820   0.747794   0.455179   1.560614   1.170460
##          |     0.4857     0.4503     0.5085     0.3114     0.3734
##          |
##   Madona |   0.715281   0.650256   0.357640   1.463076   1.072922  -0.097538
##          |     0.4529     0.4437     0.4729     0.3274     0.4020     0.5044
##          |
## Salaspil |   0.770628   0.712468   0.450745   1.439476   1.090512   0.043620
##          |     0.4724     0.4310     0.5035     0.3282     0.4018     0.5017
##          |
##   Saldus |   0.065025   0.000000  -0.292615   0.812820   0.422666  -0.747794
##          |     0.5079     0.5000     0.4541     0.4967     0.5044     0.4590
##          |
##  Sigulda |   2.210870   2.145845   1.853229   2.958665   2.568511   1.398050
##          |     0.1775     0.1395     0.2394     0.0811     0.1341     0.3039
##          |
## Smiltene |   0.812820   0.747794   0.455179   1.560614   1.170460   0.000000
##          |     0.5083     0.4680     0.5162     0.3278     0.3847     0.5048
##          |
##    Talsi |  -0.319883  -0.378044  -0.639767   0.348964   0.000000  -1.046892
##          |     0.4682     0.4810     0.4423     0.4599     0.5097     0.3973
##          |
##   Tukums |   0.520204   0.455179   0.162564   1.267999   0.877845  -0.292615
##          |     0.5024     0.5242     0.4970     0.3708     0.4750     0.4593
##          |
## Valmiera |   2.243383   2.178357   1.885742   2.991178   2.601024   1.430563
##          |     0.1865     0.1542     0.2396     0.1459     0.1627     0.3080
##          |
## Ventspil |   0.422666   0.357640   0.065025   1.170460   0.780307  -0.390153
##          |     0.5117     0.4789     0.5132     0.3967     0.4760     0.4941
## Col Mean-|
## Row Mean |     Madona   Salaspil     Saldus    Sigulda   Smiltene      Talsi
## ---------+------------------------------------------------------------------
## Salaspil |   0.130861
##          |     0.5057
##          |
##   Saldus |  -0.650256  -0.712468
##          |     0.4511     0.4386
##          |
##  Sigulda |   1.495589   1.206833   2.145845
##          |     0.3216     0.3853     0.1522
##          |
## Smiltene |   0.097538  -0.043620   0.747794  -1.398050
##          |     0.5097     0.5067     0.4773     0.3152
##          |
##    Talsi |  -0.959651  -0.995497  -0.378044  -2.297346  -1.046892
##          |     0.4318     0.4193     0.4873     0.1890     0.4078
##          |
##   Tukums |  -0.195076  -0.305343   0.455179  -1.690665  -0.292615   0.785169
##          |     0.4877     0.4695     0.5324     0.2651     0.4645     0.4829
##          |
## Valmiera |   1.528101   1.235914   2.178357   0.032512   1.430563   2.326426
##          |     0.3162     0.3789     0.1714     0.5014     0.3204     0.2100
##          |
## Ventspil |  -0.292615  -0.392584   0.357640  -1.788204  -0.390153   0.697928
##          |     0.4699     0.5065     0.4850     0.2420     0.5009     0.4318
## Col Mean-|
## Row Mean |     Tukums   Valmiera
## ---------+----------------------
## Valmiera |   1.723178
##          |     0.2621
##          |
## Ventspil |  -0.097538  -1.820717
##          |     0.5151     0.2403
##
## alpha = 0.05
## Reject Ho if p <= alpha/2
```

```
dunn_results$P.adjusted[dunn_results$P.adjusted < 0.05]
```

```
## numeric(0)
```

```
dunn_results$chi2
```

```
## [1] 19.4556
```

```
dunn_results$Z[dunn_results$P.adjusted < 0.05]
```

```
## numeric(0)
```

```
dunn_results$comparisons[dunn_results$P.adjusted < 0.05]
```

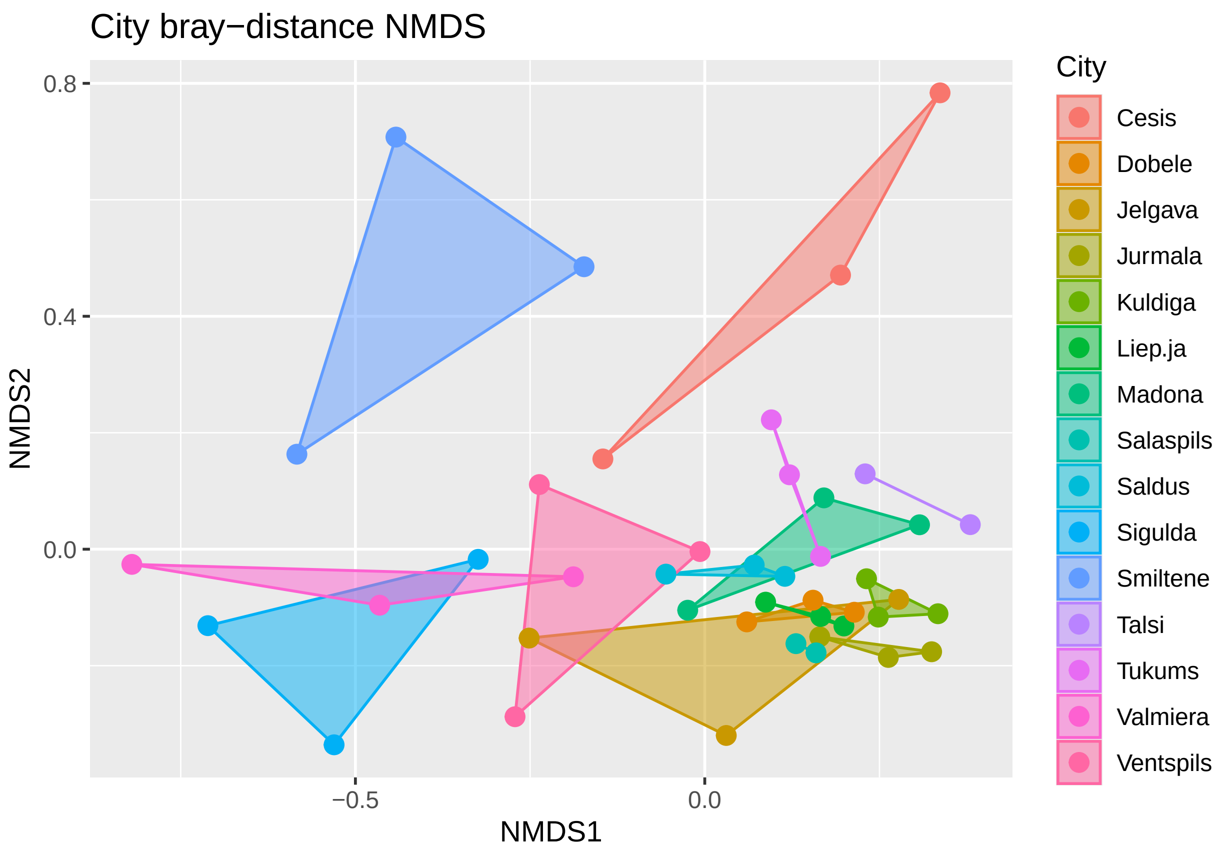
```
## character(0)
```

Beta diversity Lets compare how simmilar municipalities are between each other using bray-curtis distance on a NMDS plot.

```
GP.ord2 <- ordinate(ps_amr_CSS, "NMDS", "bray")
```

```
## Square root transformation
## Wisconsin double standardization
## Run 0 stress 0.137757
## Run 1 stress 0.2005092
## Run 2 stress 0.1722958
## Run 3 stress 0.174337
## Run 4 stress 0.153554
## Run 5 stress 0.1760838
## Run 6 stress 0.1718425
## Run 7 stress 0.138203
## ... Procrustes: rmse 0.02336348  max resid 0.1426252
## Run 8 stress 0.1720912
## Run 9 stress 0.1434781
## Run 10 stress 0.175746
## Run 11 stress 0.1763512
## Run 12 stress 0.1381716
## ... Procrustes: rmse 0.02475104  max resid 0.1419284
## Run 13 stress 0.1700024
## Run 14 stress 0.1382146
## ... Procrustes: rmse 0.02482664  max resid 0.141987
## Run 15 stress 0.1751137
## Run 16 stress 0.137757
## ... Procrustes: rmse 1.875946e-05  max resid 7.134341e-05
## ... Similar to previous best
## Run 17 stress 0.138203
## ... Procrustes: rmse 0.02336629  max resid 0.1426453
## Run 18 stress 0.1381716
## ... Procrustes: rmse 0.02475398  max resid 0.141942
## Run 19 stress 0.143372
## Run 20 stress 0.1767373
## *** Best solution repeated 1 times
```

```
p1 = plot_ordination(ps_amr_CSS, GP.ord2, type="samples", color = "City", title="City bray-distance NMDS
 geom_polygon(aes(fill=City), alpha = 1/2) + geom_point(size=3)

print(p1)
```



Calculate distances

```
tax_bray_amr <- phyloseq::distance(ps_amr_CSS, method = "bray")
```

Lets check variance with ANOVA.

```
anosim(tax_bray_amr, sample_data(ps_amr_CSS)$City, distance = "bray", permutations = 999)
```

```
##
## Call:
## anosim(x = tax_bray_amr, grouping = sample_data(ps_amr_CSS)$City,      permutations = 999, distance =
## Dissimilarity: bray
##
## ANOSIM statistic R: 0.5901
##       Significance: 0.001
##
## Permutation: free
## Number of permutations: 999
```

PERMANOVA

```
adonis2_rez<-adonis2(tax_bray_amr ~ sample_data(ps_amr_CSS)$City)
print(adonis2_rez)
```

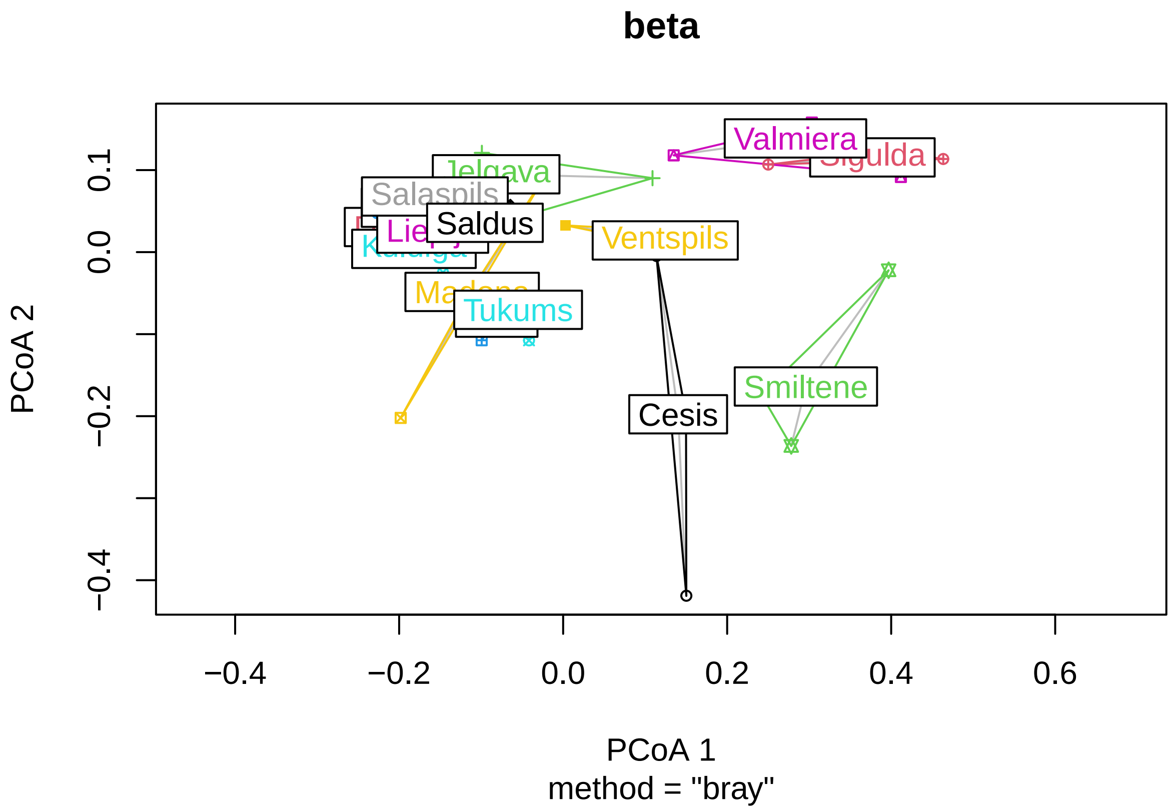
```
## Permutation test for adonis under reduced model
## Permutation: free
## Number of permutations: 999
##
## adonis2(formula = tax_bray_amr ~ sample_data(ps_amr_CSS)$City)
##          Df SumOfSqs      R2      F Pr(>F)
## Model    14   3.0982 0.65871 3.8601  0.001 ***
## Residual 28   1.6052 0.34129
## Total    42   4.7034 1.00000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Chcek beta dispersion between City

```
beta <- betadisper(tax_bray_amr, sample_data(ps_amr_CSS)$City)
print(anova(beta))
```

```
## Analysis of Variance Table
##
## Response: Distances
##           Df   Sum Sq   Mean Sq F value Pr(>F)
## Groups    14 0.078542 0.0056101   0.941 0.5309
## Residuals 28 0.166939 0.0059621
```

```
plot(beta)
```



### Population Size

Now look wether population size impact AMR.

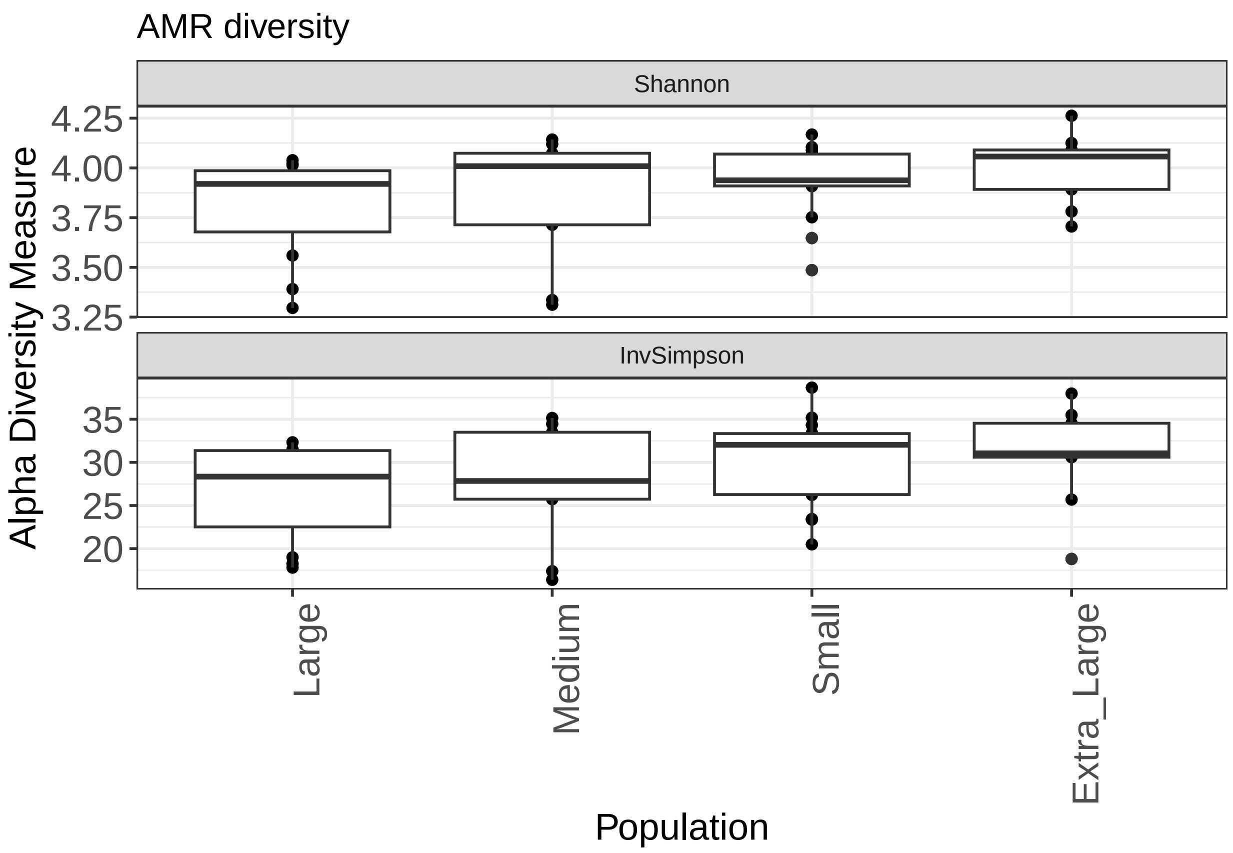
```
kruskal.test(alpha_indexes_amr$Shannon ~ sample_data(ps_scaffolds_filtered)$Population)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  alpha_indexes_amr$Shannon by sample_data(ps_scaffolds_filtered)$Population
## Kruskal-Wallis chi-squared = 4.2615, df = 3, p-value = 0.2346
```

```
kruskal.test(alpha_indexes_amr$InvSimpson ~ sample_data(ps_scaffolds_filtered)$Population)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  alpha_indexes_amr$InvSimpson by sample_data(ps_scaffolds_filtered)$Population
## Kruskal-Wallis chi-squared = 3.2291, df = 3, p-value = 0.3576
```

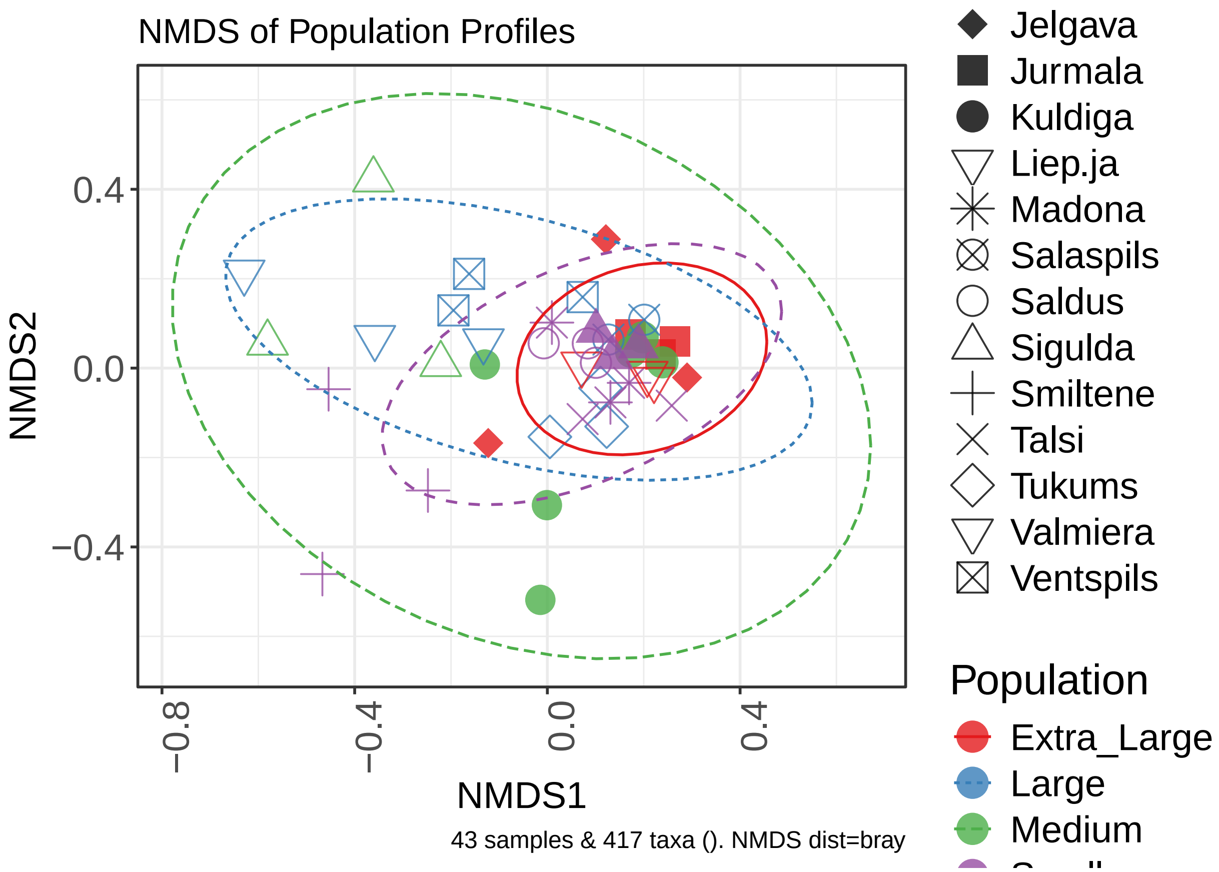
```
plot_richness(ps_scaffolds_filtered, x="Population", title="AMR diversity", measures=c("InvSimpson", "Sl
    geom_boxplot() +
    theme_bw() +
    theme(legend.position = "none", axis.text.x = element_text(angle = 90, hjust = 1))+
  theme(
    legend.position = "right",
    axis.text.x = element_text(angle = 90, hjust = 1, size = 14),
    axis.text.y = element_text(size = 14),
    axis.title = element_text(size = 14),
    legend.text = element_text(size = 14),
    legend.title = element_text(size = 14)
  )
```



```r
beta_popul <- ps_scaffolds_filtered %>%
  dist_calc(dist = "bray", binary = TRUE) %>%
  ord_calc("NMDS") %>%
  ord_plot(
    color = "Population",
    shape = "City",
    size = 5,
    alpha = 0.8
  ) +
  scale_shape_manual(values = c(16, 17, 18, 15, 19, 25, 8, 13, 1, 2, 3, 4, 5, 6, 7)) + # Manual shapes
  stat_ellipse(aes(linetype = Population, colour = Population)) +
  theme_bw() +
  scale_color_brewer(palette = "Set1") +
  scale_fill_brewer(palette = "Set1") +
  labs(
    title = "NMDS of Population Profiles",
    x = "NMDS1",
    y = "NMDS2"
  )+
  theme(
    legend.position = "right",
    axis.text.x = element_text(angle = 90, hjust = 1, size = 14),
    axis.text.y = element_text(size = 14),
    axis.title = element_text(size = 14),
    legend.text = element_text(size = 14),
    legend.title = element_text(size = 16)
  )
```

```
## Run 0 stress 0.1476541
## Run 1 stress 0.1608673
## Run 2 stress 0.1535323
## Run 3 stress 0.1791487
## Run 4 stress 0.183082
## Run 5 stress 0.1892775
## Run 6 stress 0.163148
## Run 7 stress 0.1719986
## Run 8 stress 0.1844325
## Run 9 stress 0.1584246
## Run 10 stress 0.1719985
## Run 11 stress 0.1835751
## Run 12 stress 0.1615521
## Run 13 stress 0.1786126
## Run 14 stress 0.1477985
## ... Procrustes: rmse 0.005259716  max resid 0.02982373
## Run 15 stress 0.1535323
## Run 16 stress 0.178458
## Run 17 stress 0.1847899
## Run 18 stress 0.1747663
## Run 19 stress 0.1618977
## Run 20 stress 0.1765684
## *** Best solution was not repeated -- monoMDS stopping criteria:
##     20: stress ratio > sratmax
```

```
beta_popul
```



```
adonis2_rez<-adonis2(tax_bray_amr ~ sample_data(ps_amr_CSS)$Population)
print(adonis2_rez)
```

```
## Permutation test for adonis under reduced model
## Permutation: free
## Number of permutations: 999
##
## adonis2(formula = tax_bray_amr ~ sample_data(ps_amr_CSS)$Population)
##          Df SumOfSqs      R2      F Pr(>F)
## Model     3   0.6559 0.13944 2.1065  0.009 **
## Residual 39   4.0476 0.86056
## Total    42   4.7034 1.00000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
beta <- betadisper(tax_bray_amr, sample_data(ps_amr_CSS)$Population)
print("BETADISPER")
```

```
## [1] "BETADISPER"
```

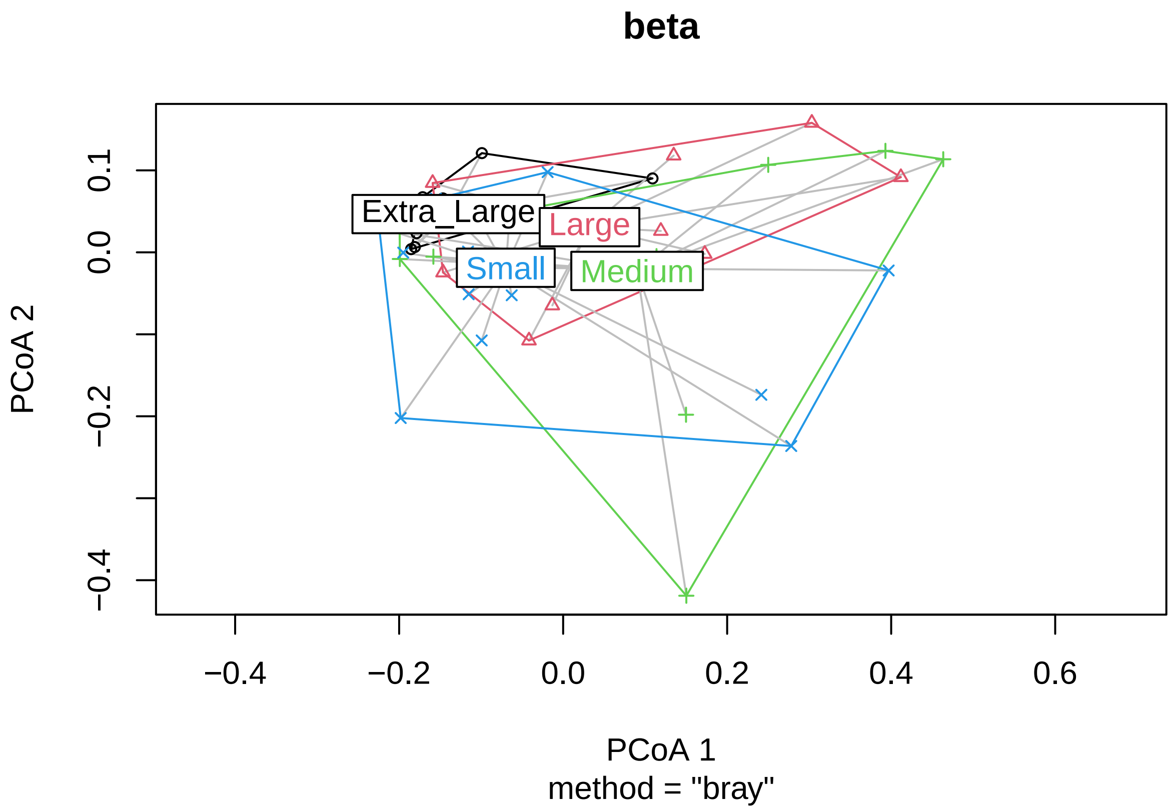
```
print("Population")
```

```
## [1] "Population"
```

```
print(anova(beta))
```

```
## Analysis of Variance Table
##
## Response: Distances
##           Df   Sum Sq   Mean Sq F value  Pr(>F)
## Groups     3 0.090647 0.0302156  3.9586 0.01479 *
## Residuals 39 0.297680 0.0076328
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
plot(beta)
```



Hospital

No significance in Regional Hospital.

```
wilcox.test(alpha_indexes_amr$Shannon~sample_data(ps_scaffolds_filtered)$Regional_Hospital,
                     p.adjust.method = "BH")
```

```
##
##  Wilcoxon rank sum exact test
##
```

```
## data:  alpha_indexes_amr$Shannon by sample_data(ps_scaffolds_filtered)$Regional_Hospital
## W = 207, p-value = 0.6196
## alternative hypothesis: true location shift is not equal to 0
```

```
wilcox.test(alpha_indexes_amr$InvSimpson~sample_data(ps_scaffolds_filtered)$Regional_Hospital,
                     p.adjust.method = "BH")
```

```
##
##  Wilcoxon rank sum exact test
##
## data:  alpha_indexes_amr$InvSimpson by sample_data(ps_scaffolds_filtered)$Regional_Hospital
## W = 246, p-value = 0.6718
## alternative hypothesis: true location shift is not equal to 0
```

Hospital Types

```
kruskal.test(alpha_indexes_amr$Shannon ~ sample_data(ps_scaffolds_filtered)$Hospital_Type)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  alpha_indexes_amr$Shannon by sample_data(ps_scaffolds_filtered)$Hospital_Type
## Kruskal-Wallis chi-squared = 18.538, df = 7, p-value = 0.009763
```

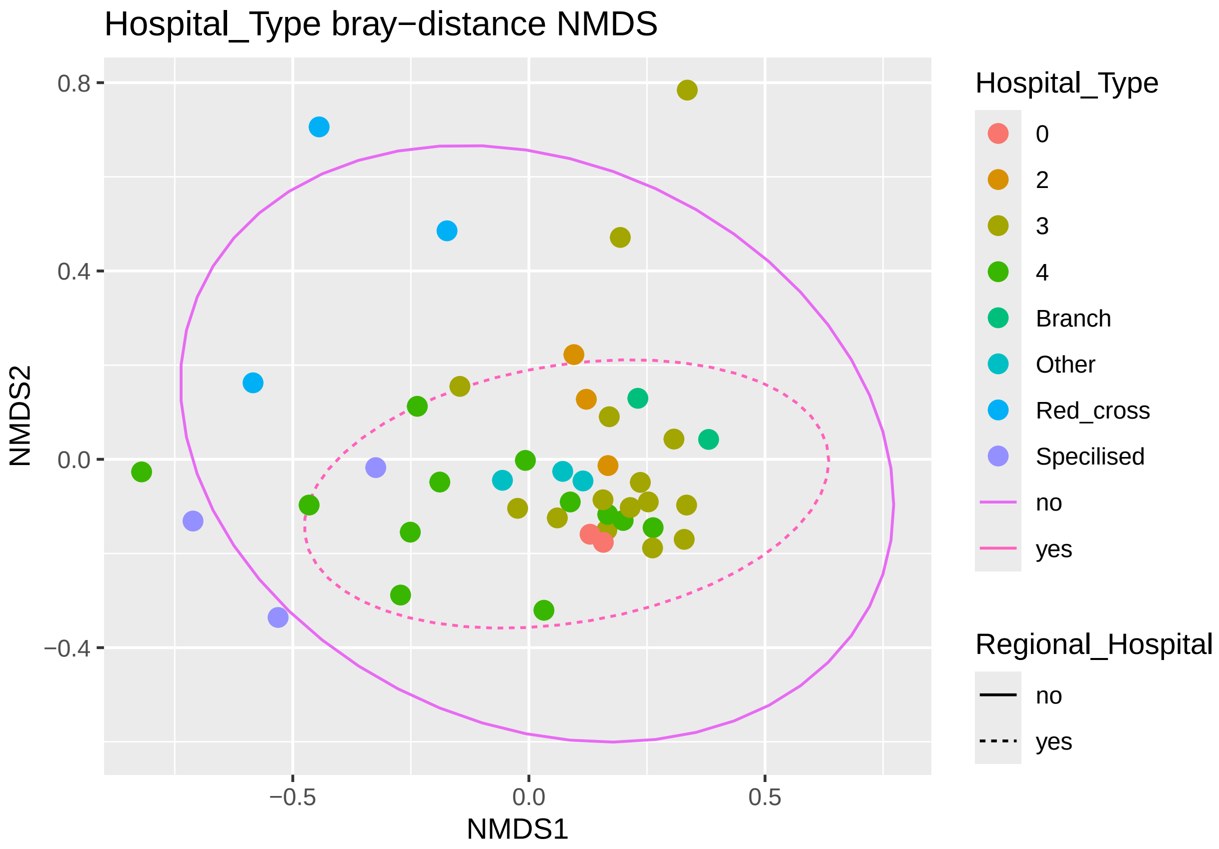
```
kruskal.test(alpha_indexes_amr$InvSimpson ~ sample_data(ps_scaffolds_filtered)$Hospital_Type)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  alpha_indexes_amr$InvSimpson by sample_data(ps_scaffolds_filtered)$Hospital_Type
## Kruskal-Wallis chi-squared = 12.451, df = 7, p-value = 0.08666
```

```
GP.ord2 <- ordinate(ps_amr_CSS, "NMDS", "bray")
```

```
## Square root transformation
## Wisconsin double standardization
## Run 0 stress 0.137757
## Run 1 stress 0.1381956
## ... Procrustes: rmse 0.02197242  max resid 0.1298134
## Run 2 stress 0.1377036
## ... New best solution
## ... Procrustes: rmse 0.004229607  max resid 0.02448258
## Run 3 stress 0.1475182
## Run 4 stress 0.1554664
## Run 5 stress 0.1784969
## Run 6 stress 0.1783075
## Run 7 stress 0.1812036
## Run 8 stress 0.179745
## Run 9 stress 0.1685441
## Run 10 stress 0.1784377
## Run 11 stress 0.1382029
## ... Procrustes: rmse 0.02403351  max resid 0.1433942
## Run 12 stress 0.1791362
## Run 13 stress 0.1381719
## ... Procrustes: rmse 0.02530655  max resid 0.1425349
## Run 14 stress 0.1760179
## Run 15 stress 0.1803947
## Run 16 stress 0.1377036
## ... Procrustes: rmse 1.172938e-05  max resid 6.288059e-05
## ... Similar to previous best
## Run 17 stress 0.143372
## Run 18 stress 0.1755806
## Run 19 stress 0.1382029
## ... Procrustes: rmse 0.02403364  max resid 0.1434358
## Run 20 stress 0.1434781
## *** Best solution repeated 1 times
```

```
p1 = plot_ordination(ps_amr_CSS, GP.ord2, type="samples", color = "Hospital_Type", title="Hospital_Type
 stat_ellipse(aes(linetype = Regional_Hospital, colour = Regional_Hospital)) + geom_point(size=3)

print(p1)
```



```
adonis2_rez<-adonis2(tax_bray_amr ~ sample_data(ps_amr_CSS)$Hospital_Type)
print(adonis2_rez)
```

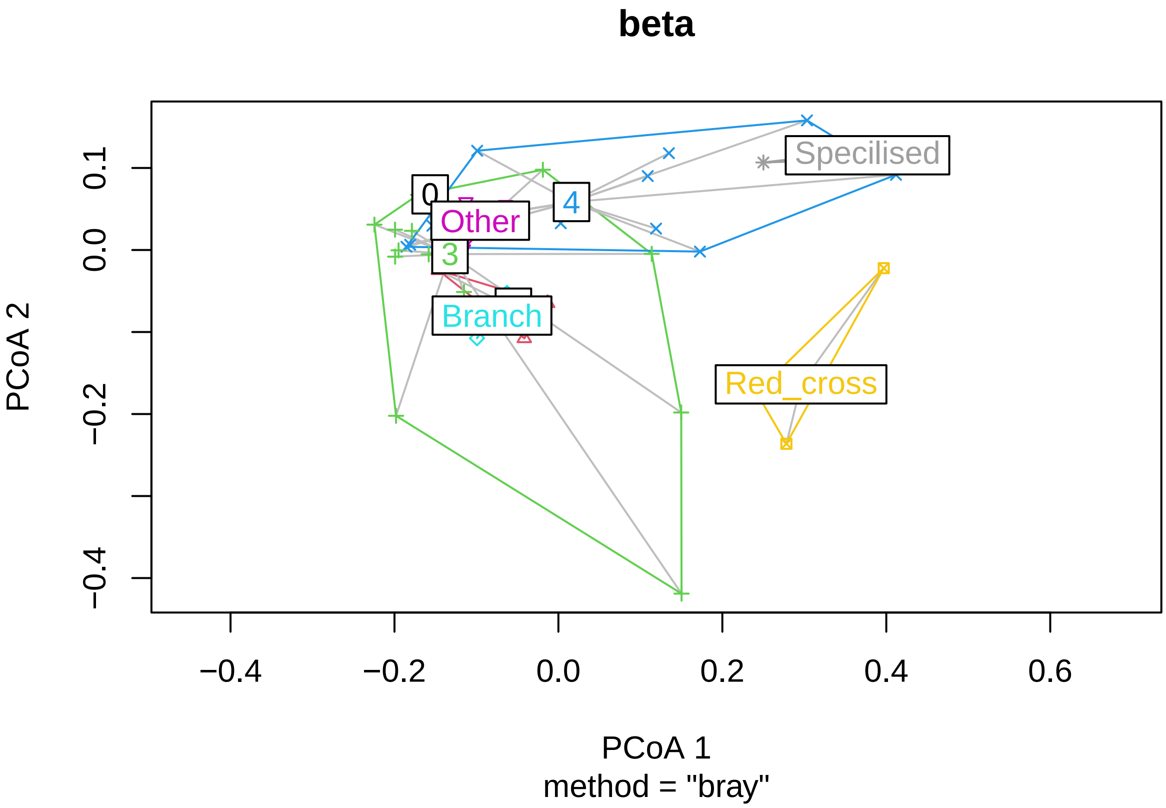
```
## Permutation test for adonis under reduced model
## Permutation: free
## Number of permutations: 999
##
## adonis2(formula = tax_bray_amr ~ sample_data(ps_amr_CSS)$Hospital_Type)
##          Df SumOfSqs      R2      F Pr(>F)
## Model     7   1.7110 0.36377 2.8588  0.001 ***
## Residual 35   2.9925 0.63623
## Total    42   4.7034 1.00000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Check BETADISPER between Hospital_Types

```
beta <- betadisper(tax_bray_amr, sample_data(ps_amr_CSS)$Hospital_Type)
print(anova(beta))
```

```
## Analysis of Variance Table
##
## Response: Distances
##           Df  Sum Sq   Mean Sq F value   Pr(>F)
## Groups     7 0.18447 0.0263526  3.5618 0.005392 **
## Residuals 35 0.25895 0.0073986
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
plot(beta)
```



```
dunn_results <- dunn.test(alpha_indexes_amr$Shannon,
          sample_data(ps_scaffolds_filtered)$Hospital_Type,
          method="bh")
```

```
##   Kruskal-Wallis rank sum test
##
## data: x and group
## Kruskal-Wallis chi-squared = 18.5385, df = 7, p-value = 0.01
##
##
##                             Comparison of x by group
##                               (Benjamini-Hochberg)
## Col Mean-|
## Row Mean |          0          2          3          4     Branch      Other
## ---------+------------------------------------------------------------------
##        2 |  -0.189022
##          |     0.4408
##          |
##        3 |  -0.645351  -0.495291
##          |     0.3301     0.3776
##          |
##        4 |   0.729910   1.130959   2.693740
##          |     0.3258     0.2581     0.0495
##          |
##   Branch |  -0.836217  -0.727008  -0.465499  -1.824776
##          |     0.3319     0.3115     0.3593     0.1361
##          |
##    Other |   0.043620   0.260102   0.831082  -0.801953   0.959651
##          |     0.4826     0.4280     0.3157     0.3114     0.3147
##          |
## Red_cros |   1.206833   1.560614   2.510036   0.843079   2.122864   1.300512
##          |     0.2450     0.1661     0.0423     0.3493     0.0945     0.2257
##          |
## Specilis |   1.643039   2.048306   3.139643   1.459966   2.559069   1.788204
##          |     0.1561     0.0946    0.0237*     0.1837     0.0490     0.1291
## Col Mean-|
## Row Mean |   Red_cros
## ---------+-----------
## Specilis |   0.487692
##          |     0.3650
##
## alpha = 0.05
## Reject Ho if p <= alpha/2
```

```
dunn_results$chi2
```

```
## [1] 18.53848
```

```
dunn_results$P.adjusted[dunn_results$P.adjusted < 0.05]
```

```
## [1] 0.04945864 0.04225157 0.02368146 0.04897794
```

```r
dunn_results$Z[dunn_results$P.adjusted < 0.05]
```

```
## [1] 2.693740 2.510036 3.139644 2.559070
```

```r
dunn_results$comparisons[dunn_results$P.adjusted < 0.05]
```

```
## [1] "3 - 4"               "3 - Red_cross"       "3 - Specilised"
## [4] "Branch - Specilised"
```

```r
dunn_results <- dunn.test(alpha_indexes_amr$InvSimpson,
          sample_data(ps_scaffolds_filtered)$Hospital_Type,
          method="bh")
```

```
##   Kruskal-Wallis rank sum test
##
## data: x and group
## Kruskal-Wallis chi-squared = 12.4514, df = 7, p-value = 0.09
##
##
##                            Comparison of x by group
##                              (Benjamini-Hochberg)
## Col Mean-|
## Row Mean |          0          2          3          4     Branch      Other
## ---------+------------------------------------------------------------------
##        2 |  -0.305343
##          |     0.4627
##          |
##        3 |  -1.005055  -0.755529
##          |     0.3391     0.3937
##          |
##        4 |   0.104272   0.555198   2.159104
##          |     0.4755     0.4265     0.1439
##          |
##   Branch |  -0.995497  -0.785169  -0.317385  -1.407684
##          |     0.3195     0.4035     0.4779     0.3184
##          |
##    Other |  -0.712468  -0.455179   0.167895  -1.130959   0.378044
##          |     0.3704     0.4543     0.4667     0.3285     0.4703
##          |
## Red_cros |  -0.043620   0.292615   1.133293  -0.185066   1.046892   0.747794
##          |     0.4826     0.4491     0.3599     0.4778     0.3443     0.3744
##          |
## Specilis |   1.206833   1.690665   2.938169   1.583343   2.297346   2.145845
##          |     0.3539     0.2545     0.0462     0.2645     0.1512     0.1116
## Col Mean-|
## Row Mean |   Red_cros
## ---------+-----------
## Specilis |   1.398050
##          |     0.2837
##
## alpha = 0.05
## Reject Ho if p <= alpha/2
```

```
dunn_results$P.adjusted[dunn_results$P.adjusted < 0.05]
```

```
## [1] 0.0462219
```

```
dunn_results$chi2
```

```
## [1] 12.45137
```

```
dunn_results$Z[dunn_results$P.adjusted < 0.05]
```

```
## [1] 2.93817
```

```
dunn_results$comparisons[dunn_results$P.adjusted < 0.05]
```

```
## [1] "3 - Specilised"
```

SIAMCAT Using siamcat we look for genes that are explaining most of the variations by factor groups.

```
library(SIAMCAT)
```

```
## Loading required package: mlr3
```

```
## Registered S3 methods overwritten by 'pROC':
##   method    from
##   print.roc huge
##   plot.roc  huge
```

```
relab_ps = transform_sample_counts(ps_scaffolds_filtered, function(x) x/sum(x))

psglom = tax_glom(relab_ps, "ARG")
```

```
create_siamcat_plot <- function(psg, label_column, case_value, output_prefix) {
  # Create label
  sc_label <- create.label(meta=sample_data(psg), label=label_column, case=case_value)

  # Create SIAMCAT object
  siamcat_o <- siamcat(phyloseq=psg, label=sc_label)


  # Filter features
  siamcat_o <- filter.features(siamcat_o, filter.method='abundance', cutoff=1e-03)
  siamcat_o <- filter.features(siamcat_o, filter.method='prevalence', cutoff=0.05, feature.type='filtere

  # Check associations
  siamcat_o <- check.associations(siamcat_o)

  association.plot(siamcat_o, panels=c("fc", "prevalence"), prompt = FALSE, verbose = 0)
  # Final association plot
```

```r
  svg(paste0("../plots/",output_prefix, "_final_association.svg"), width=12, height=7)
  association.plot(siamcat_o, panels=c("fc", "prevalence"), prompt = FALSE, verbose = 0)
  dev.off()
  cat("plot saved to", paste0(output_prefix, "_final_association.svg"), "\n")

  return(siamcat_o)
}
```

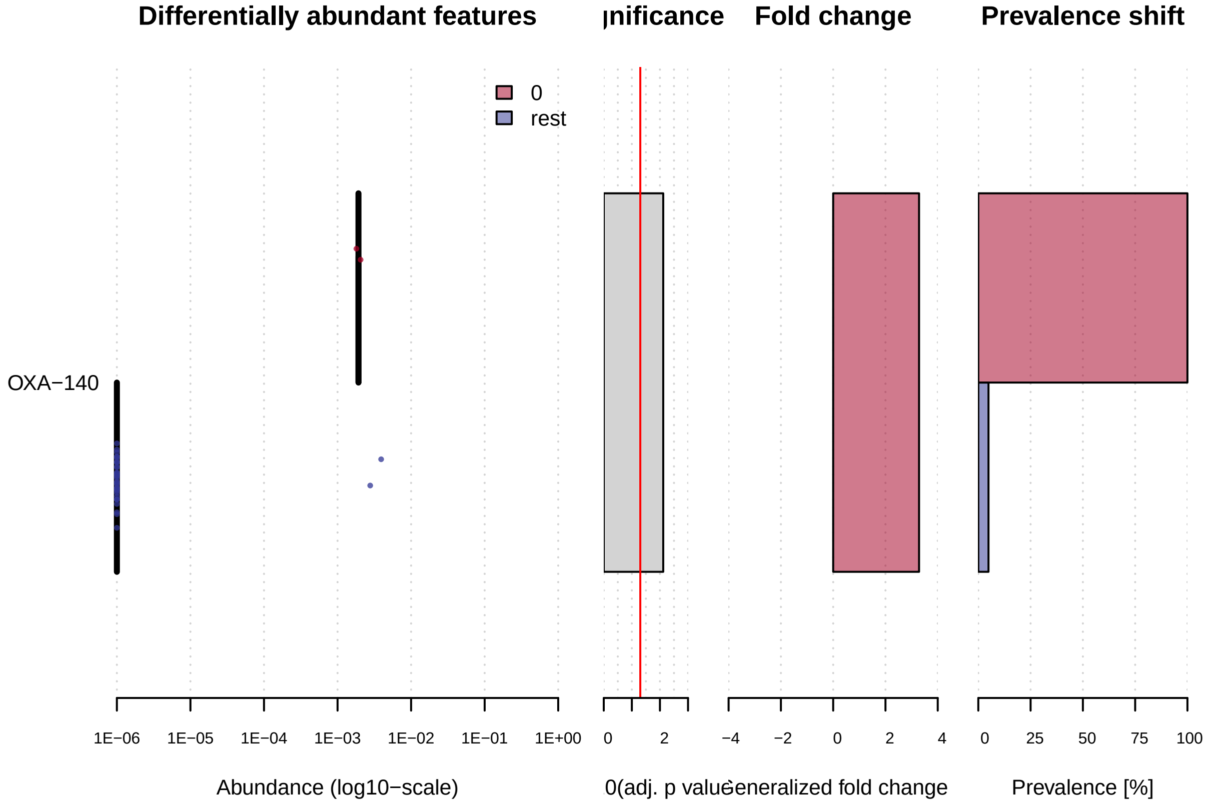
Here most differentially found genes are plotted

```r
label_case_pairs <- list(
  list(label = "Hospital_Type", case = "0"),
  list(label = "Hospital_Type", case = "2"),
  list(label = "Hospital_Type", case = "3"),
  list(label = "Hospital_Type", case = "4"),
  list(label = "Hospital_Type", case = "Specilised"),
  list(label = "Hospital_Type", case = "Other"),
  list(label = "Hospital_Type", case = "Red_cross"),
  list(label = "Hospital_Type", case = "Branch")
  # Add more pairs as needed
)

#psglom = tax_glom(relab_ps, "ARG")

# Loop through the pairs
for (pair in label_case_pairs) {
  label_column <- pair$label
  case_value <- pair$case

  # Create a unique output prefix for each pair
  output_prefix <- paste0("siamcat_", label_column, "_", case_value)

  # Run the function
  siamcat_result <- create_siamcat_plot(psglom, label_column, case_value, output_prefix)

  # You can do additional processing with siamcat_result here if needed

  volcano.plot(siamcat_result)

  print(rownames(associations(siamcat_result))[associations(siamcat_result)$p.adj<0.05])
  # Print a message to indicate progress
  cat("Completed processing for", label_column, "with case", case_value, "\n")
}
```

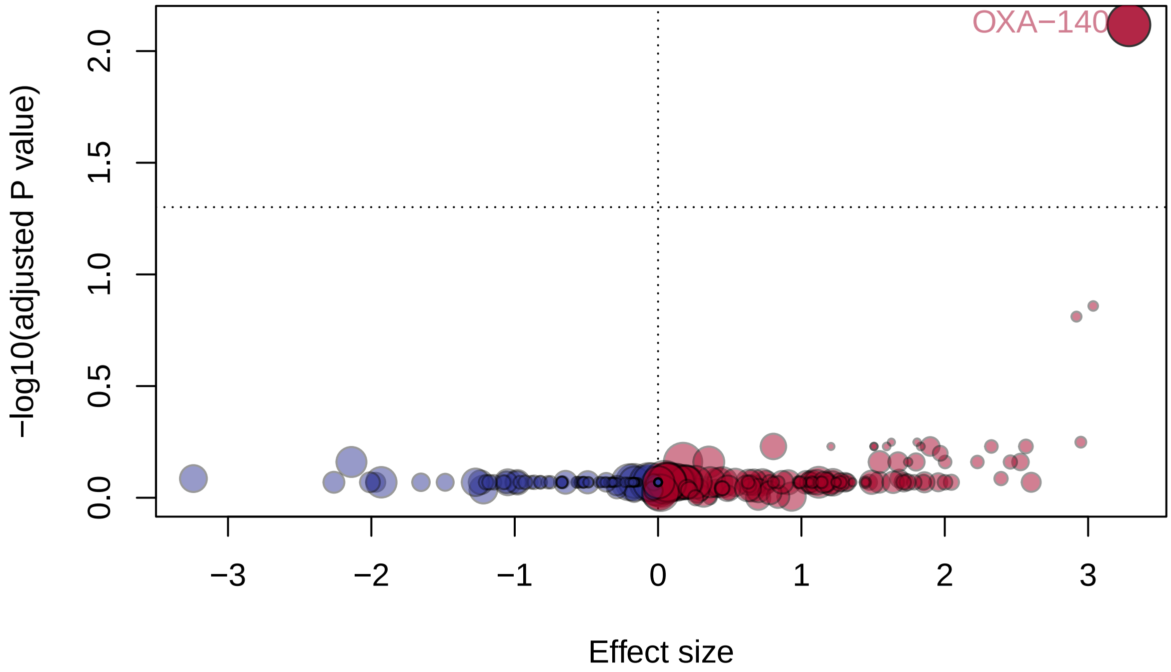
```
## Label used as case:
##    0
## Label used as control:
##    rest
```

```
## + finished create.label.from.metadata in 0.007 s
```

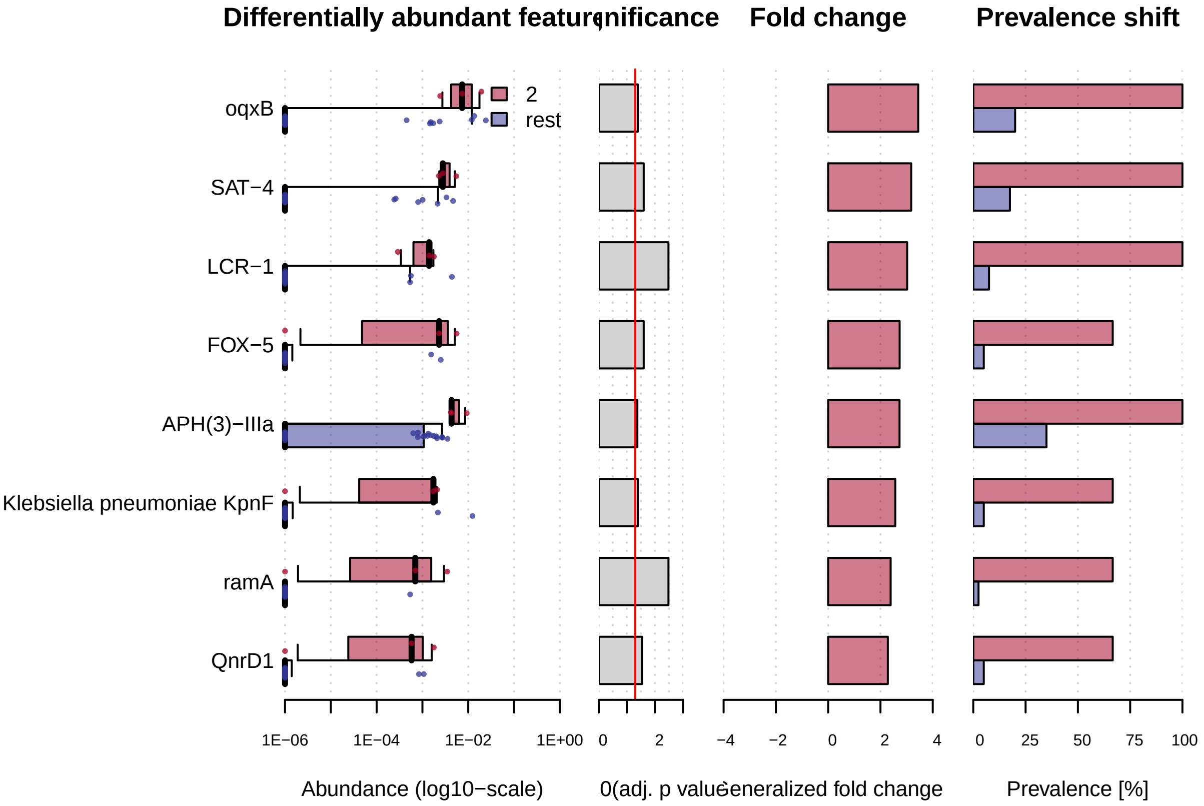
```
## + starting validate.data
## +++ checking overlap between labels and features

## + Keeping labels of 43 sample(s).

## +++ checking sample number per class

## Data set has a limited number of training examples:
##  rest    41
##  0   2
## Note that a dataset this small/skewed is not necessarily suitable for analysis in this pipeline.

## +++ checking overlap between samples and metadata

## + finished validate.data in 0.036 s

## Features successfully filtered
## Features successfully filtered

## Less than 5 associations found. Consider changing your alpha value.
## Less than 5 associations found. Consider changing your alpha value.
```



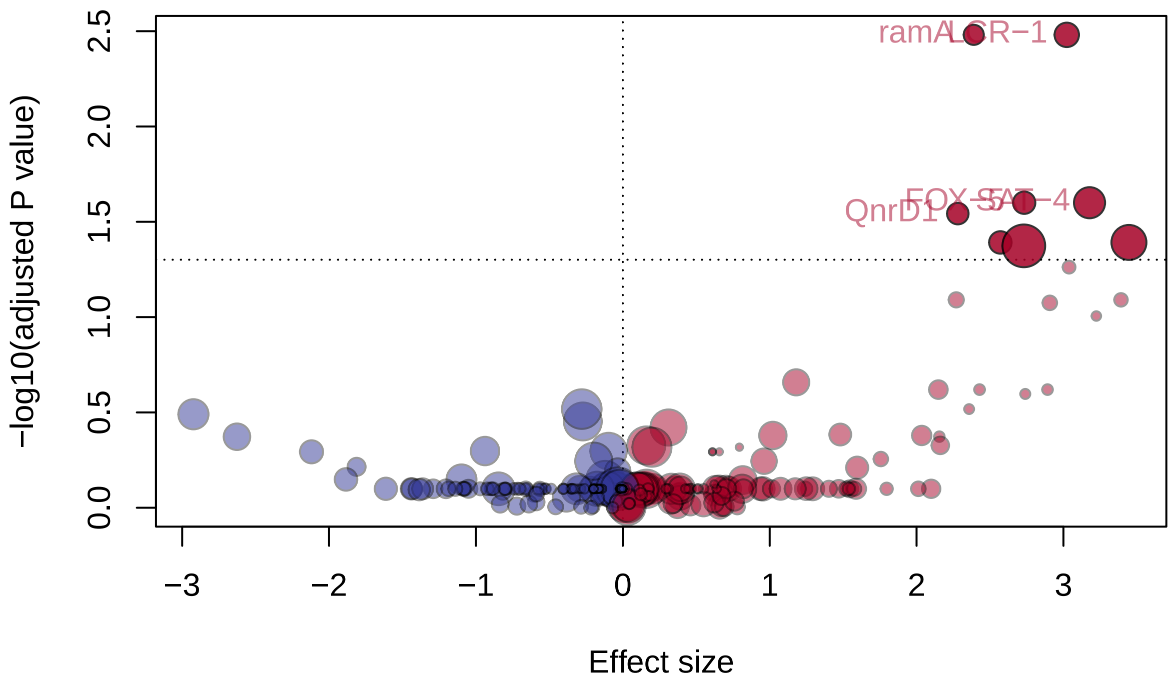
```
## plot saved to siamcat_Hospital_Type_0_final_association.svg

## Fewer significant features at alpha 0.05 than desired features for annotation (5)!
```



```
## [1] "OXA-140"
## Completed processing for Hospital_Type with case 0

## Label used as case:
##    2
## Label used as control:
##    rest

## + finished create.label.from.metadata in 0.004 s

## + starting validate.data

## +++ checking overlap between labels and features

## + Keeping labels of 43 sample(s).

## +++ checking sample number per class

## Data set has a limited number of training examples:
##  rest    40
##  2   3
## Note that a dataset this small/skewed is not necessarily suitable for analysis in this pipeline.

## +++ checking overlap between samples and metadata
## + finished validate.data in 0.054 s

## Features successfully filtered
## Features successfully filtered

## Warning in ci.auc.roc(roc, ...): ci.auc() of a ROC curve with AUC == 1 is
## always 1-1 and can be misleading.
```

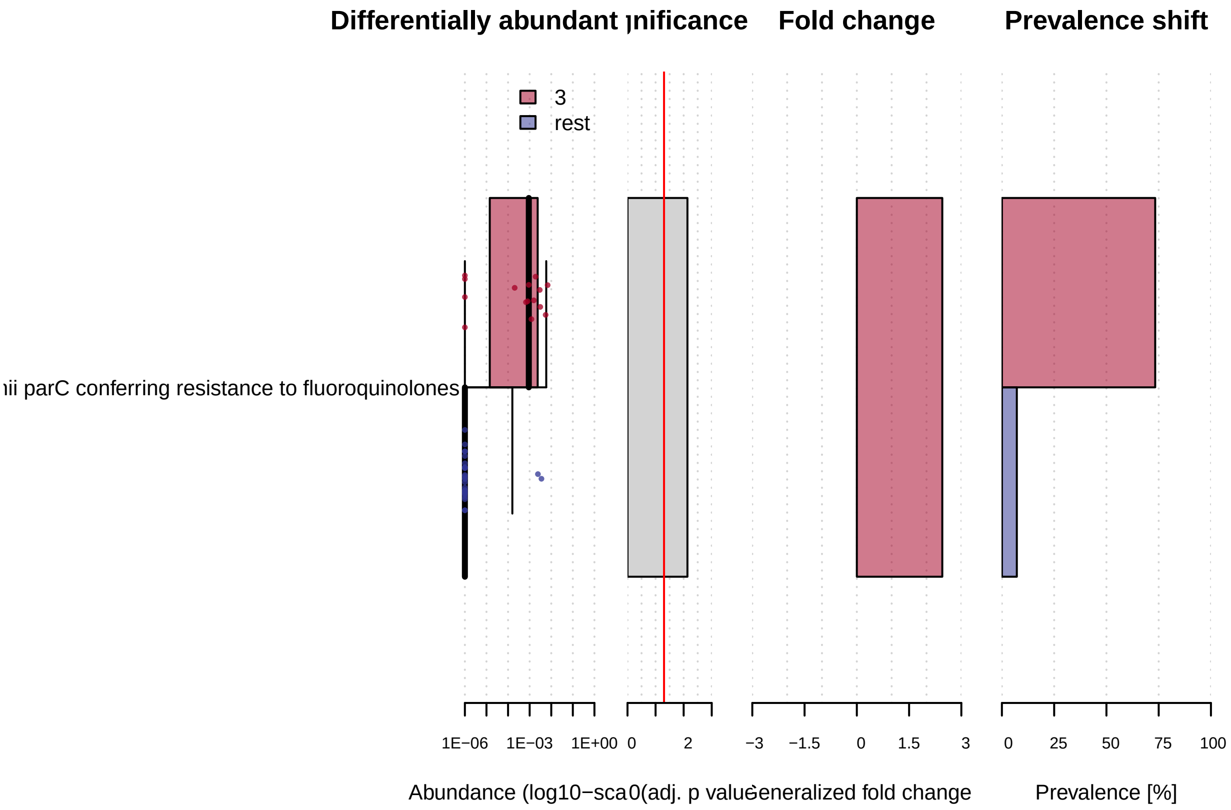


```
## plot saved to siamcat_Hospital_Type_2_final_association.svg
```



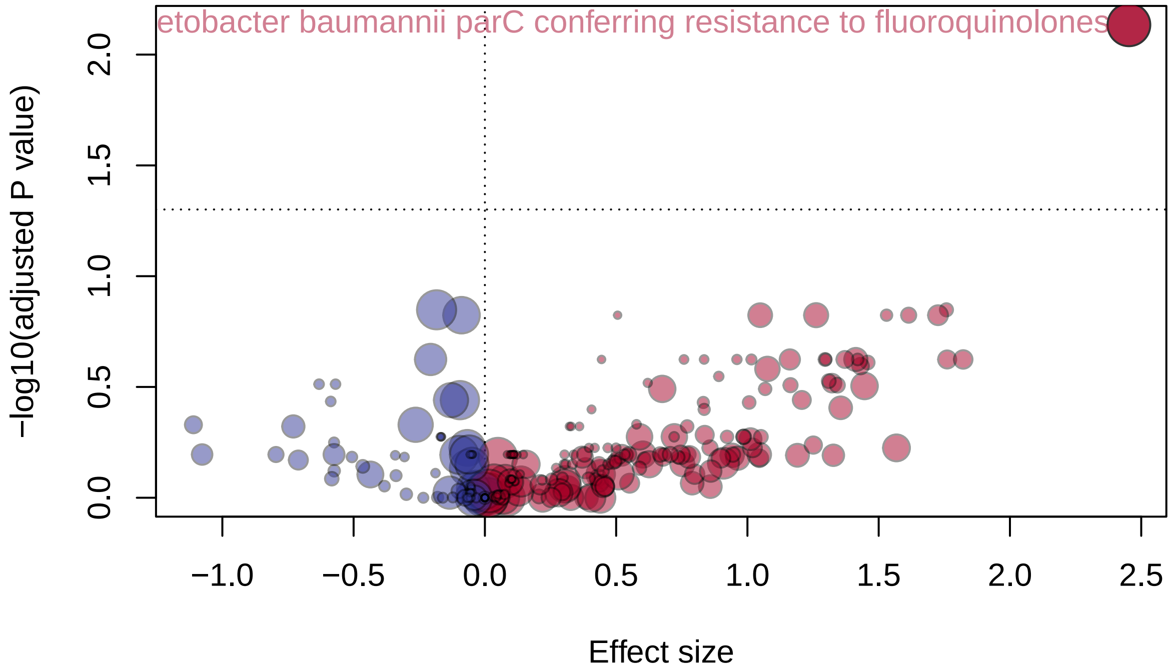
```
## [1] "oqxB"                        "Klebsiella pneumoniae KpnF"
## [3] "APH(3)-IIIa"                 "SAT-4"
## [5] "FOX-5"                       "LCR-1"
## [7] "ramA"                        "QnrD1"
## Completed processing for Hospital_Type with case 2

## Label used as case:
##    3
## Label used as control:
##    rest

## + finished create.label.from.metadata in 0.004 s

## + starting validate.data

## +++ checking overlap between labels and features

## + Keeping labels of 43 sample(s).

## +++ checking sample number per class

## +++ checking overlap between samples and metadata

## + finished validate.data in 0.055 s
## Features successfully filtered
## Features successfully filtered
```

```
## Less than 5 associations found. Consider changing your alpha value.
## Less than 5 associations found. Consider changing your alpha value.
```



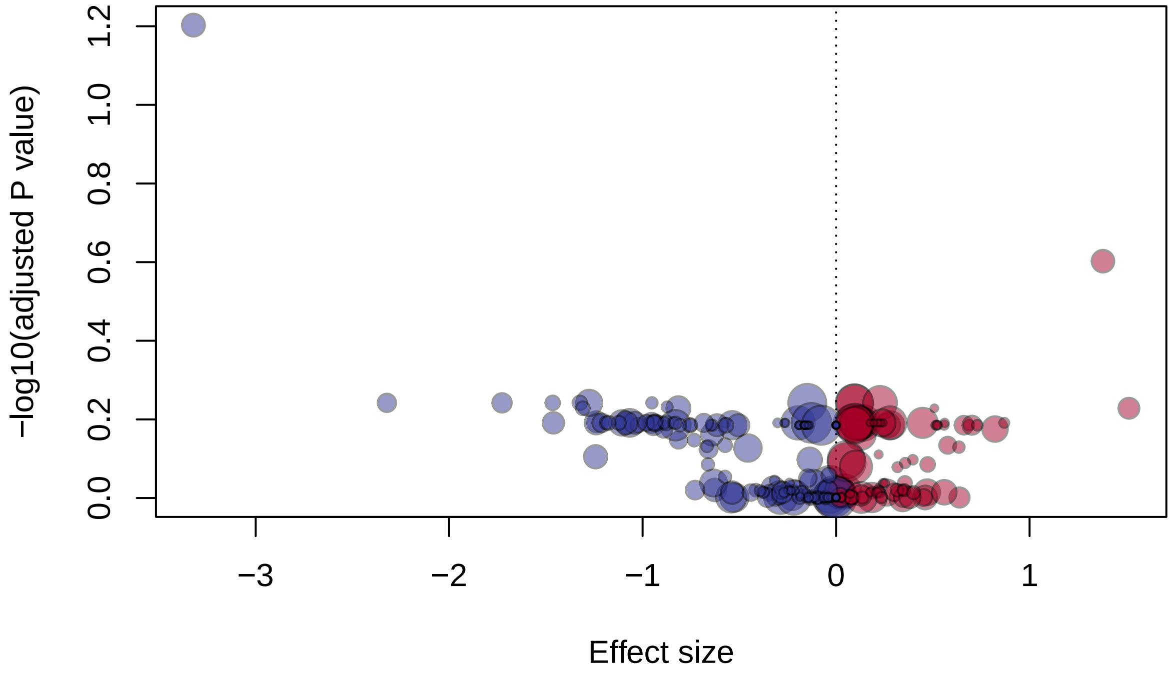
```
## plot saved to siamcat_Hospital_Type_3_final_association.svg
```

```
## Fewer significant features at alpha 0.05 than desired features for annotation (5)!
```



```
## [1] "Acinetobacter baumannii parC conferring resistance to fluoroquinolones"
## Completed processing for Hospital_Type with case 3

## Label used as case:
##    4
## Label used as control:
##    rest

## + finished create.label.from.metadata in 0.004 s

## + starting validate.data

## +++ checking overlap between labels and features

## + Keeping labels of 43 sample(s).

## +++ checking sample number per class

## +++ checking overlap between samples and metadata

## + finished validate.data in 0.053 s

## Features successfully filtered
## Features successfully filtered
## No significant associations found. No plot will be produced.
##
## No significant associations found. No plot will be produced.
```

```
## plot saved to siamcat_Hospital_Type_4_final_association.svg
```



```
## character(0)
## Completed processing for Hospital_Type with case 4
```

```
## Label used as case:
##    Specilised
## Label used as control:
##    rest
```

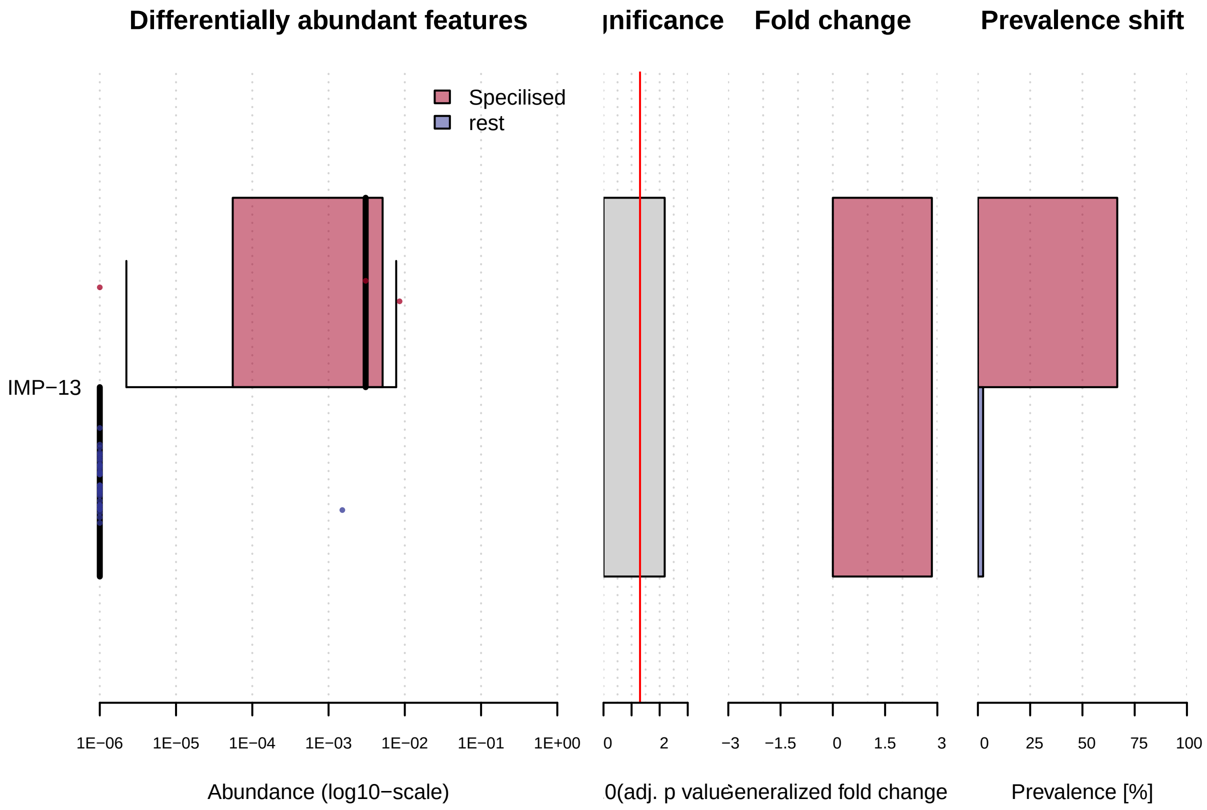
```
## + finished create.label.from.metadata in 0.005 s
```

```
## + starting validate.data
```

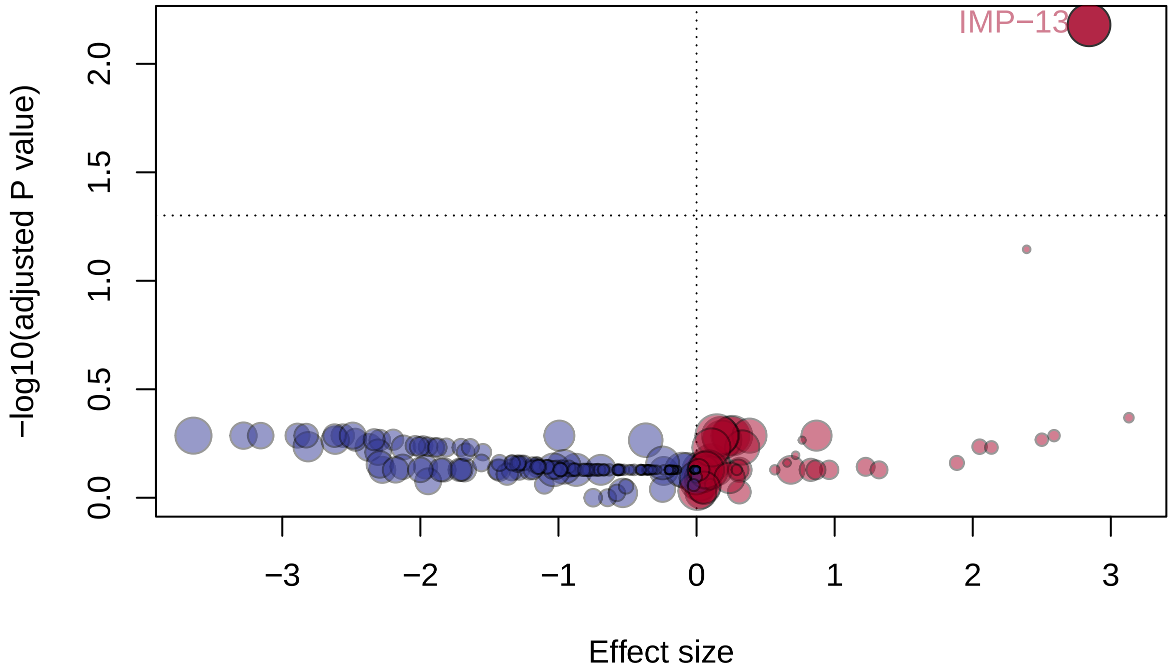
```
## +++ checking overlap between labels and features
```

```
## + Keeping labels of 43 sample(s).
```

```
## +++ checking sample number per class
## Data set has a limited number of training examples:
##  rest    40
##  Specilised  3
## Note that a dataset this small/skewed is not necessarily suitable for analysis in this pipeline.

## +++ checking overlap between samples and metadata

## + finished validate.data in 0.052 s

## Features successfully filtered
## Features successfully filtered

## Less than 5 associations found. Consider changing your alpha value.
## Less than 5 associations found. Consider changing your alpha value.
```



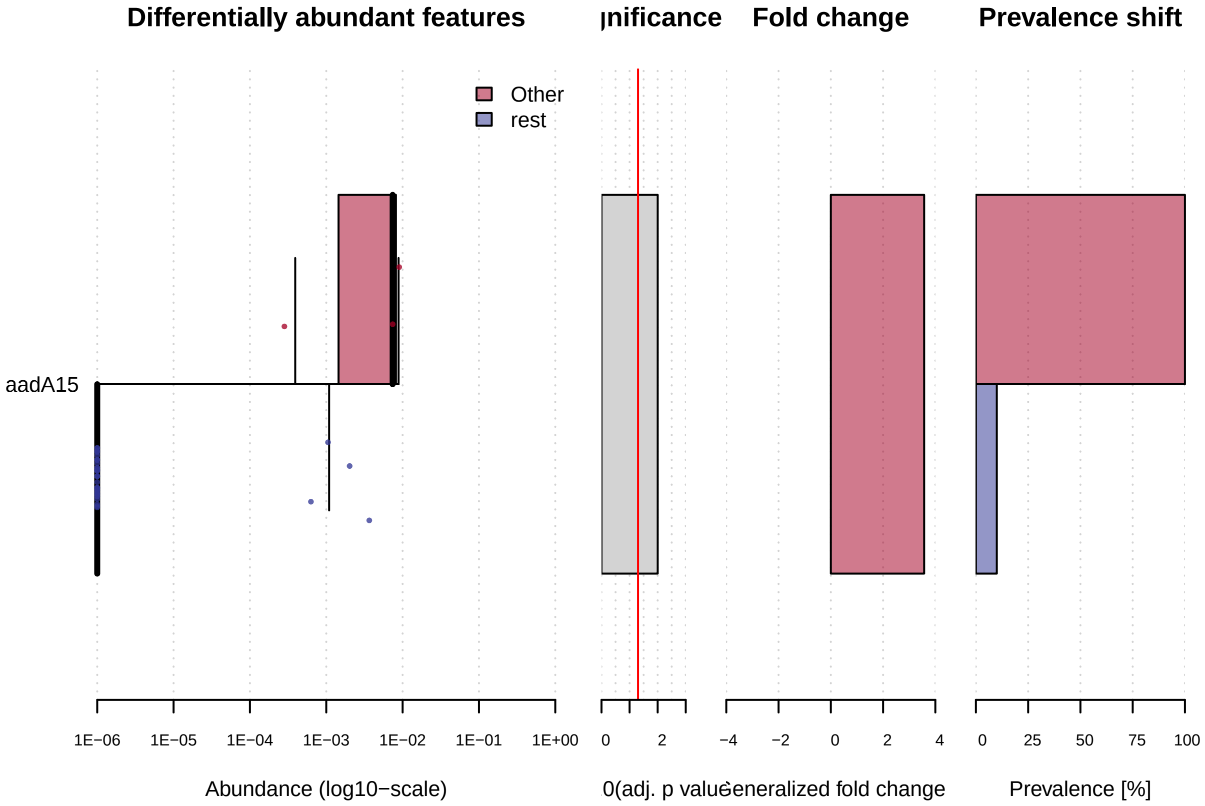
```
## plot saved to siamcat_Hospital_Type_Specilised_final_association.svg

## Fewer significant features at alpha 0.05 than desired features for annotation (5)!
```



```
## [1] "IMP-13"
## Completed processing for Hospital_Type with case Specilised

## Label used as case:
##    Other
## Label used as control:
##    rest

## + finished create.label.from.metadata in 0.006 s

## + starting validate.data

## +++ checking overlap between labels and features

## + Keeping labels of 43 sample(s).

## +++ checking sample number per class

## Data set has a limited number of training examples:
##  rest    40
##  Other   3
## Note that a dataset this small/skewed is not necessarily suitable for analysis in this pipeline.

## +++ checking overlap between samples and metadata
## + finished validate.data in 0.069 s
```

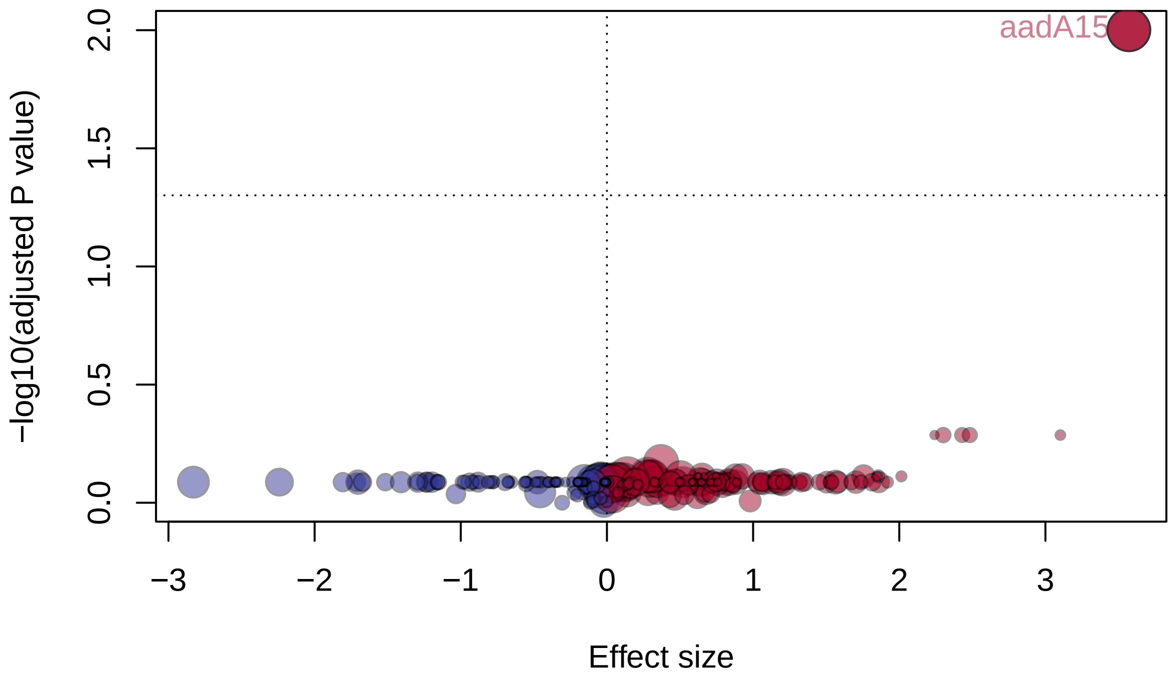
```
## Features successfully filtered
## Features successfully filtered
```

```
## Less than 5 associations found. Consider changing your alpha value.
## Less than 5 associations found. Consider changing your alpha value.
```



```
## plot saved to siamcat_Hospital_Type_Other_final_association.svg
```

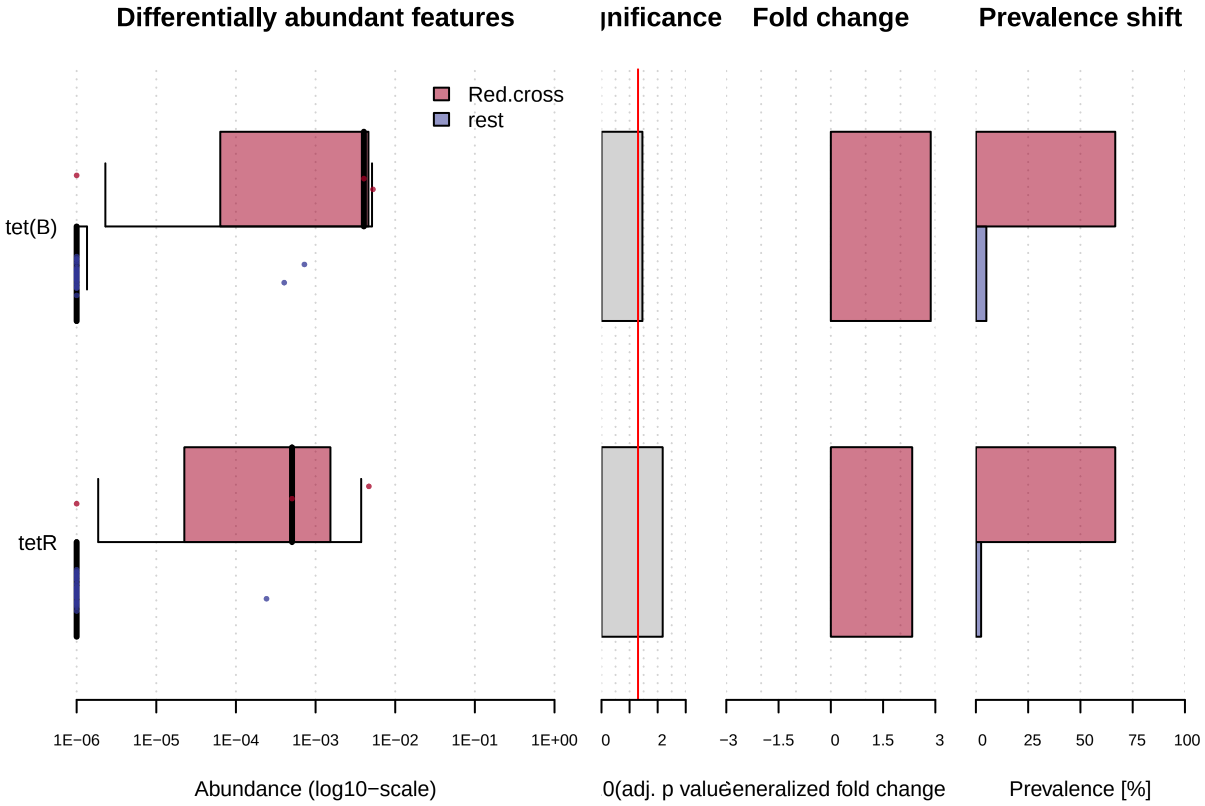
```
## Fewer significant features at alpha 0.05 than desired features for annotation (5)!
```



```
## [1] "aadA15"
## Completed processing for Hospital_Type with case Other

## Label used as case:
##    Red_cross
## Label used as control:
##    rest

## + finished create.label.from.metadata in 0.005 s

## + starting validate.data

## +++ checking overlap between labels and features

## + Keeping labels of 43 sample(s).

## +++ checking sample number per class

## Data set has a limited number of training examples:
##  rest    40
##  Red.cross   3
## Note that a dataset this small/skewed is not necessarily suitable for analysis in this pipeline.

## +++ checking overlap between samples and metadata
## + finished validate.data in 0.059 s
```

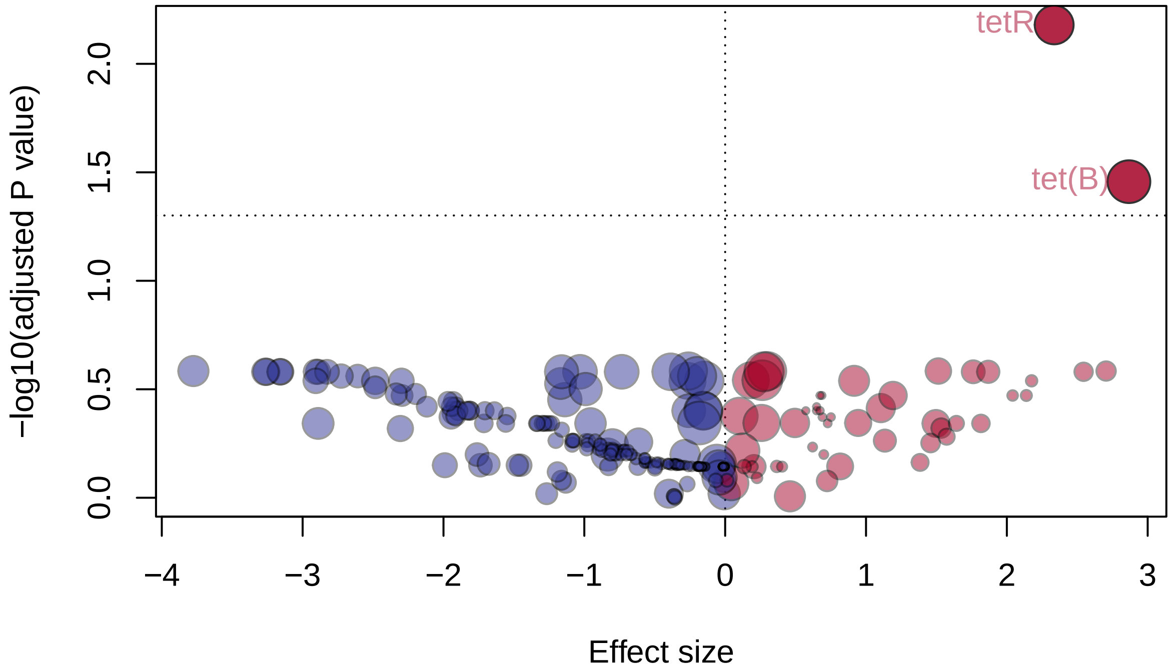
```
## Features successfully filtered
## Features successfully filtered
```

```
## Less than 5 associations found. Consider changing your alpha value.
## Less than 5 associations found. Consider changing your alpha value.
```



```
## plot saved to siamcat_Hospital_Type_Red_cross_final_association.svg
```

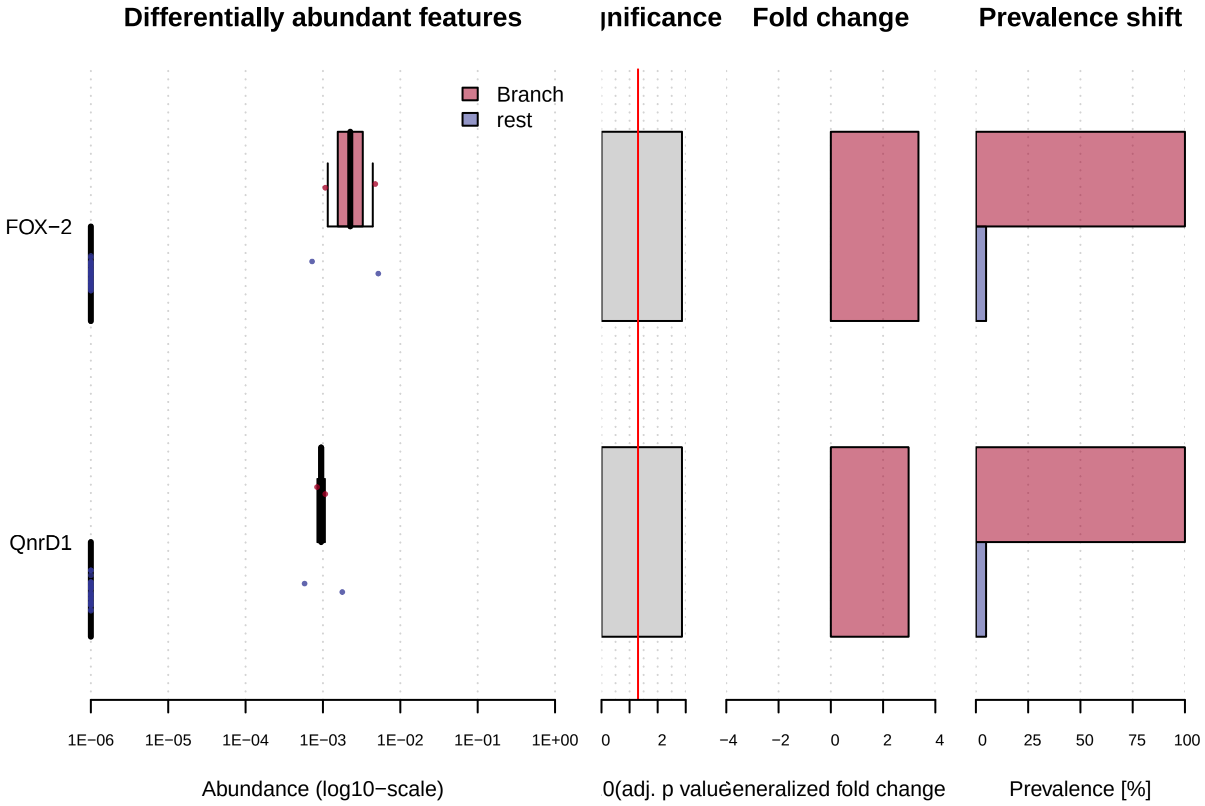
```
## Fewer significant features at alpha 0.05 than desired features for annotation (5)!
```



```
## [1] "tet(B)" "tetR"
## Completed processing for Hospital_Type with case Red_cross

## Label used as case:
##    Branch
## Label used as control:
##    rest

## + finished create.label.from.metadata in 0.005 s

## + starting validate.data

## +++ checking overlap between labels and features

## + Keeping labels of 43 sample(s).

## +++ checking sample number per class

## Data set has a limited number of training examples:
##  rest    41
##  Branch  2
## Note that a dataset this small/skewed is not necessarily suitable for analysis in this pipeline.

## +++ checking overlap between samples and metadata
## + finished validate.data in 0.263 s
```

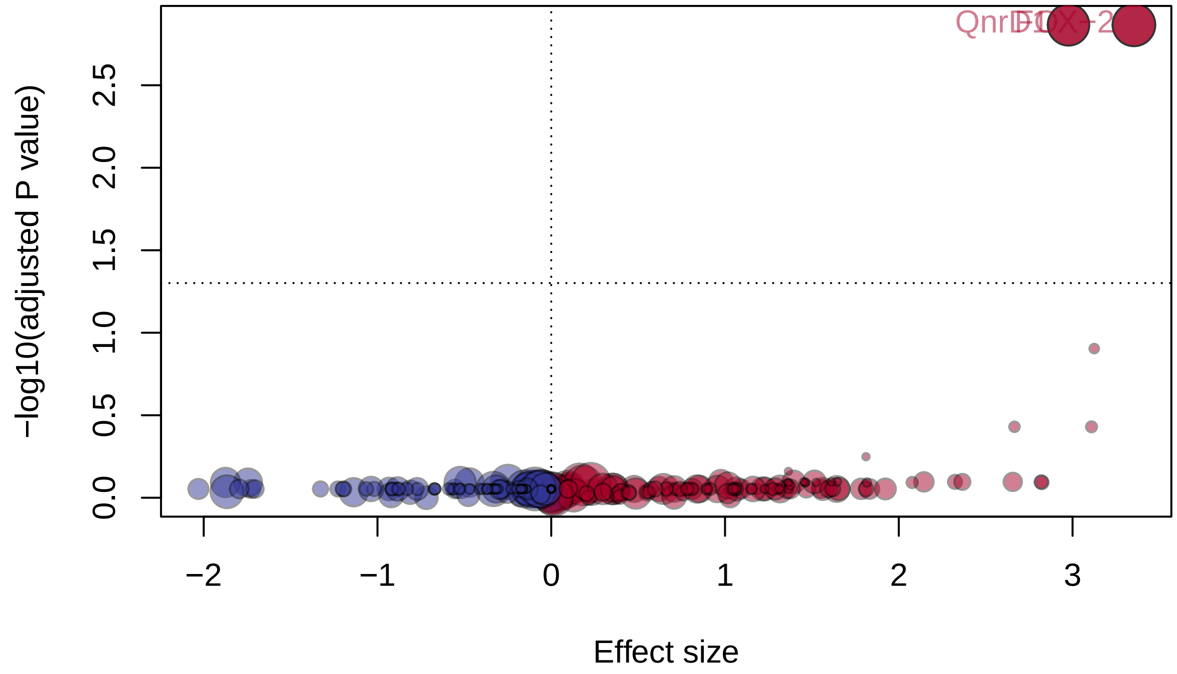
```
## Features successfully filtered
## Features successfully filtered
```

```
## Less than 5 associations found. Consider changing your alpha value.
## Less than 5 associations found. Consider changing your alpha value.
```



```
## plot saved to siamcat_Hospital_Type_Branch_final_association.svg
```

```
## Fewer significant features at alpha 0.05 than desired features for annotation (5)!
```



```
## [1] "FOX-2" "QnrD1"
## Completed processing for Hospital_Type with case Branch
```

Industry

In this section industrial factor importance for changes in diversity are evaluated.

Show what are factors used.

```
colnames(sample_data(ps_amr_CSS))[7:12]
```

```
## [1] "Industrial_wastewater_impact"
## [2] "Industrial_wastewater_impact_from_food"
## [3] "Dairy_farming"
## [4] "Meat_production"
## [5] "Metal_processing"
## [6] "Washrooms"
```

```
# remove NA values
ps_amr_ww_impact = subset_samples(ps_scaffolds_filtered, sample_data(ps_scaffolds_filtered)$Industrial_
alpha_indexes_ww_impact <- estimate_richness(ps_amr_ww_impact, split = TRUE, c("Shannon", "Simpson", "In
kruskal.test(alpha_indexes_ww_impact$Shannon ~ sample_data(ps_amr_ww_impact)$Industrial_wastewater_impac
```

Industrial_wastewater_impact:

```
##
##  Kruskal-Wallis rank sum test
##
## data:  alpha_indexes_ww_impact$Shannon by sample_data(ps_amr_ww_impact)$Industrial_wastewater_impact
## Kruskal-Wallis chi-squared = 5.4309, df = 3, p-value = 0.1428
```

```
kruskal.test(alpha_indexes_ww_impact$Simpson ~ sample_data(ps_amr_ww_impact)$Industrial_wastewater_impac
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  alpha_indexes_ww_impact$Simpson by sample_data(ps_amr_ww_impact)$Industrial_wastewater_impact
## Kruskal-Wallis chi-squared = 4.4167, df = 3, p-value = 0.2198
```

Find which pairs are important

```
dunn_results <- dunn.test(alpha_indexes_ww_impact$Shannon,
          sample_data(ps_amr_ww_impact)$Industrial_wastewater_impact,
          method="bh")
```

```
##   Kruskal-Wallis rank sum test
##
## data: x and group
## Kruskal-Wallis chi-squared = 5.4309, df = 3, p-value = 0.14
##
##
##                              Comparison of x by group
##                                (Benjamini-Hochberg)
## Col Mean-|
## Row Mean |       High        Low     Medium
## ---------+---------------------------------
##      Low |  -0.693831
##          |     0.2927
##          |
##   Medium |   0.090955   0.952157
##          |     0.4638     0.2558
##          |
## Seasonal |  -2.019413  -1.708128  -2.224614
##          |     0.0652     0.0876     0.0783
##
## alpha = 0.05
## Reject Ho if p <= alpha/2
```

```r
# remove NA values
ps_amr_ww_impact = subset_samples(ps_scaffolds_filtered, sample_data(ps_scaffolds_filtered)$Industrial_w

alpha_indexes_ww_impact <- estimate_richness(ps_amr_ww_impact, split = TRUE, c("Shannon", "Simpson", "I
kruskal.test(alpha_indexes_ww_impact$Shannon ~ sample_data(ps_amr_ww_impact)$Industrial_wastewater_impac
```

Industrial_wastewater_impact_from_food

```
##
##  Kruskal-Wallis rank sum test
##
## data:  alpha_indexes_ww_impact$Shannon by sample_data(ps_amr_ww_impact)$Industrial_wastewater_impact_
## Kruskal-Wallis chi-squared = 4.9957, df = 3, p-value = 0.1721
```

```r
kruskal.test(alpha_indexes_ww_impact$Simpson ~ sample_data(ps_amr_ww_impact)$Industrial_wastewater_impac
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  alpha_indexes_ww_impact$Simpson by sample_data(ps_amr_ww_impact)$Industrial_wastewater_impact_
## Kruskal-Wallis chi-squared = 3.6937, df = 3, p-value = 0.2965
```

Find which pairs are important

```r
dunn_results <- dunn.test(alpha_indexes_ww_impact$Shannon,
          sample_data(ps_amr_ww_impact)$Industrial_wastewater_impact_from_food,
          method="bh")
```

```
##   Kruskal-Wallis rank sum test
##
## data: x and group
## Kruskal-Wallis chi-squared = 4.9957, df = 3, p-value = 0.17
##
##
##                              Comparison of x by group
##                                (Benjamini-Hochberg)
## Col Mean-|
## Row Mean |       High        Low     Medium
## ---------+---------------------------------
##      Low |  -0.519487
##          |     0.3621
##          |
##   Medium |   0.295679   0.828486
##          |     0.3837     0.3055
##          |
## Seasonal |  -1.968065  -1.728197  -2.150237
##          |     0.0736     0.0840     0.0946
##
## alpha = 0.05
## Reject Ho if p <= alpha/2
```

Binary Factor alpha diversity Municipalities were classefied by Dairy_farming, Meat_production, Metal_processing and car washing facilites connected to wastewater system.

```
wilcox.test(alpha_indexes_amr$Shannon~sample_data(ps_scaffolds_filtered)$Dairy_farming, p.adjust.method
```

```
##
##  Wilcoxon rank sum exact test
##
## data:  alpha_indexes_amr$Shannon by sample_data(ps_scaffolds_filtered)$Dairy_farming
## W = 285, p-value = 0.1865
## alternative hypothesis: true location shift is not equal to 0
```

```
wilcox.test(alpha_indexes_amr$Shannon~sample_data(ps_scaffolds_filtered)$Meat_production, p.adjust.metho
```

```
##
##  Wilcoxon rank sum exact test
##
## data:  alpha_indexes_amr$Shannon by sample_data(ps_scaffolds_filtered)$Meat_production
## W = 122, p-value = 0.5917
## alternative hypothesis: true location shift is not equal to 0
```

```
wilcox.test(alpha_indexes_amr$Shannon~sample_data(ps_scaffolds_filtered)$Metal_processing, p.adjust.meth
```

```
##
##  Wilcoxon rank sum exact test
##
## data:  alpha_indexes_amr$Shannon by sample_data(ps_scaffolds_filtered)$Metal_processing
## W = 168, p-value = 0.398
## alternative hypothesis: true location shift is not equal to 0
```

```
wilcox.test(alpha_indexes_amr$Shannon~sample_data(ps_scaffolds_filtered)$Washrooms, p.adjust.method = "
```

```
##
##  Wilcoxon rank sum exact test
##
## data:  alpha_indexes_amr$Shannon by sample_data(ps_scaffolds_filtered)$Washrooms
## W = 51, p-value = 0.01183
## alternative hypothesis: true location shift is not equal to 0
```

```
wilcox.test(alpha_indexes_amr$Simpson~sample_data(ps_scaffolds_filtered)$Dairy_farming, p.adjust.method
```

```
##
##  Wilcoxon rank sum exact test
##
## data:  alpha_indexes_amr$Simpson by sample_data(ps_scaffolds_filtered)$Dairy_farming
## W = 261, p-value = 0.4616
## alternative hypothesis: true location shift is not equal to 0
```

```
wilcox.test(alpha_indexes_amr$Simpson~sample_data(ps_scaffolds_filtered)$Meat_production, p.adjust.metho
```

```
##
##  Wilcoxon rank sum exact test
##
## data:  alpha_indexes_amr$Simpson by sample_data(ps_scaffolds_filtered)$Meat_production
## W = 101, p-value = 0.2345
## alternative hypothesis: true location shift is not equal to 0
```

```
wilcox.test(alpha_indexes_amr$Simpson~sample_data(ps_scaffolds_filtered)$Metal_processing, p.adjust.meth
```

```
##
##  Wilcoxon rank sum exact test
##
## data:  alpha_indexes_amr$Simpson by sample_data(ps_scaffolds_filtered)$Metal_processing
## W = 161, p-value = 0.5295
## alternative hypothesis: true location shift is not equal to 0
```

```
wilcox.test(alpha_indexes_amr$Simpson~sample_data(ps_scaffolds_filtered)$Washrooms, p.adjust.method = "
```

```
##
##  Wilcoxon rank sum exact test
##
## data:  alpha_indexes_amr$Simpson by sample_data(ps_scaffolds_filtered)$Washrooms
## W = 88, p-value = 0.2227
## alternative hypothesis: true location shift is not equal to 0
```

Beta diversity industry

```
# Microbial Community Diversity Analysis Tutorial with Phyloseq
# Permanova
# https://deneflab.github.io/MicrobeMiseq/demos/mothur_2_phyloseq.html

# remove NAs
ps_amr_ww_impact = subset_samples(ps_amr_CSS, sample_data(ps_amr_CSS)$Industrial_wastewater_impact != "
tax_bray_ww <- phyloseq::distance(ps_amr_ww_impact, method = "bray")

adonis2_rez<-adonis2(tax_bray_ww ~ sample_data(ps_amr_ww_impact)$Industrial_wastewater_impact)
print(adonis2_rez)
```

```
## Permutation test for adonis under reduced model
## Permutation: free
## Number of permutations: 999
##
## adonis2(formula = tax_bray_ww ~ sample_data(ps_amr_ww_impact)$Industrial_wastewater_impact)
##          Df SumOfSqs      R2      F Pr(>F)
## Model     3   0.5789 0.12307 1.8245  0.025 *
## Residual 39   4.1246 0.87693
## Total    42   4.7034 1.00000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

From Food

```
# remove NAs
ps_amr_ww_impact = subset_samples(ps_amr_CSS, sample_data(ps_amr_CSS)$Industrial_wastewater_impact_from_

tax_bray_ww <- phyloseq::distance(ps_amr_ww_impact, method = "bray")

adonis2_rez<-adonis2(tax_bray_ww ~ sample_data(ps_amr_ww_impact)$Industrial_wastewater_impact_from_food
print(adonis2_rez)
```

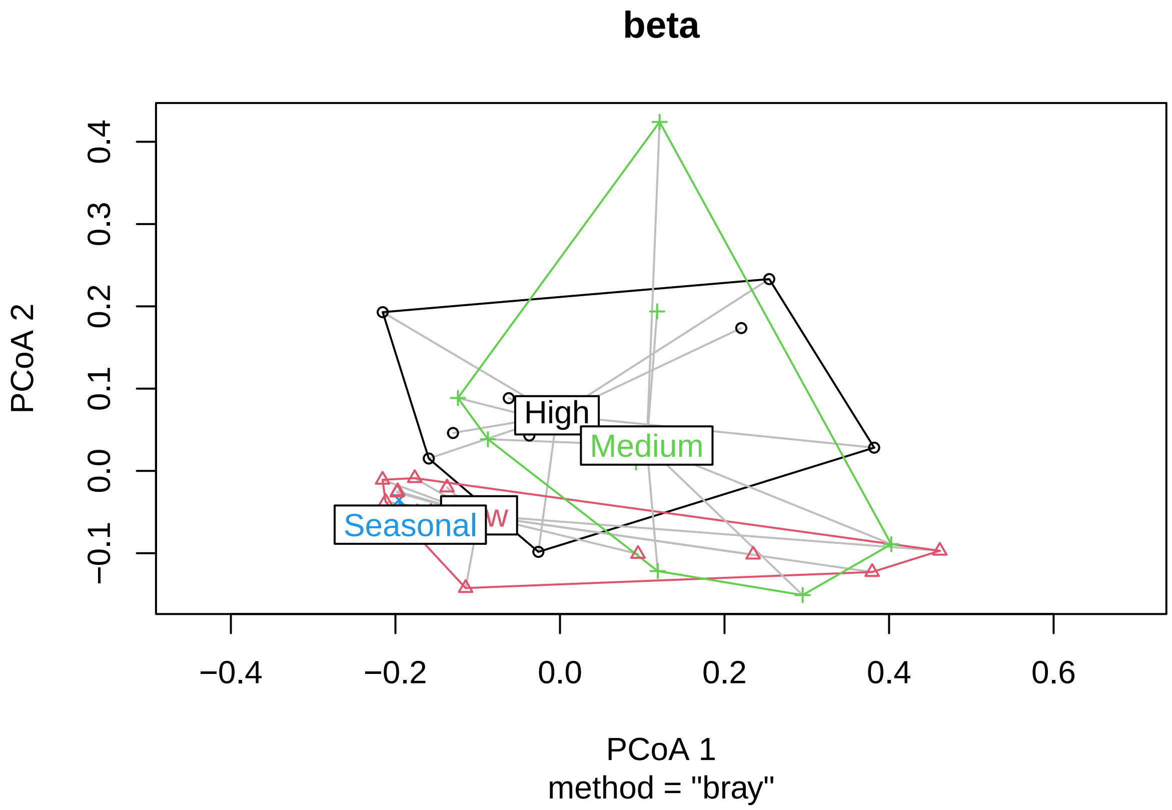
```
## Permutation test for adonis under reduced model
## Permutation: free
## Number of permutations: 999
##
## adonis2(formula = tax_bray_ww ~ sample_data(ps_amr_ww_impact)$Industrial_wastewater_impact_from_food
##          Df SumOfSqs      R2      F Pr(>F)
## Model     3   0.6642 0.16758 2.0803  0.018 *
## Residual 31   3.2994 0.83242
## Total    34   3.9637 1.00000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
metadata_col = colnames(sample_data(ps_amr_CSS))[9:12]
```

```
beta <- betadisper(tax_bray_ww, sample_data(ps_amr_ww_impact)$Industrial_wastewater_impact_from_food)
print(anova(beta))
```

```
## Analysis of Variance Table
##
## Response: Distances
##           Df  Sum Sq  Mean Sq F value Pr(>F)
## Groups     3 0.08412 0.028040  2.4393 0.0831 .
## Residuals 31 0.35635 0.011495
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
plot(beta)
```



Show rest of the factors.

```
set.seed(7)

for (param in metadata_col){
# Using Bray-Curtis distance by default
if (param == "norm_factor" || param == "Sample" || param == "Date"){
    next
}

print(param)
formula_str <- paste("tax_bray_amr ~", param)
adonis2_rez<-adonis2(as.formula(formula_str), data = data.frame(sample_data(ps_amr_CSS)))

# View results
print(adonis2_rez)

}
```

```
## [1] "Dairy_farming"
## Permutation test for adonis under reduced model
## Permutation: free
## Number of permutations: 999
##
## adonis2(formula = as.formula(formula_str), data = data.frame(sample_data(ps_amr_CSS)))
##          Df SumOfSqs      R2      F Pr(>F)
## Model     1   0.1853 0.03941 1.682  0.104
## Residual 41   4.5181 0.96059
## Total    42   4.7034 1.00000
## [1] "Meat_production"
## Permutation test for adonis under reduced model
## Permutation: free
## Number of permutations: 999
##
## adonis2(formula = as.formula(formula_str), data = data.frame(sample_data(ps_amr_CSS)))
##          Df SumOfSqs      R2      F Pr(>F)
## Model     1   0.1392 0.02959 1.2501    0.2
## Residual 41   4.5643 0.97041
## Total    42   4.7034 1.00000
## [1] "Metal_processing"
## Permutation test for adonis under reduced model
## Permutation: free
## Number of permutations: 999
##
## adonis2(formula = as.formula(formula_str), data = data.frame(sample_data(ps_amr_CSS)))
##          Df SumOfSqs      R2     F Pr(>F)
## Model     1   0.1102 0.02344 0.984  0.383
## Residual 41   4.5932 0.97656
## Total    42   4.7034 1.00000
## [1] "Washrooms"
## Permutation test for adonis under reduced model
## Permutation: free
## Number of permutations: 999
##
## adonis2(formula = as.formula(formula_str), data = data.frame(sample_data(ps_amr_CSS)))
##          Df SumOfSqs      R2      F Pr(>F)
## Model     1   0.2435 0.05177 2.2386  0.043 *
## Residual 41   4.4599 0.94823
## Total    42   4.7034 1.00000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```