Supplemental File of

MicrobiotaProcess: A tidy framework facilitating microbiome or other related ecology data reproducible analysis

Shuangbin Xu, Li Zhan, Wenli Tang, Zehan Dai, Lang Zhou, Tingze Feng, Meijun Chen, Shanshan Liu, Xiaocong Fu, Tianzhi Wu, Erqiang Hu and Guangchuang Yu*

*correspondence: Guangchuang Yu <gcyu1@smu.edu.cn>

1 Installation

To install `MicrobiotaProcess` package, please enter the following command in R:

```
if (!requireNamespace("BiocManager", quietly = TRUE))
   install.packages("BiocManager")
BiocManager::install("MicrobiotaProcess")
```

To reproduce the analysis in this document, the several extra packages also needed to be installed.

```
cranpkgs <- c("aplot", "ggpp",
              "broom", "forcats",
              "ggrepel", "ggVennDiagram",
              "patchwork", "shadowtext",
              "ggupset", "ggnewscale")

for (i in cranpkgs){
    if (!requireNamespace(i, quietly = TRUE)){
        install.packages(i)
    }
}

Biocpkgs <- c("SummarizedExperiment", "clusterProfiler",
              "edgeR", "enrichplot", "tidybulk",
              "ggtree", "ggtreeExtra", "MicrobiomeProfiler")

for (i in Biocpkgs){
    if (!requireNamespace(i, quietly = TRUE)){
        BiocManager::install(i)
    }
}
```

2 Analysis of 16s rDNA dataset about 43 pediatric CD stool samples from iHMP

Here, we use the 43 pediatric IBD stool samples as example, which were obtained from the Integrative Human Microbiome Project Consortium (iHMP) (Research Network Consortium 2014).

2.1 Importing the output of dada2

The datasets were downloaded from web[1]. It contains `ibd_asv_table.txt`, which is feature table (*row features* X *column samples*), `ibd_meta.csv` (metadata file of samples), and `ibd_taxa.txt` (the taxonomic annotation of features). In the session, we use *mp_import_dada2* of *MicrobiotaProcess* to import the dataset, and return a *MPSE* object.

```
library(MicrobiotaProcess)
otuda <- read.table("./data/IBD_data/ibd_asv_table.txt", header=T,
                    check.names=F, comment.char="", row.names=1, sep="\t")
```

[1] https://www.microbiomeanalyst.ca/MicrobiomeAnalyst/resources/data/ibd_data.zip

```
# building the output format of removeBimeraDenovo of dada2
otuda <- data.frame(t(otuda), check.names=F)
sampleda <- read.csv("./data/IBD_data/ibd_meta.csv", row.names=1, comment.char="")
taxda <- read.table("./data/IBD_data/ibd_taxa.txt", header=T,
                    row.names=1, check.names=F, comment.char="")
# the feature names should be the same with rownames of taxda.
taxda <- taxda[match(colnames(otuda), rownames(taxda)),]
mpse <- mp_import_dada2(seqtab = otuda, taxatab = taxda, sampleda = sampleda)
# view the reads depth of samples and the prevalence of the OTUs. In this example,
# mpse %>% mp_extract_assay(.abundant=Abundance) %>% rowSums() %>% sort %>% head(100)
# mpse %>% mp_extract_assay(.abundant=Abundance) %>% colSums() %>% sort %>% head()
# Or
# head(sort(rowSums(assay(mpse, "Abundance"))), 100)
# head(sort(colSums(assay(mpse, "Abundance"))))
# In this example, we can find some OTUs have very low frequency in the samples.
# and some taxonomy are unreasonable, for example, the probability of chloroplasts
# in the intestine should be low. We can also remove the features.
mpse2 <- mpse %>%
         dplyr::filter(!Phylum %in% c("p__un_k__Bacteria", "p__Chloroflexi") &
                       !Class %in% "c__Chloroplast" &
                       !Family %in% "f__mitochondria"
         ) %>%
         mp_filter_taxa(.abundance = Abundance, min.abun = 1, min.prop = 0.1)
mpse2
```

```
## # A MPSE-tibble (MPSE object) abstraction: 9,890 x 11
## # OTU=230 | Samples=43 | Assays=Abundance | Taxanomy=Kingdom, Phylum, Class, Order, Family, Genus, Species
##    OTU    Sample Abundance Group Kingdom Phylum Class Order Family Genus Species
##    <chr>  <chr>      <int> <chr> <chr>   <chr>  <chr> <chr> <chr>  <chr> <chr>
##  1 OTU_2~ S2067~         0 CD    k__Bac~ p__Ac~ c__A~ o__A~ f__Ac~ g__A~ s__un_~
##  2 OTU_5~ S2067~         0 CD    k__Bac~ p__Ac~ c__A~ o__A~ f__Ac~ g__A~ s__un_~
##  3 OTU_7~ S2067~         0 CD    k__Bac~ p__Ac~ c__A~ o__A~ f__Mi~ g__R~ s__muc~
##  4 OTU_42 S2067~         0 CD    k__Bac~ p__Ac~ c__A~ o__B~ f__Bi~ g__B~ s__ado~
##  5 OTU_1~ S2067~         0 CD    k__Bac~ p__Ac~ c__A~ o__B~ f__Bi~ g__B~ s__un_~
##  6 OTU_1~ S2067~         0 CD    k__Bac~ p__Ac~ c__A~ o__B~ f__Bi~ g__B~ s__un_~
##  7 OTU_3~ S2067~         0 CD    k__Bac~ p__Ac~ c__C~ o__C~ f__Co~ g__A~ s__un_~
##  8 OTU_1~ S2067~         0 CD    k__Bac~ p__Ac~ c__C~ o__C~ f__Co~ g__C~ s__aer~
##  9 OTU_3~ S2067~         0 CD    k__Bac~ p__Ac~ c__C~ o__C~ f__Co~ g__E~ s__len~
## 10 OTU_1~ S2067~         0 CD    k__Bac~ p__Ba~ c__B~ o__B~ f__[O~ g__O~ s__un_~
## # ... with 9,880 more rows
```

2.2 Other import functions

MicrobiotaProcess also presents some other import functions S1 to parse the output of the upstream pipelines. In addition, some common object of R can also be converted to *MPSE* object, such as *phyloseq* (McMurdie 2013), *SummarizedExperiment* (Morgan et al. 2021), *TreeSummarizedExperiment* (Huang et al. 2021), *biom* (McMurdie and Paulson 2021) (output of *biomformat* by *read_biom*) 3.1.

Table S1: List of import functions provided by MicrobiotaProcess

Package	Import Function	Description
MicrobiotaProcess	mp_import_qiime2	Import function to load the output of qiime2
	mp_import_qiime	Import function to read the now legacy-format QIIME OTU table (tsv format)
	mp_import_metaphlan	Import function to read the output of MetaPhlAn

2.3 alpha diversity analysis

2.3.1 rarefaction visualization

Rarefaction, based on sampling technique, was used to compensate for the effect of sample size on the number of units observed in a sample. *MicrobiotaProcess* provided *mp_cal_rarecurve* and *mp_plot_rarecurve* to calculate and plot the curves.

```
library(MicrobiotaProcess)
library(patchwork)
cols <- c("orange", "deepskyblue")
mpse2 %<>%
    mp_rrarefy(.abundance=Abundance) %>%
    mp_cal_rarecurve(.abundance=RareAbundance, chunks=500)

p_rare <- mpse2 %>%
          mp_plot_rarecurve(
            .rare = RareAbundanceRarecurve,
            .alpha = c(Observe, Chao1, ACE),
          ) +
          theme(
            legend.key.width = unit(0.3, "cm"),
            legend.key.height = unit(0.3, "cm"),
            legend.spacing.y = unit(0.01,"cm"),
            legend.text = element_text(size=4)
          )

prare1 <- mpse2 %>%
          mp_plot_rarecurve(
            .rare = RareAbundanceRarecurve,
            .alpha = c(Observe, Chao1, ACE),
            .group = Group
          ) +
          scale_fill_manual(values = cols)+
          scale_color_manual(values = cols)+
          theme_bw()+
          theme(
            axis.text=element_text(size=8), panel.grid=element_blank(),
            strip.background = element_rect(colour=NA,fill="grey"),
            strip.text.x = element_text(face="bold")
          )

prare2 <- mpse2 %>%
          mp_plot_rarecurve(
            .rare = RareAbundanceRarecurve,
            .alpha = c(Observe, Chao1, ACE),
            .group = Group,
            plot.group = TRUE
          ) +
          scale_color_manual(values = cols)+
          scale_fill_manual(values = cols) +
          theme_bw()+
          theme(
            axis.text=element_text(size=8), panel.grid=element_blank(),
            strip.background = element_rect(colour=NA,fill="grey"),
            strip.text.x = element_text(face="bold")
          )

(p_rare / prare1 / prare2) + patchwork::plot_annotation(tag_levels="A")
```

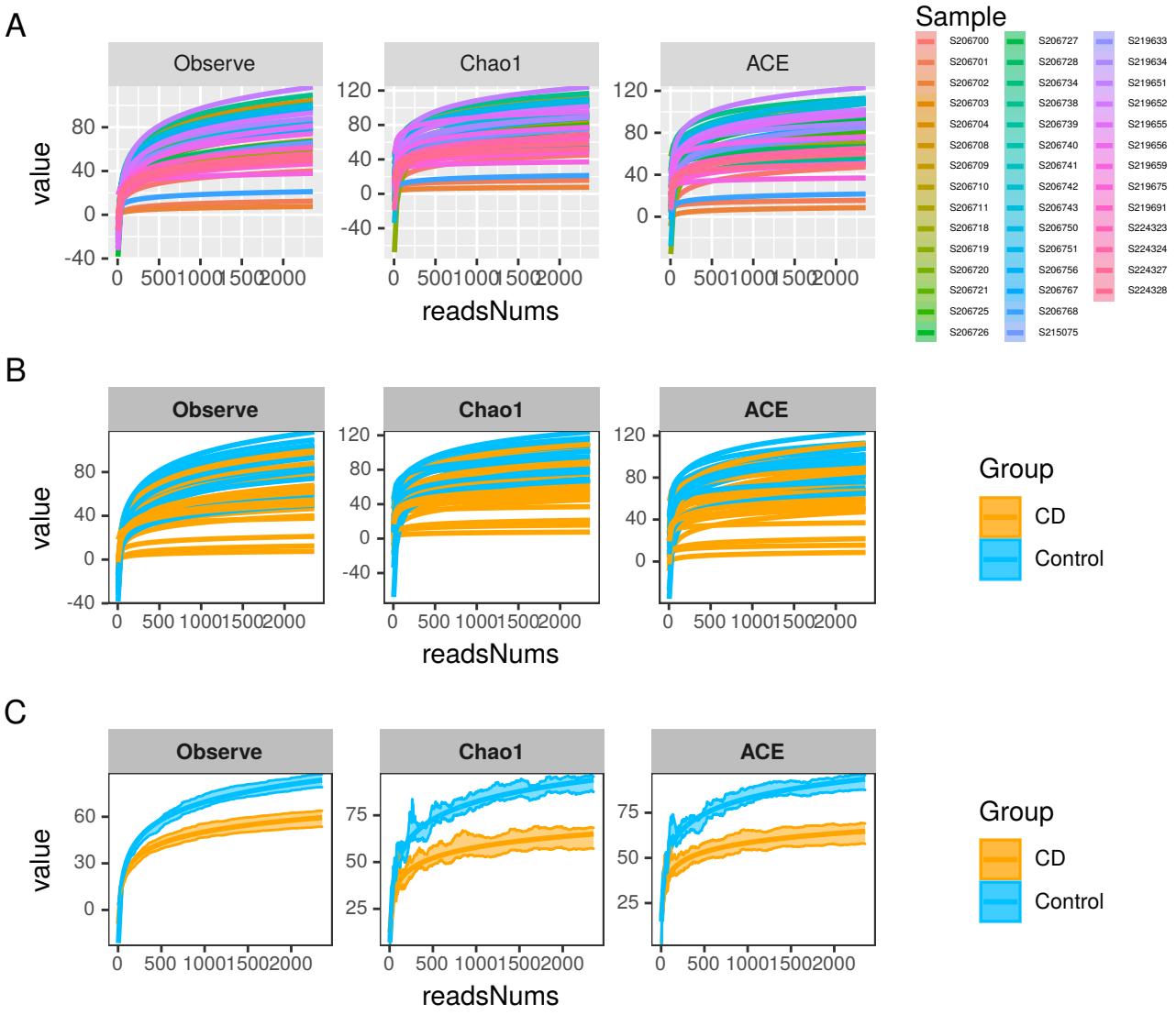

Fig. S1: **This examples show *MicrobiotaProcess* provided *mp_cal_rarecurve* and *mp_plot_rarecurve* to calculate and visualize the rarefaction curve.** The horizontal coordinate represents the sequencing depth of samples, the vertical coordinate shows the Alpha diversity index (such as Observe OTU, Chao1 and ACE). The *mp_plot_rarecurve* provides three types of visualization. (A) the rarefaction curve for each sample. (B) the rarefaction curve for each sample with colored group (specified *.group* argument in *mp_plot_rarecurve*). (C) the rarefaction curve for each group with standard error of the mean (specified *.group* argument and *plot.group=TRUE* in *mp_plot_rarecurve*)

Since the curves in each sample were near saturation, the sequencing data were great enough with very few new species undetected

2.3.2 Calculation and different analysis of alpha index

Alpha index can evaluate the richness and abundance of microbial communities. *MicrobiotaProcess* provides *mp_cal_alpha* to calculate alpha index. Six common diversity measures (*Observe*, *Chao1*, *ACE*, *Shannon*, *Simpson*, *Pielou*) are supported. And the different groups of samples can be tested and visualize by *mp_plot_alpha*. This following example shows how to use *mp_cal_alpha* and *mp_plot_alpha* of *MicrobiotaProcess* to analysis the alpha diversity of the community. The *RareAbundance* is rarefied (default), which will be used to calculate the alpha diversity index, users can specified the *force=TRUE* of *mp_cal_alpha* to calculated the index if the abundance is not be rarefied (**??**).

```
library(MicrobiotaProcess)
mpse2 %<>% mp_cal_alpha(.abundance = RareAbundance)
p_alpha <- mpse2 %>%
           mp_plot_alpha(
              .alpha = c(Observe, Chao1, ACE, Shannon, Simpson, Pielou),
```

```
            .group = Group,
        ) +
        scale_fill_manual(values=cols)+
        scale_color_manual(values=cols) +
        theme(legend.position="none",
              strip.background = element_rect(colour=NA, fill="grey"))
p_alpha
```

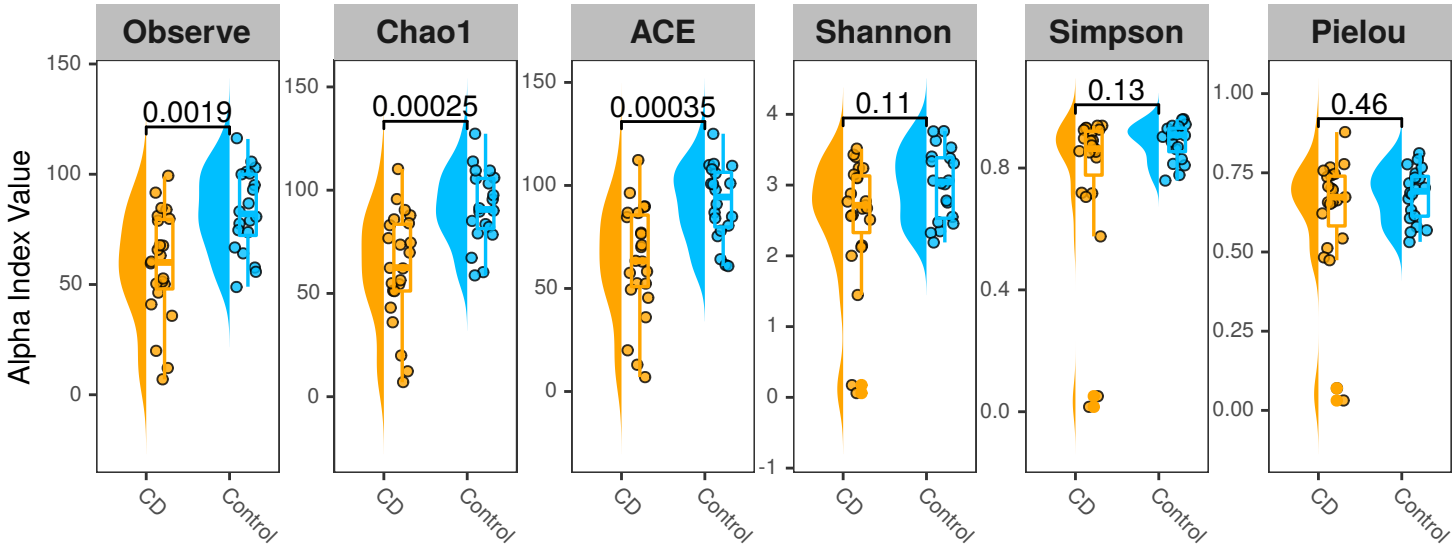

Fig. S2: **The raincloud plot of alpha diversity index** The horizontal coordinate represents each group (by *.group* argument of *mp_plot_alpha*), the vertical coordinate represents the alpha diversity index.

2.4 Taxonomy composition analysis

2.4.1 Statistics and visualization of specific levels

MicrobiotaProcess presents the *mp_cal_abundance* and *mp_plot_abundance* for the calculation and visualization of composition of microbial communities. After the *mp_cal_abundance* done, you can get the abundance of specific levels of class by *mp_extract_abundance* 2.5.4.

```
library(ggplot2)
library(MicrobiotaProcess)
# The relative abundance of all taxonomy for samples will be calculated
mpse2 %<>% mp_cal_abundance(.abundance = RareAbundance)
# The relative abundance of all taxonomy for group will be calculated
mpse2 %<>% mp_cal_abundance(.abundance = RareAbundance, .group = Group)
# The 30 most abundant taxonomy will be visualized.
pclass <- mpse2 %>%
          mp_plot_abundance(
             .abundance = RareAbundance,
             .group = Group,
             taxa.class = Class,
             topn = 30
          ) +
          xlab(NULL) +
          ylab("relative abundance (%)") +
          theme(
             legend.key.width = unit(0.3, "cm"),
             legend.key.height = unit(0.3, "cm")
          ) +
          xlab(NULL) +
          ylab("relative abundance (%)") +
```

```
          theme(
             legend.key.width = unit(0.3, "cm"),
             legend.key.height = unit(0.3, "cm"),
             legend.text = element_text(size=6)
          )
pclass
```

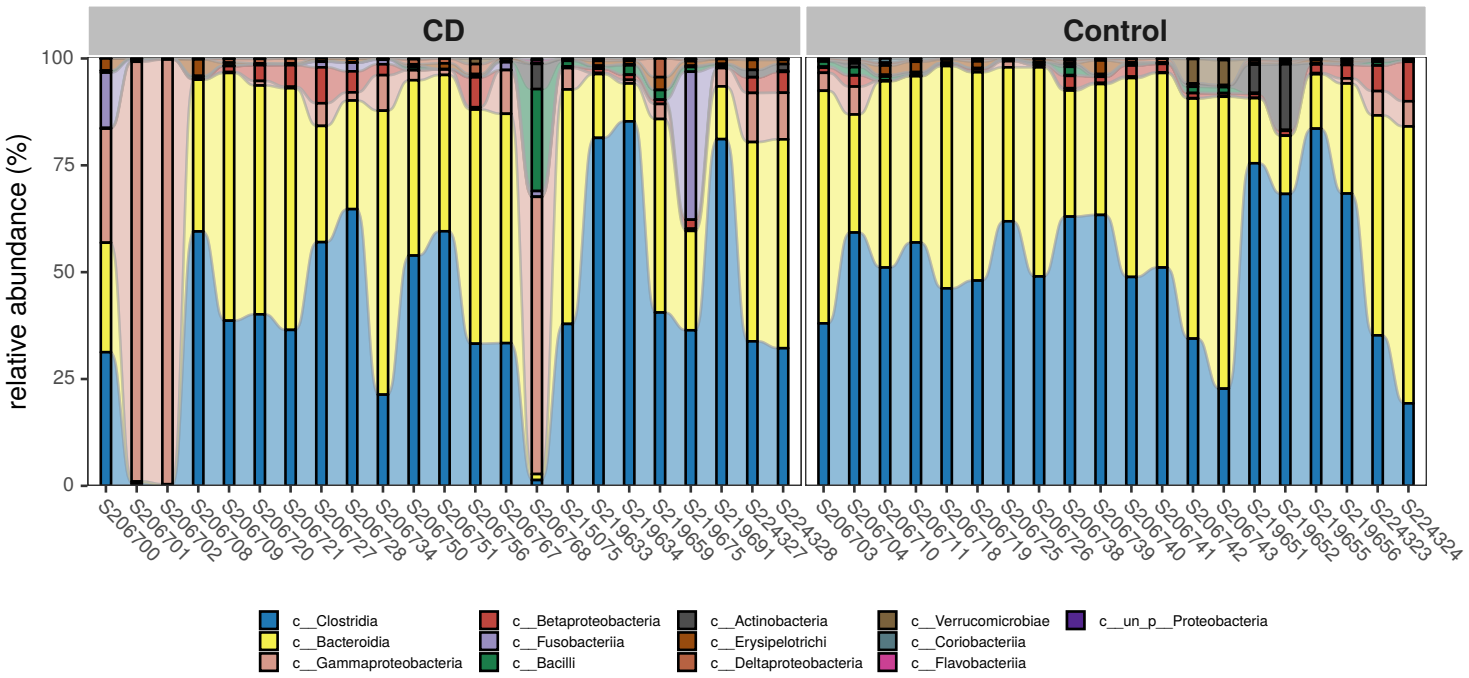

Fig. S3: **The relative abundance of each sample in *class* level**

The relative abundance of groups also can be visualized by providing *.group* argument and setting *plot.group=TRUE* in the *mp_plot_abundance*. If you want to view the raw abundance (count or others) of taxa, you can set the *relative* parameter of *mp_plot_abundance* to *FALSE*.

```
# Show the abundance in different groups.
fclass <- mpse2 %>%
          mp_plot_abundance(
             .abundance = RareAbundance,
             .group = Group,
             taxa.class = Class,
             topn = 30,
             plot.group = TRUE
          ) +
          xlab(NULL) +
          ylab("relative abundance (%)") +
          theme(legend.position = "none")

pclass2 <- mpse2 %>%
          mp_plot_abundance(
             .abundance = RareAbundance,
             .group = Group,
             relative = FALSE,
             taxa.class = Class,
             topn = 30
          ) +
          xlab(NULL) +
          ylab("count reads") +
          theme(legend.key.width = unit(0.3, "cm"),
                legend.key.height = unit(0.3, "cm"),
```

```
            legend.text = element_text(size=6)
        )

aplot::plot_list(pclass2, fclass, widths=c(10, 1), tag_levels = "A")
```

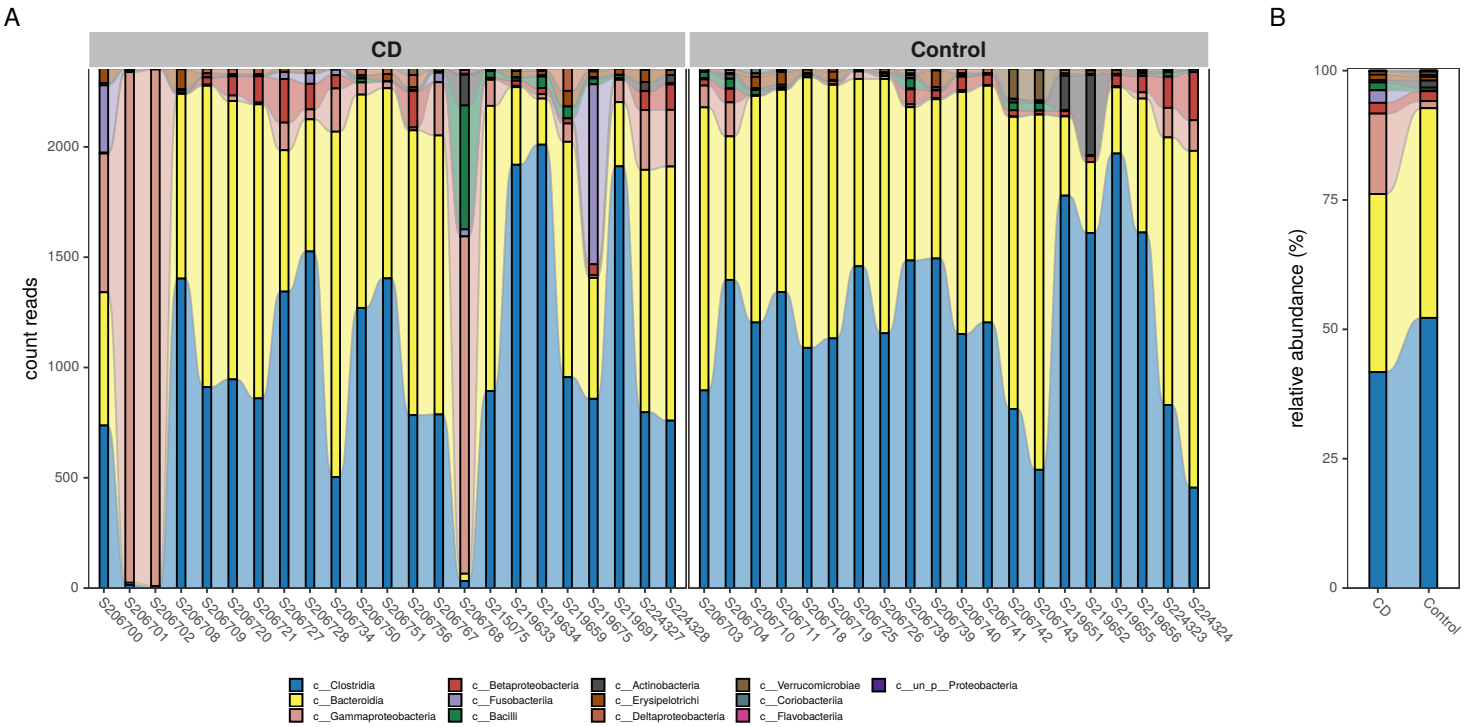

Fig. S4: **This example show how to displayed the abundance (count or other) of sample and the relative abundance of groups**. The Abundance (count by rarefied) of each sample (A) and the relative abundance of group (B), these results show the *Gammaproteobacteria* of *CD* group might be more abundant than the *control* group.

The abundance of features also can be visualized by `mp_plot_abundance` with heatmap plot by setting `geom="heatmap"`.

```
hclass1 <- mpse2 %>%
          mp_plot_abundance(
             .abundance = RareAbundance,
             .group = Group,
             taxa.class = Class,
             topn = 30,
             geom = "heatmap"
          ) %>%
          set_scale_theme(
            x = list(scale_fill_viridis_c(option = "H"),
                     theme(
                       axis.text.x = element_text(size = 6),
                       axis.text.y = element_text(size = 7),
                       legend.title = element_text(size = 7),
                       legend.text = element_text(size = 5),
                       legend.key.width = unit(0.3, "cm"),
                       legend.key.height = unit(0.3, "cm")
                     )
                ),
            aes_var = RelRareAbundance
          ) %>%
          set_scale_theme(
            x = list(scale_fill_manual(values = cols),
                     theme(
                       legend.key.height = unit(0.3, "cm"),
                       legend.key.width = unit(0.3, "cm"),
```

```
                      legend.spacing.y = unit(0.02, "cm"),
                      legend.text = element_text(size = 7),
                      legend.title = element_text(size = 9)
                    )
                ),
            aes_var = Group
          )

hclass2 <- mpse2 %>%
          mp_plot_abundance(
             .abundance = RareAbundance,
             .group = Group,
             taxa.class = Class,
             topn = 30,
             geom = 'heatmap',
             relative = FALSE
          ) %>%
          set_scale_theme(
            x = list(scale_fill_viridis_c(option = "H"),
                     theme(
                       axis.text.x = element_text(size = 6),
                       axis.text.y = element_text(size = 7),
                       legend.title = element_text(size = 7),
                       legend.text = element_text(size = 5),
                       legend.key.width = unit(0.3, "cm"),
                       legend.key.height = unit(0.3, "cm")
                     )
                ),
            aes_var = RareAbundance
          ) %>%
          set_scale_theme(
            x = list(scale_fill_manual(values = cols),
                     theme(
                       legend.key.height = unit(0.3, "cm"),
                       legend.key.width = unit(0.3, "cm"),
                       legend.spacing.y = unit(0.02, "cm"),
                       legend.text = element_text(size = 7),
                       legend.title = element_text(size = 9)
                     )
                ),
            aes_var = Group
          )

p <- aplot::plot_list(hclass1, hclass2, nrow = 1, tag_levels = "A")
p
```

2.4.2 Venn or Upset plot

The Venn or UpSet plot can help us to obtain the difference between groups in overview. MicrobiotaProcess provides `mp_cal_venn` (`mp_plot_venn`) and `mp_cal_upset` (`mp_plot_upset`) to perform the Venn and Upset analysis.

```
mpse2 %<>%
    mp_cal_venn(
      .abundance = RareAbundance,
      .group = Group
    )

venn_p <- mpse2 %>%
    mp_plot_venn(
      .group = Group,
```

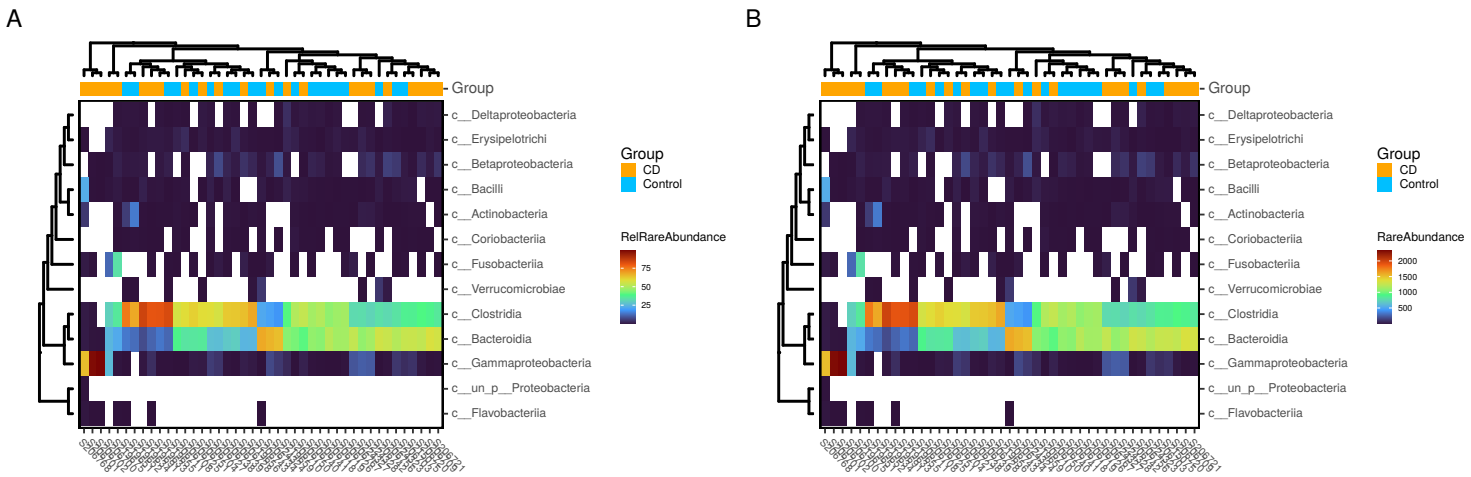

Fig. S5: **The heatmap of abundance for each sample in *class* level.** The color (continuous) of heatmap represents the abundance of taxon, the color of bar represents the group name of sample, the horizontal coordinate represents the sample, and the vertical coordinate represents the taxon.

```
      set_size = 2.5,
      label_size = 2,
      edge_size = 2.5
    ) +
    scale_colour_manual(values = cols) +
    scale_fill_viridis_c(guide = guide_colorbar(barwidth=.3, barheight=2)) +
    theme(
      legend.title = element_text(size = 8),
      legend.text = element_text(size = 6)
    )

mpse2 %<>%
    mp_cal_upset(
      .abundance = RareAbundance,
      .group = Group
    )

upset_p <- mpse2 %>%
    mp_plot_upset(
      .group = Group
    ) +
    theme_bw() +
    theme(
      plot.background = element_blank(),
      panel.border = element_blank(),
      panel.grid = element_blank(),
      axis.line.x.bottom = element_line(size = .5),
      axis.line.y.left = element_line(size = .5)
    ) +
    ggupset::theme_combmatrix(
      combmatrix.label.extra_spacing = 40
    )

library(ggpp)
p.up.venn <- upset_p +
             ggpp::annotate(
               "plot_npc",
               npcx = "right",
               npcy = "top",
               label = venn_p,
```

```
                vp.width = 0.6,
                vp.height = 0.4
            )
p.up.venn
```

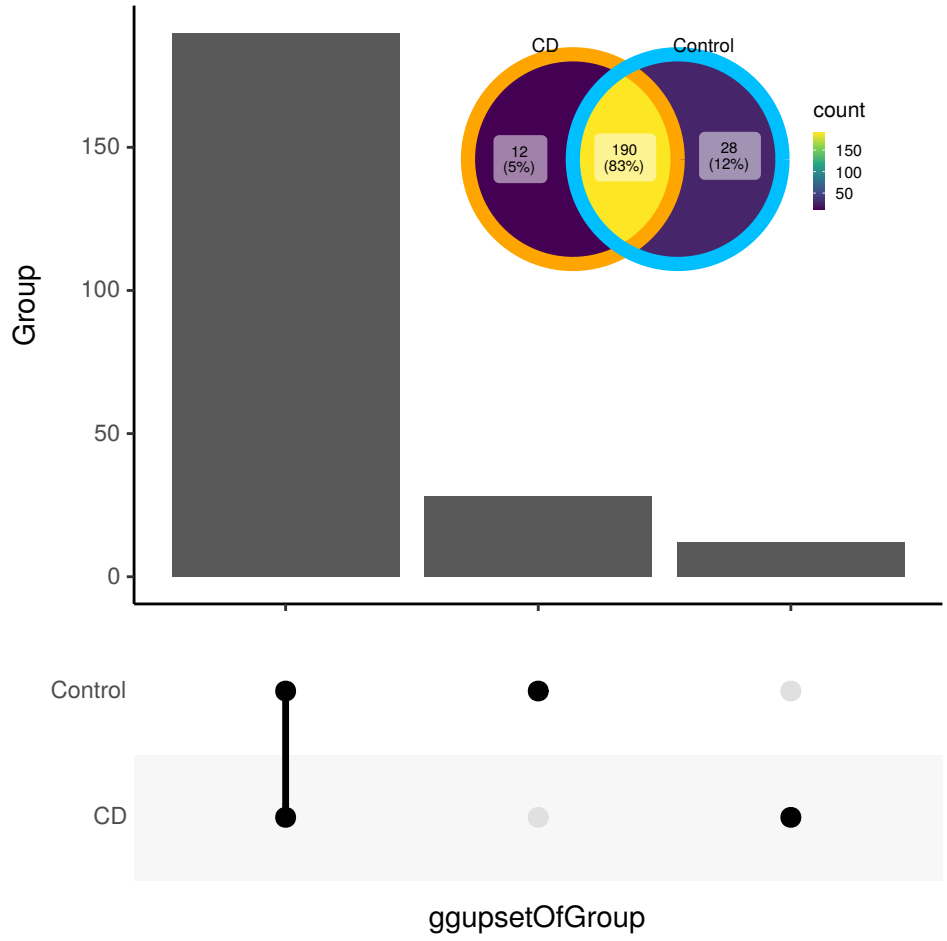

Fig. S6: **The venn diagram and upset plot for groups in OTU/ASV level**

2.5 beta analysis

2.5.1 PCA analysis

`PCA` (Principal component analysis) and `PCoA` (Principal Coordinate Analysis) are general statistical procedures to compare dissimilarity of samples. And `PCoA` can based on the phylogenetic or count-based distance metrics, such as `Bray-Curtis`, `Jaccard`, `Unweighted-UniFrac` and `weighted-UniFrac`. `MicrobiotaProcess` presents the `mp_cal_dist`, `mp_cal_pca`, `mp_cal_pcoa`, `mp_cal_dca`, `mp_cal_nmds`, `mp_cal_cca`, `mp_cal_rda`, `mp_adonis`, `mp_anosim`, `mp_mrpp`, `mp_envfit` and `mp_mantel` for the analysis.

```
library(MicrobiotaProcess)
library(patchwork)
# hellinger transform
mpse2 %<>%
    mp_decostand(
        .abundance = Abundance,
        method = "hellinger"
    )

mpse2 %<>% mp_cal_pca(.abundance = hellinger)
# Visulizing the result
pcaplot1 <- mpse2 %>%
```

```
            mp_plot_ord(
              .ord = pca,
              .group = Group,
              .starshape = Group,
              .size = Observe
            ) +
            scale_fill_manual(values = cols) +
            scale_size_continuous(
              range = c(1, 3),
              guide = guide_legend(override.aes = list(starshape = 15))
            ) +
            theme(
              legend.key.width = unit(0.3, "cm"),
              legend.key.height = unit(0.3, "cm"),
              legend.text = element_text(size = 6),
              legend.title = element_text(size = 7)
            )
# .dim = c(1, 3) to show the first and third principal components.
pcaplot2 <- mpse2 %>%
            mp_plot_ord(
              .ord = pca,
              .dim = c(1, 3),
              .group = Group,
              .starshape = Group,
              .size = Observe
            ) +
            scale_fill_manual(values = cols) +
            scale_size_continuous(
              range = c(1, 3),
              guide = guide_legend(override.aes = list(starshape = 15))
            ) +
            theme(
              legend.key.width = unit(0.3, "cm"),
              legend.key.height = unit(0.3, "cm"),
              legend.text = element_text(size = 6),
              legend.title = element_text(size = 7)
            )

(pcaplot1 | pcaplot2) + plot_annotation(tag_levels = "A")
```

2.5.2 PCoA analysis

```
# distmethod
# "unifrac",  "wunifrac", "manhattan", "euclidean", "canberra", "bray", "kulczynski" ...(vegdist, dist)
mpse2 %<>%
    mp_cal_dist(
      .abundance = hellinger,
      distmethod = "bray"
    )

# PCoA analysis
mpse2 %<>%
    mp_cal_pcoa(
      .abundance = hellinger,
      distmethod = "bray"
    )
pcoaplot1 <- mpse2 %>%
             mp_plot_ord(
               .ord = pcoa,
```

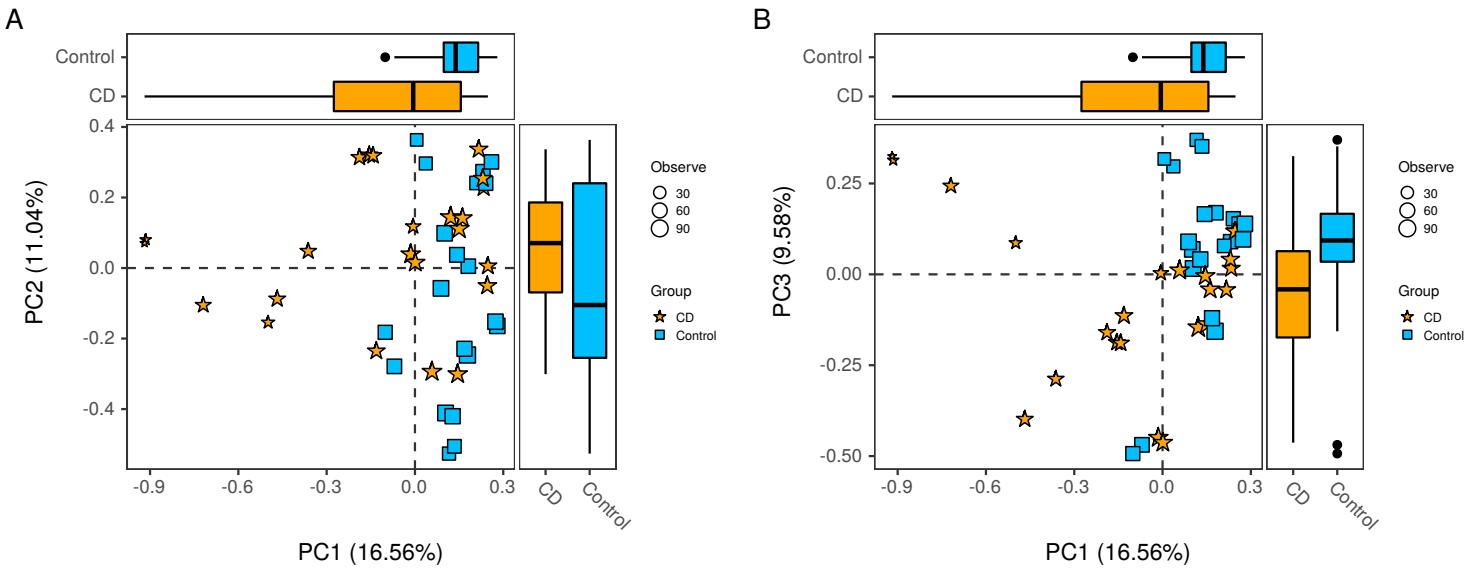

Fig. S7: **The PCA plot of the community**. Each point represents one sample, the size of point represents the observe OTU of the sample. The color of point represents the group name of the sample, based on the first and second component (A), based on the first and third component (B).

```
            .group = Group,
            .starshape = Group,
            .color = Group,
            .size = Observe,
            ellipse = TRUE,
            show.legend = FALSE
          ) +
          scale_color_manual(
            values = cols
          ) +
          scale_fill_manual(values = cols) +
          scale_size_continuous(
            range = c(1, 3),
            guide = guide_legend(override.aes = list(starshape = 15))
          ) +
          theme(
            legend.key.width = unit(0.3, "cm"),
            legend.key.height = unit(0.3, "cm"),
            legend.text = element_text(size=6),
            legend.title = element_text(size=7)
          )
# first and third principal co-ordinates
pcoaplot2 <- mpse2 %>%
           mp_plot_ord(
             .ord = pcoa,
             .group = Group,
             .starshape = Group,
             .color = Group,
             .size = Observe,
             ellipse = TRUE,
             .dim = c(1, 3),
             show.legend = FALSE
           ) +
           scale_color_manual(
             values = cols
           ) +
           scale_fill_manual(
```

```
              values = cols
            ) +
            scale_size_continuous(
              range = c(1, 3),
              guide = guide_legend(override.aes = list(starshape = 15))
            ) +
            theme(
              legend.key.width = unit(0.3, "cm"),
              legend.key.height = unit(0.3, "cm"),
              legend.text = element_text(size = 6),
              legend.title = element_text(size = 7)
            )
(pcoaplot1 | pcoaplot2) + plot_annotation(tag_levels = "A")
```

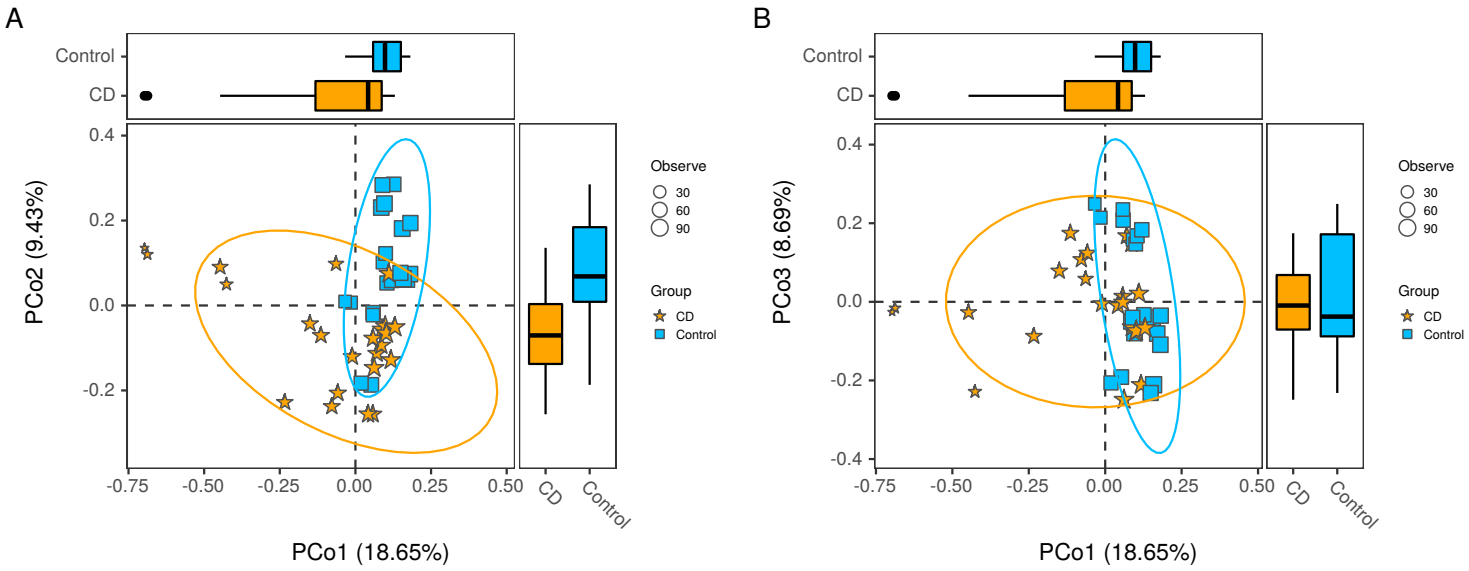

Fig. S8: **The PCoA plot based on Bray-Curtis distance.**

The result of distance between the samples also can be visualized by `mp_plot_dist` with heatmap or boxplot.

```
pdist1 <- mpse2 %>%
          mp_plot_dist(
            .distmethod = bray,
            .group = Group
          ) %>%
          set_scale_theme(
            x = scale_fill_manual(
                  values=cols,
                  guide = guide_legend(
                            keywidth = 0.5,
                            keyheight = 0.5,
                            label.theme=element_text(size=6)
                          )
                ),
            aes_var = Group
          ) %>%
          set_scale_theme(
            x = list(scale_size_continuous(range = c(1, 3)),
                     scale_color_viridis_c(option = "H"),
                     theme(
                       legend.key.width = unit(0.3, "cm"),
                       legend.text = element_text(size = 6),
                       legend.title = element_text(size = 7)
                     )
```

```
                ),
            aes_var = bray
          )

pdist2 <- mpse2 %>%
          mp_plot_dist(
            .distmethod = bray,
            .group = Group,
            group.test = TRUE
          ) +
          scale_color_manual(
            values = c("orange", "#00A08A", "deepskyblue")
          ) +
          scale_fill_manual(
            values = c("orange", "#00A08A", "deepskyblue")
          )
aplot::plot_list(pdist1, pdist2, widths = c(3, 1), nrow=1, tag_levels = "A")
```

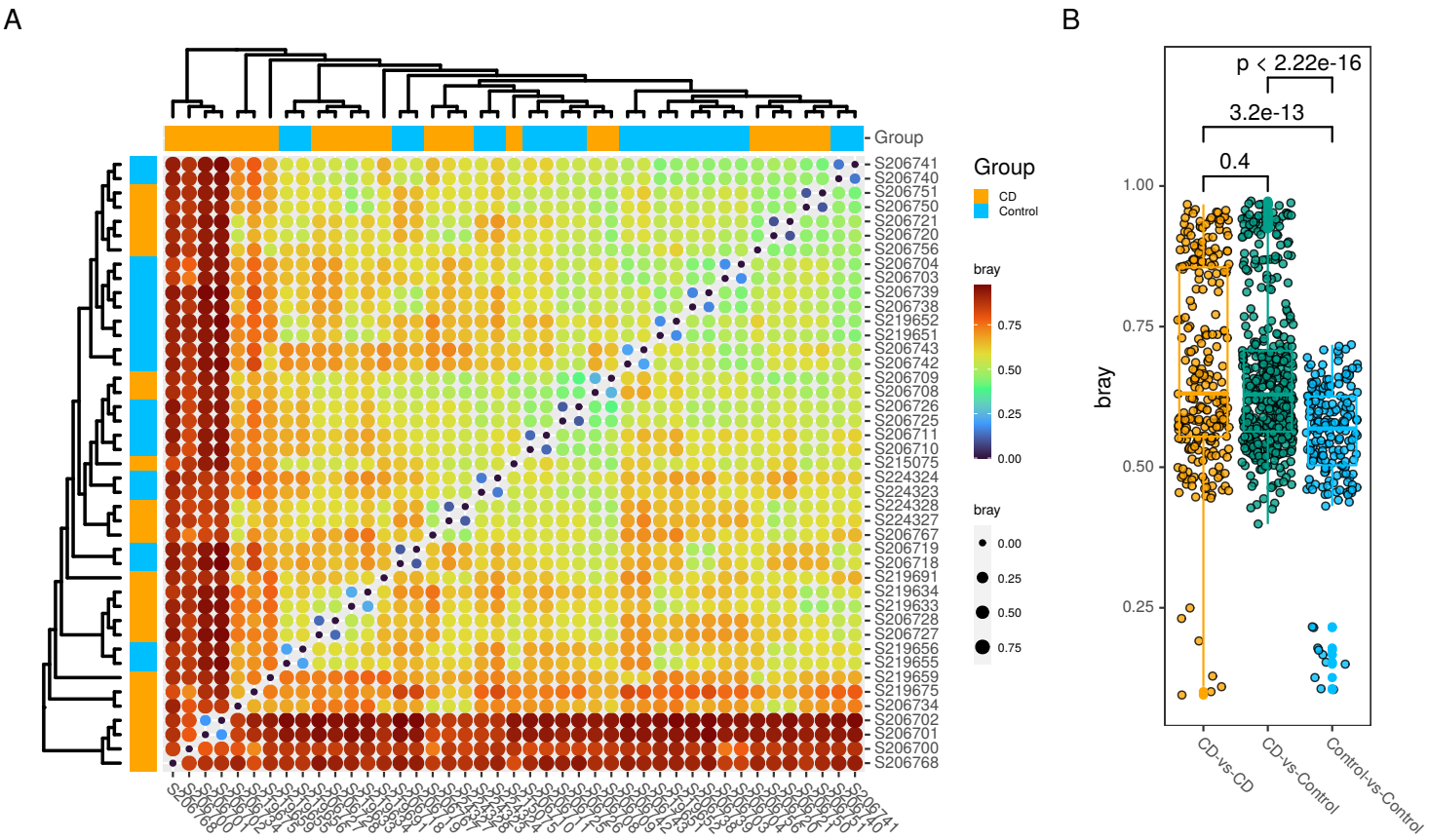

Fig. S9: **The distance heatmap and the boxplot for each sample**. The size and color of the heatmap represent the distance of each sample, the color of bar represent the group of sample (A). The boxplot represent the distance pairs of sample among the group, it show the dissimilarity of sample between the *control* and *CD* is significant, which is consistent with the result of Permutational Multivariate Analysis of Variance 2.5.3.

2.5.3 Permutational Multivariate Analysis of Variance

We also can perform the Permutational Multivariate Analysis of Variance using `mp_adonis` wrapping the `adonis` of `vegan` (Oksanen et al. 2020).

```
mpse2 %<>% mp_adonis(
            .abundance = hellinger,
            distmethod = "bray",
            .formula = ~Group,
            permutation = 9999,
            action = "add")
mpse2 %>%
    mp_extract_internal_attr(name=adonis) %>%
    mp_fortify()
```

```
## # A tibble: 3 x 7
##   factors      Df SumsOfSqs MeanSqs F.Model     R2 `Pr(>F)`
##   <chr>     <dbl>     <dbl>   <dbl>   <dbl>  <dbl>    <dbl>
## 1 Group         1     0.789   0.789    3.88 0.0864   0.0001
## 2 Residuals    41     8.34    0.203   NA    0.914   NA
## 3 Total        42     9.12   NA       NA    1       NA
```

From the result, we found the `pvalue` of the analysis of `adonis` is smaller than 0.05 for the `Group`, meaning the dissimilarity of samples between the `Group` is significant, which is consistent with the 2.5.2.

2.5.4 hierarchical cluster analysis of samples

`beta diversity` metrics can assess the differences between microbial communities. It can be visualized with `PCA` or `PCoA`, this can also be visualized with hierarchical clustering based on ggplot2 (Wickham 2011), ggtree (Yu et al. 2017) and ggtreeExtra (Xu et al. 2021)

```
library(ggplot2)
library(MicrobiotaProcess)
library(ggtree)
library(ggtreeExtra)
mpse2 %<>%
    mp_cal_clust(
      .abundance = hellinger,
      distmethod = "bray",
      action = "add"
    )
hcsample <- mpse2 %>%
            mp_extract_internal_attr(name=SampleClust)
# rectangular layout + relative abundance of phyla
phy.tb <- mpse2 %>%
          mp_extract_abundance(
            taxa.class = Phylum,
            topn = 30
          ) %>%
          tidyr::unnest(cols=RareAbundanceBySample) %>%
          dplyr::rename(Phyla="label")

cplot1 <- ggtree(
            hcsample,
            layout = "rectangular"
          ) +
          geom_treescale(fontsize = 2) +
          geom_tippoint(mapping=aes(color=Group)) +
          geom_fruit(
            data = phy.tb,
            geom = geom_col,
```

```
            mapping = aes(x = RelRareAbundanceBySample, y = Sample, fill = Phyla),
            orientation = "y",
            offset = 0.08,
            pwidth = 3,
            width = .6,
            axis.params = list(
              axis = "x",
              title = "The relative abundance of phyla (%)",
              title.size = 3,
              title.height = 0.04,
              text.size = 2,
              vjust = 1
            )
          ) +
          geom_tiplab(as_ylab = TRUE) +
          scale_color_manual(
            values = cols,
            guide = guide_legend(
              keywidth = .5,
              keyheight = .5,
              title.theme = element_text(size = 8),
              label.theme = element_text(size = 6)
            )
          ) +
          scale_fill_manual(
            values=c(colorRampPalette(RColorBrewer::brewer.pal(12,"Set2"))(6)),
            guide = guide_legend(
              keywidth = .5,
              keyheight = .5,
              title.theme = element_text(size = 8),
              label.theme = element_text(size = 6)
            )
          ) +
          scale_x_continuous(expand = c(0, 0.01))
cplot1
```

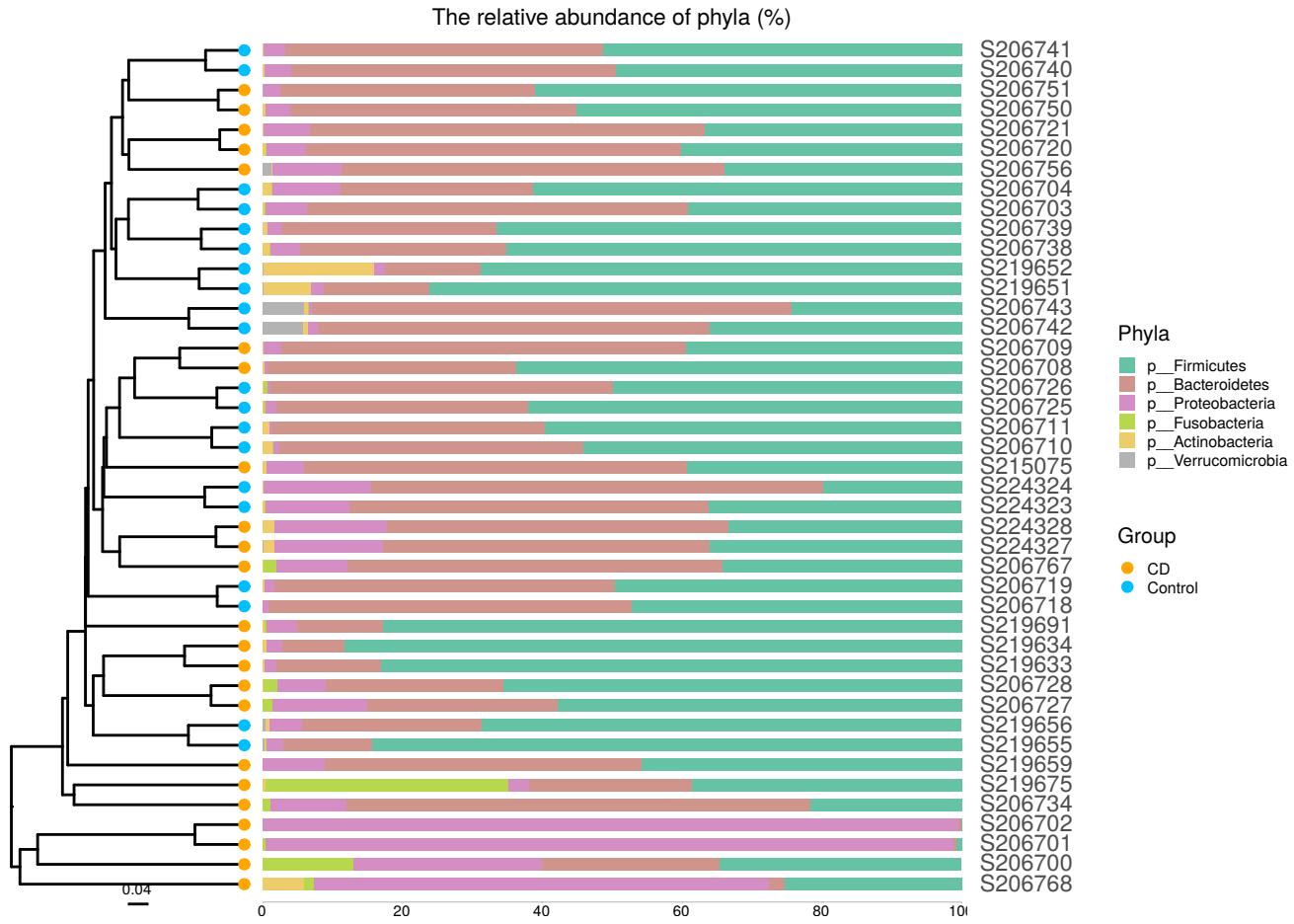

Fig. S10: **The hierarchical clustering plot of samples based on Bray-Curtis distance calculated with abundance of OTU/ASV and the relative Abundance of phyla for samples**

2.6 biomarker discovery

This package provides `mp_diff_analysis` to detect the biomarker. And the result (with `action = "get"`) can be visualized by `ggdiffbox`, `ggdiffclade`, `ggeffectsize`, `ggdifftaxbar` and `mp_plot_diff_res`, or displayed manually using `ggtree` (Yu et al. 2017) and `ggtreeExtra` (Xu et al. 2021).

```
# for the kruskal_test and wilcox_test
library(coin)
library(MicrobiotaProcess)

# get result (diffAnalysisClass) of the different analysis with action = 'get'.
deres <- mpse2 %>%
         mp_diff_analysis(
            .abundance = RareAundance,
            .group = Group,
            first.test.method = "kruskal_test",
            filter.p = "pvalue",
            first.test.alpha = 0.05,
            strict = TRUE,
            second.test.method = "wilcox_test",
            second.test.alpha = 0.05,
            subcl.min = 3,
            subcl.test = TRUE,
            ml.method = "lda",
            ldascore = 3,
            action = "get"
         )
```

```
mpse2 <- mpse2 %>%
         mp_diff_analysis(
            .abundance = RareAundance,
            .group = Group,
            first.test.method = "kruskal_test",
            filter.p = "pvalue",
            first.test.alpha = 0.05,
            strict = TRUE,
            second.test.method = "wilcox_test",
            second.test.alpha = 0.05,
            subcl.min = 3,
            subcl.test = TRUE,
            ml.method = "lda",
            ldascore = 3,
            action = "add"
         )
```

2.6.1 visualization of different results by ggdiffclade

The color of discriminative taxa represent the taxa is more abundant in the corresponding group. The point size shows the negative logarithms (base 10) of pvalue. The bigger size of point shows more significant (lower pvalue), the *pvalue* was calculated in the first step test (default is *kruskal.test*).

```
diffclade_p <- ggdiffclade(
                    obj=deres,
                    alpha=0.3,
                    linewd=0.15,
                    skpointsize=0.6,
                    layout="radial",
                    taxlevel=3,
                    removeUnkown = TRUE,
                    reduce = FALSE # This argument is to remove the branch of unknown taxonomy.
                ) +
                scale_fill_manual(
                    values = cols
                ) +
                guides(color = guide_legend(
                                    keywidth = 0.1,
                                    keyheight = 0.2,
                                    order = 3,
                                    ncol=1)
                ) +
                theme(
                    panel.background = element_rect(fill=NA),
                    legend.position = "right",
                    plot.margin = margin(0,0,0,0),
                    legend.key.width = unit(0.2, "cm"),
                    legend.key.height = unit(0.2, "cm"),
                    legend.spacing.y = unit(0.02, "cm"),
                    legend.title = element_text(size=7),
                    legend.text = element_text(size=6),
                    legend.box.spacing = unit(0.02,"cm")
                )
diffclade_p
```

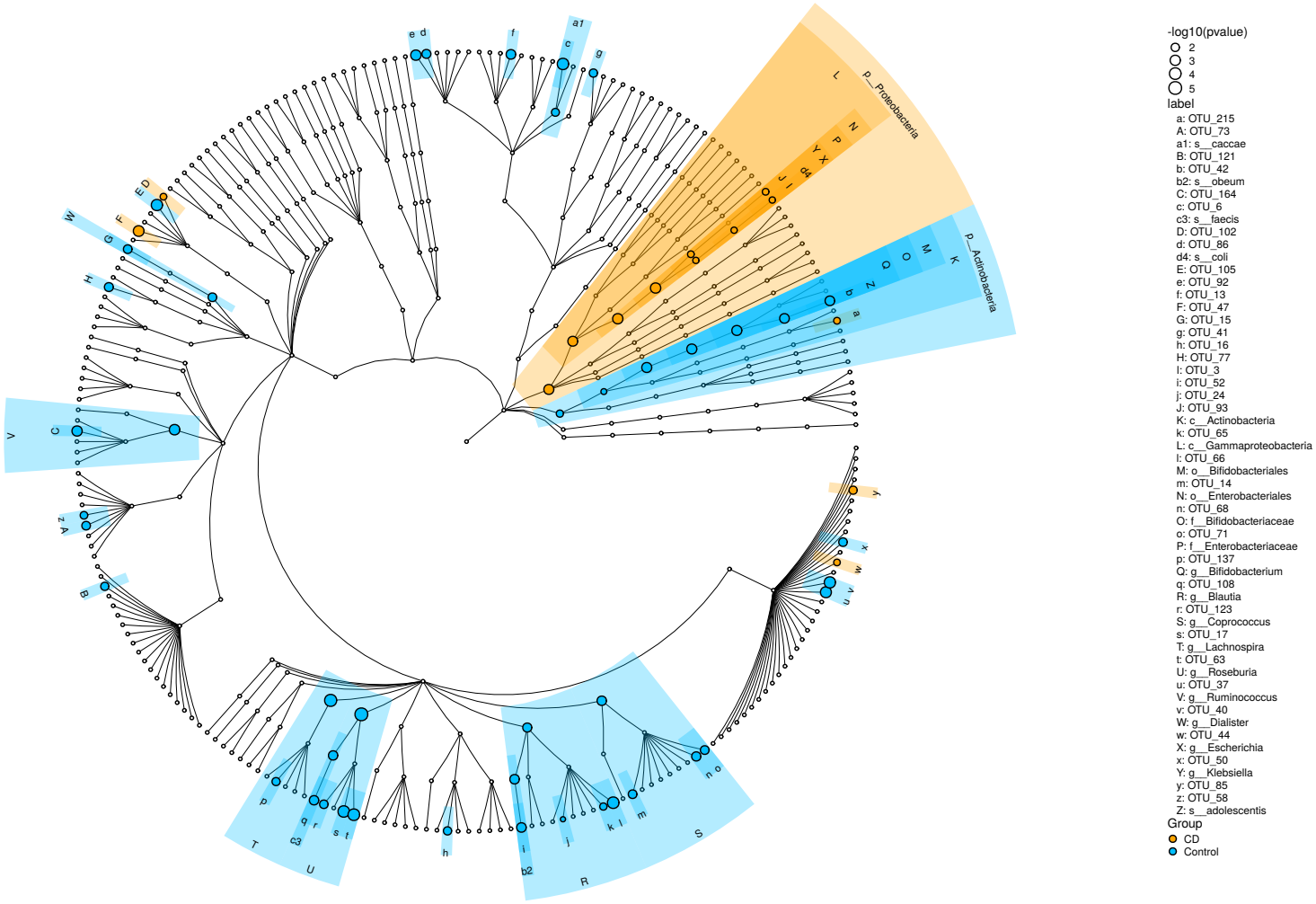

Fig. S11: **The taxa tree clade plot of different analysis result**.

We also can visualized the result with `ggtree` (Yu et al. 2017) and `ggtreeExtra` (Xu et al. 2021).

```
taxa.tree <- mpse2 %>% mp_extract_tree(type='taxatree')
p1 <- ggtree(
        taxa.tree,
        layout="radial",
        size = 0.3
      ) +
      geom_point(
        data = td_filter(!isTip),
        fill="white",
        size=1,
        shape=21
      )
# display the high light of phylum clade.
p2 <- p1 +
      geom_hilight(
        data = td_filter(nodeClass == "Phylum"),
        mapping = aes(node = node, fill = label)
      )
# display the relative abundance of features(OTU)
p3 <- p2 +
      ggnewscale::new_scale("fill") +
      geom_fruit(
         data = td_unnest(RareAbundanceBySample),
         geom = geom_star,
```

```
        mapping = aes(
                      x = fct_reorder(Sample, Group, .fun=min),
                      size = RelRareAbundanceBySample,
                      fill = Group,
                      subset = RelRareAbundanceBySample > 0
                    ),
        starshape = 13,
        starstroke = 0.25,
        offset = 0.04,
        pwidth = 1.5,
        grid.params = list(vline = TRUE, size = 0.01, linetype = 1)
      ) +
      scale_size_continuous(
        name="Relative Abundance (%)",
        range = c(0.5, 3)
      ) +
      scale_fill_manual(values=cols)
# display the tip labels of taxa tree
p4 <- p3 + geom_tiplab(size=2, offset=12.8)
# display the LDA of significant OTU.
p5 <- p4 +
      ggnewscale::new_scale("fill") +
      geom_fruit(
        geom = geom_col,
        mapping = aes(
                      x = LDAmean,
                      fill = Sign_Group,
                      subset = !is.na(LDAmean)
                      ),
        orientation = "y",
        offset = 0.5,
        pwidth = 1,
        axis.params = list(axis = "x",
                           title = "Log10(LDA)",
                           title.height = 0.005,
                           title.size = 2,
                           text.size = 1.8,
                           vjust = 1),
        grid.params = list(linetype = 3)
      )

# display the significant (FDR) taxonomy after kruskal.test (default)
p6 <- p5 +
      ggnewscale::new_scale("size") +
      geom_point(
        data=td_filter(!is.na(fdr)),
        mapping = aes(size = -log10(fdr),
                      fill = Sign_Group,
                      ),
        shape = 21,
      ) +
      scale_size_continuous(range=c(1, 3)) +
      scale_fill_manual(values=cols)

p6 <- p6 + theme(
            legend.key.height = unit(0.3, "cm"),
            legend.key.width = unit(0.3, "cm"),
            legend.spacing.y = unit(0.02, "cm"),
            legend.text = element_text(size = 7),
            legend.title = element_text(size = 9),
```

```
            )
p6
```

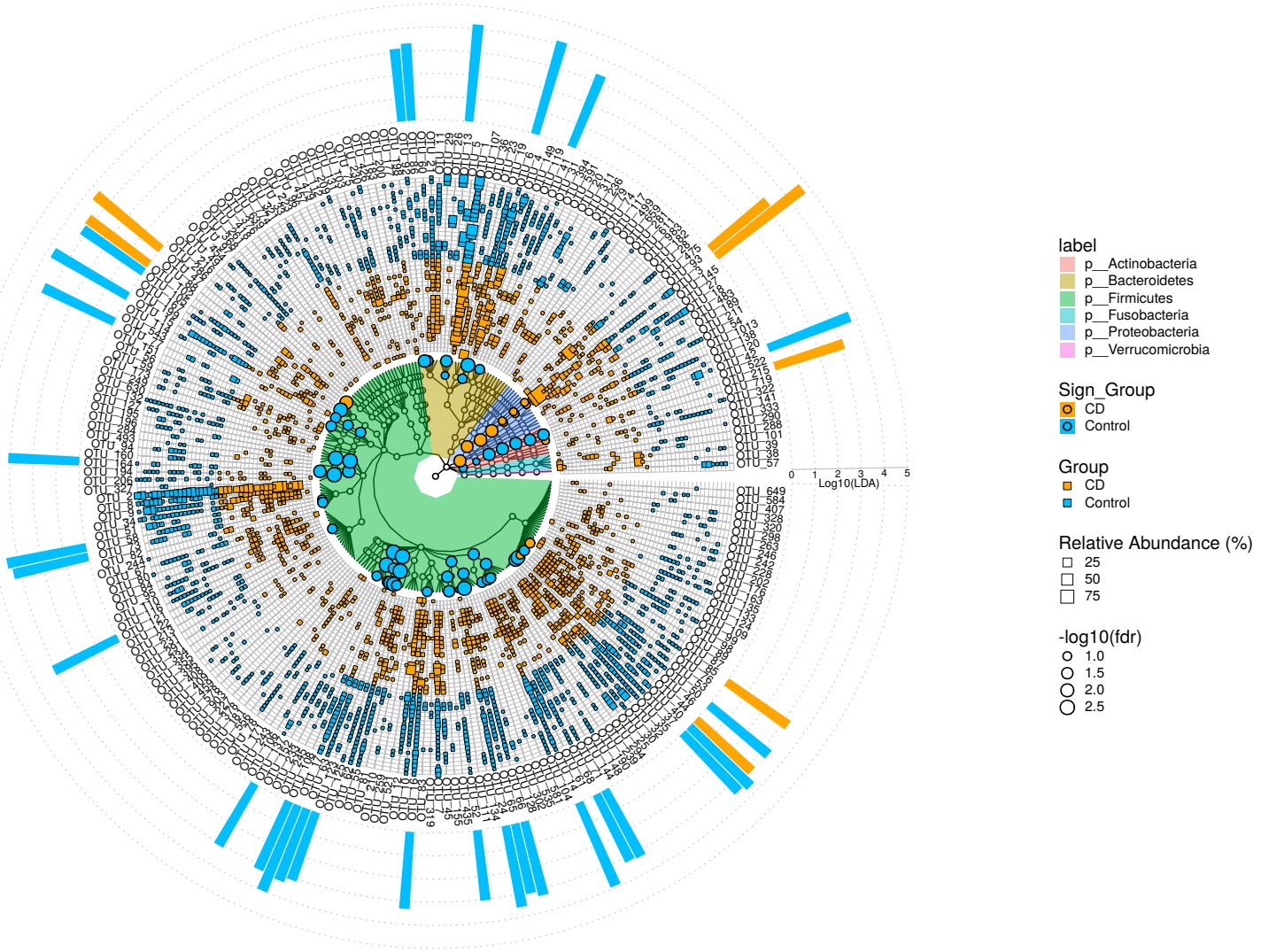

Fig. S12: **The taxa tree of the community with the relative abundance of each OTU/ASV on sample and the LDA of different OTU/ASV**. The taxa tree is built with the taxa of all samples. The high light color of taxa tree represents the phyla of the clade. The external point layer represents the relative abundance of each OTU on sample. The external bar layer represents the LDA of the different OTU. The colored points represent the different taxa, the size represents the *pvalue* or *fdr*.

To decreases coding burden, we also developed *mp_plot_diff_res* to visualize the result of different analysis.

```
library(ggplot2)
pp <- mpse2 %>%
    mp_plot_diff_res() +
    scale_fill_manual(
      values = cols
    ) +
    scale_fill_manual(
      aesthetics = "fill_new",
      values = cols
    )
pp
```

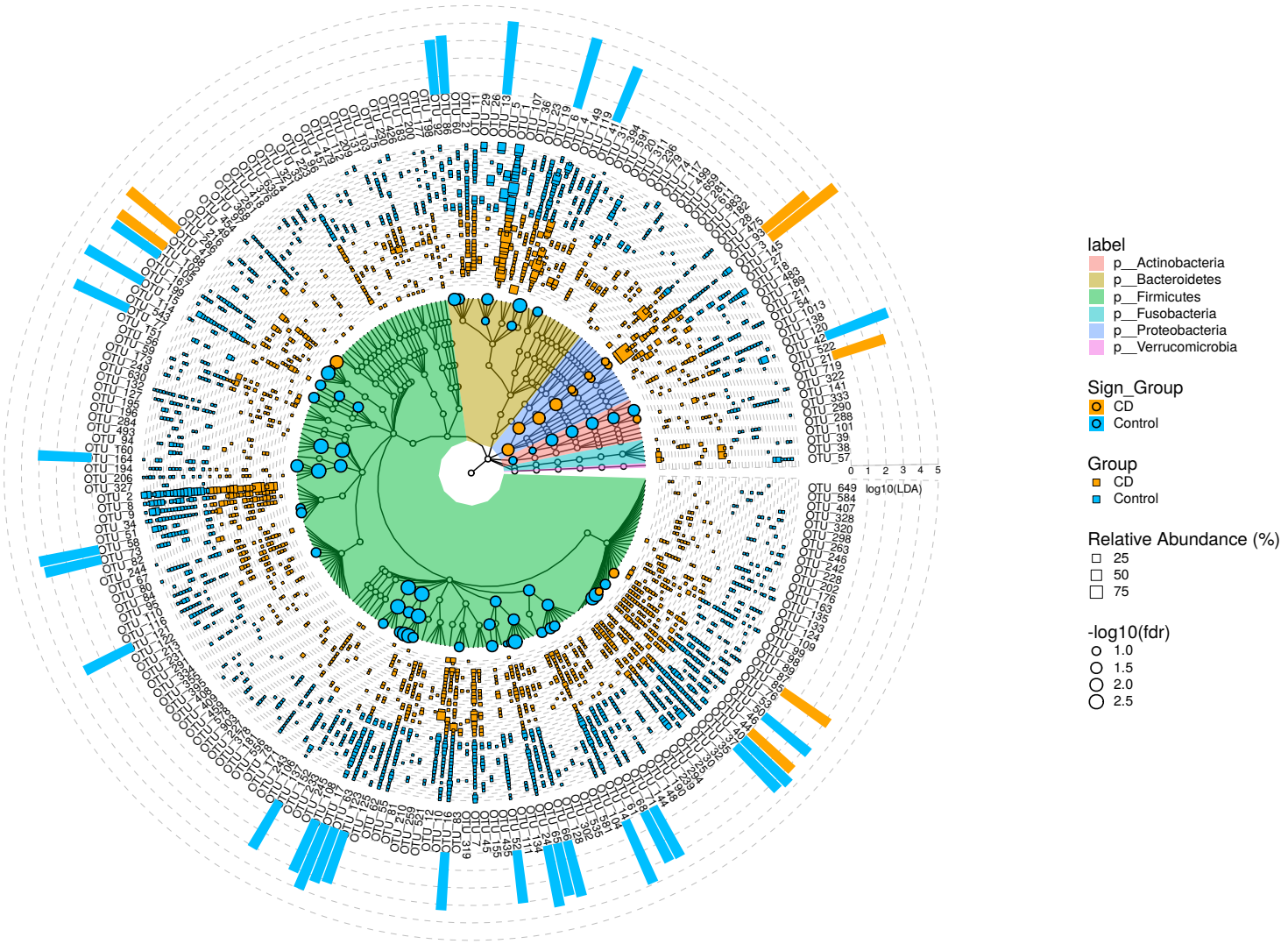

Fig. S13: seel also Fig. S12

2.6.2 visualization of different results by ggdiffbox

The left panel represents the relative abundance or abundance (according the standard_method) of biomarker, the right panel represents the confident interval of effect size (LDA or MDA) of biomarker. The bigger confident interval shows that the biomarker is more fluctuant, owing to the influence of samples number.

```
diffbox <- ggdiffbox(obj=deres, box_notch=FALSE,
            colorlist=cols, l_xlabtext="relative abundance")
diffbox
```

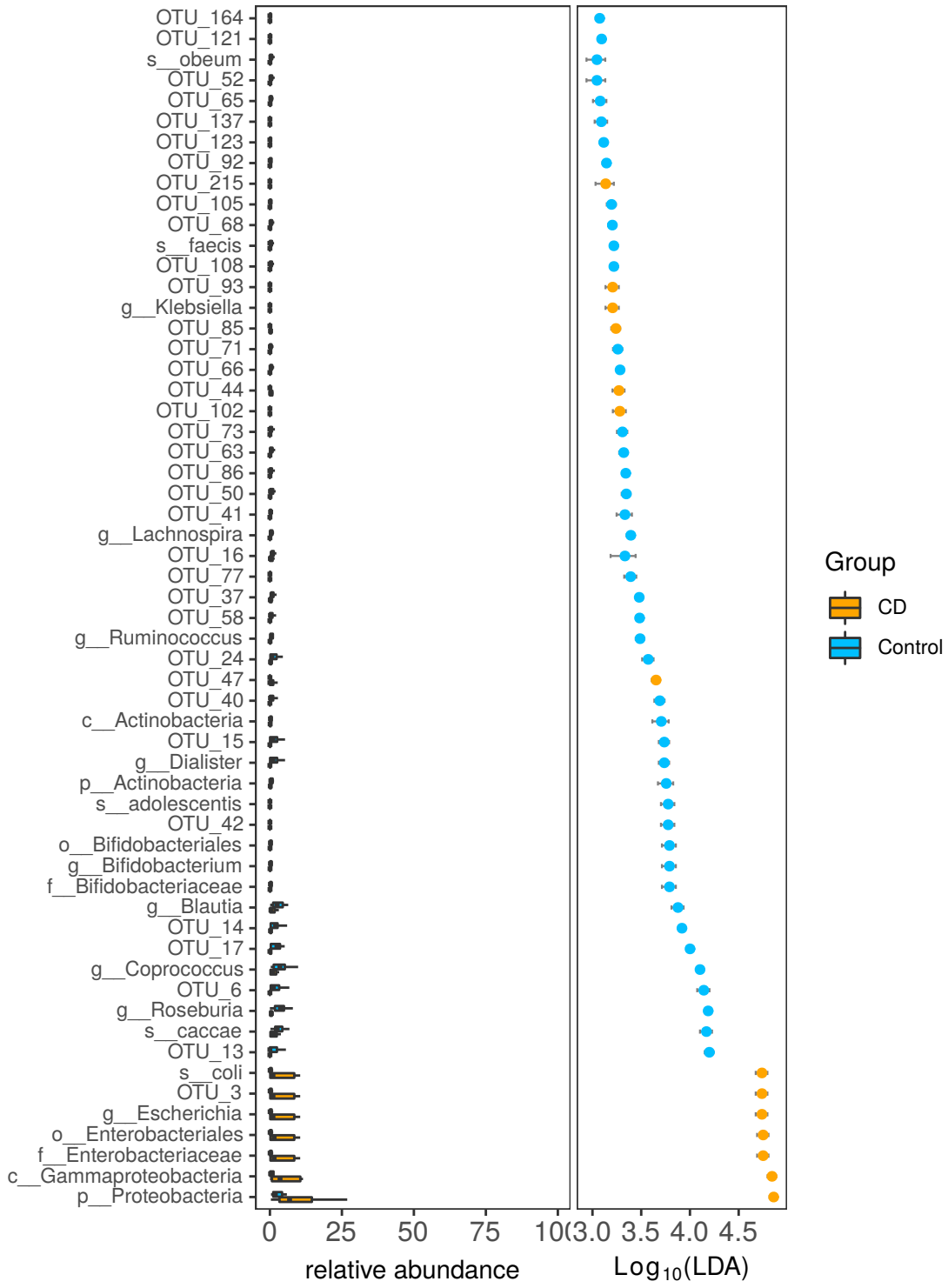

Fig. S14: **The boxplot and the LDA score of different taxa.** The left panel represents the relative abundance of the different taxa, the right panel represents the LDA effect size (95% confidence interval) of different taxa.

2.6.3 visualization of different results by `ggdifftaxbar`

`ggdifftaxbar` can visualize the abundance of biomarker in each samples of groups, the mean and median abundance of groups or subgroups are also showed. `output` parameter is the directory of output.

```
ggdifftaxbar(obj=deres, xtextsize=1.5,
             output="IBD_biomarkder_barplot",
             coloslist=cols)
```

2.7 Interoperable with the existing computing ecosystem

Because the *MPSE* object *MicrobiotaProcess* inherits the *SummarizedExperiment* object (Morgan et al. 2021), The related inherited methods for signature *SummarizedExperiment* can also be applied to the *MPSE*. For example, the *tidybulk* (Mangiola et al. 2021) provides an R tidy framework for modular transcriptomic data analysis. It provides a *test_differential_abundance* to perform differential transcription testing using edgeR quasi-likelihood edgeR likelihood-ratio (LR), limma-voom, limma-voom-with-quality-weights or DESeq2. It can also be compatible with *MPSE*.

```
library(tidybulk)
library(edgeR)
library(aplot)
library(shadowtext)
library(ggrepel)
mpse2 %<>% test_differential_abundance(.abundance = Abundance, .formula = ~Group)
# extract the different OTUs from the MPSE class
res <- mpse2 %>% dplyr::filter(FDR <= .05 & abs(logFC) >= 2)
pp <- res %>%
      mp_plot_abundance(
        .abundance = RareAbundance,
        force = TRUE,
        relative = TRUE,
        feature.dist = "bray",
        geom = "heatmap",
        topn = "all",
        .group = Group
      ) %>%
      set_scale_theme(
        x = list(scale_fill_viridis_c(option = "H"),
                 theme(
                   axis.text.x = element_text(size = 6),
                   axis.text.y = element_text(size = 6),
                   legend.title = element_text(size = 7),
                   legend.text = element_text(size = 5),
                   legend.key.width = unit(0.3, "cm"),
                   legend.key.height = unit(0.3, "cm")
                 )
            ),
        aes_var = RelRareAbundance
      ) %>%
      set_scale_theme(
        x = list(scale_fill_manual(values = cols),
                 theme(
                   legend.key.height = unit(0.3, "cm"),
                   legend.key.width = unit(0.3, "cm"),
                   legend.spacing.y = unit(0.02, "cm"),
                   legend.text = element_text(size = 7),
                   legend.title = element_text(size = 9)
                 )
            ),
        aes_var = Group
      )

f <- res %>%
     mp_extract_taxonomy %>%
     ggplot() +
     geom_text(
       mapping = aes(y=OTU, x=0, label=Genus, color=Phylum),
       hjust = 0,
       size = 2
     ) +
     scale_x_continuous(expand=c(0, 0, 0, 0.1)) +
```

```
    theme_bw() +
    theme(
      legend.text = element_text(size = 5),
      legend.title = element_text(size = 7),
      legend.key.width = unit(0.3, "cm"),
      legend.key.height = unit(0.3, "cm"),
      panel.background = element_blank(),
      panel.grid = element_blank(),
      axis.text = element_blank(),
      axis.ticks = element_blank(),
      panel.border = element_blank()
    ) +
    labs(x = NULL, y = NULL)

pp <- pp %>% insert_right(f, width = 0.2)

sample.tree <- res %>%
     select(-bray) %>%  # remove the bray, Because it was the result of all OTU,
     mp_cal_clust(.abundance = RelRareAbundanceBySample, distmethod = "bray") %>%
     ggtree(layout = igraph::layout_with_kk, color = "#afb7b8") +
     geom_nodepoint(color = "#afb7b8", size = .5) +
     geom_tippoint(aes(fill = Group), shape = 21, size=3) +
     geom_text_repel(
       data = td_filter(isTip),
       mapping = aes(label = label),
       size = 2,
       max.overlaps = 30,
       colour = "black",
       bg.colour = "white"
     ) +
     scale_fill_manual(
       values = cols,
       guide = guide_legend(
          title.theme = element_text(size = 7),
          label.theme = element_text(size = 5),
       )
     )
```

```
## # Note: MPSE object is converted to a tibble data (tbl_mpse object) for independent data analysis.
## #    A new MPSE object can be returned by setting keep.mpse = TRUE.
```

```
p <- mpse2 %>%
     mp_cal_dist(
        .abundance = RelRareAbundanceBySample,
        distmethod = "bray",
        cal.feature.dist = T
     ) %>%
     hclust() %>%
     ggtree(layout = igraph::layout_with_kk, color = "#bed0d1") +
     geom_nodepoint(color = "#bed0d1", size = .5)

# The data.frame contained results of test_differential_abundance
otu.tab <- mpse2 %>% mp_extract_feature()
p <- p %<+% otu.tab +
    geom_tippoint(
      mapping = aes(fill = logFC, size = -log10(FDR)),
      shape = 21,
      color = "grey"
    ) +
    scale_fill_viridis_c(
      option="C",
```

```
      guide = guide_colorbar(
         title.theme = element_text(size = 7),
         label.theme = element_text(size = 5),
         barheight = unit(1.5, "cm"),
         barwidth = unit(.3, "cm")
      )
    ) +
    scale_size_continuous(
      range = c(.5, 6),
      guide = guide_legend(
         key.width = .3,
         key.height = .3,
         label.theme = element_text(size = 5),
         title.theme = element_text(size = 7)
      )
    ) +
    geom_text_repel(
      data = td_filter(FDR <= .05 & abs(logFC) >= 2),
      mapping = aes(x = x, y = y, label = label),
      size = 2,
      min.segment.length = 0.1,
      segment.size = .25,
      segment.colour = 'grey18',
      colour = "black",
      bg.colour = 'white'
      #max.overlaps = 60,
    )

design <- "
  12
  13
  13
"

plot_list(pp, sample.tree, p, design = design, tag_levels = "A")
```

We compared the different result between the *edgeR* (Robinson, McCarthy, and Smyth 2010) and *MicrobiotaProcess*. We found the number of the different OTUs based on edgeR is more than the MicrobiotaProcess. We think this is because we didn't remove the low-abundance OTUs. This operation is generally needed in standard whole-transcriptome workflows. However, if it is preformed in the microbiome analysis, many low-abundance OTUs will be removed. More different OTUs were identified by the operation using edgeR (Robinson, McCarthy, and Smyth 2010).

```
DE.method <- list(
    EdgeR = mpse2 %>%
            mp_extract_feature %>%
            dplyr::filter(FDR<=0.05 & abs(logFC)>=2) %>%
            pull(OTU),
    MP = mpse2 %>%
         mp_extract_feature %>%
         dplyr::filter(pvalue <=0.05) %>%
         pull(OTU)
)
library(ggVennDiagram)
ggVennDiagram(DE.method, edge_size = 3, set_size = 4) +
    scale_color_manual(values=c("pink", "gold")) +
    scale_fill_viridis_c(option="C")
```

Then we extract the same different OTUs, we found the abundance of same OTUs belonged to *Bifidobacterium*, *Faecalibacterium*, *Roseburia* and *Coprobacillus* were significantly decreased in CD group compared to the Control group, the abundance of several OTUs belonged to *Escherichia*, *Klebsiella* and *Haemophilus*, which belonged to Gammaproteobacteria, were significantly enriched in CD group.

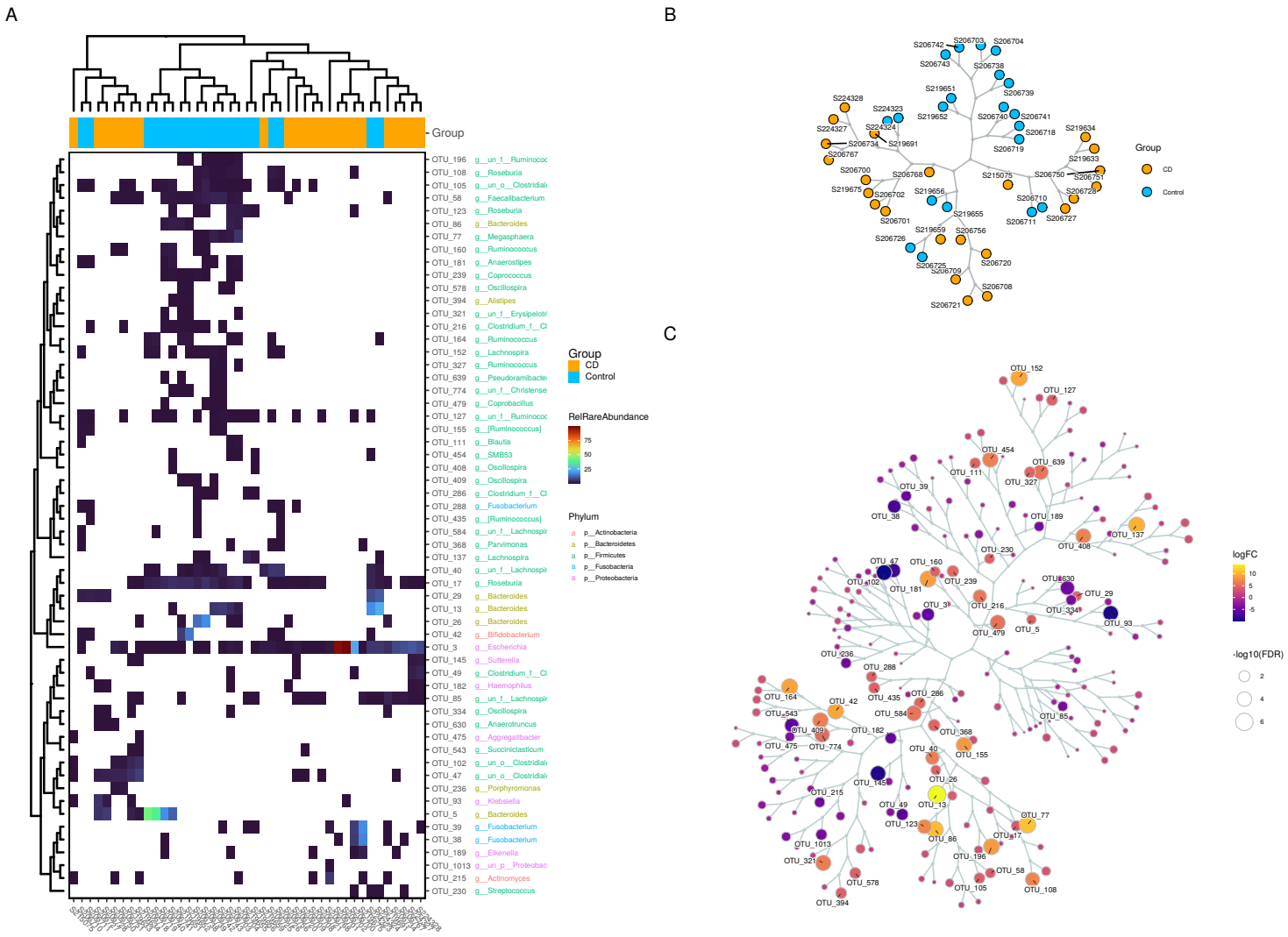

Fig. S15: **The results of different OTUs based on the edgeR_quasi_likelihood with tidybulk** A. The relative abundance heatmap of the different OTUs. B. The hierarchical cluster of samples based on the relative abundance of the different OTUs. C. The hierarchical cluster of OTUs based on the relative abundance of total OTUs, the different OTUs were labeled with their names. We found the cluster of different OTUs in the heatmap is consistent with the different OTUs in the background of total OTUs (C).

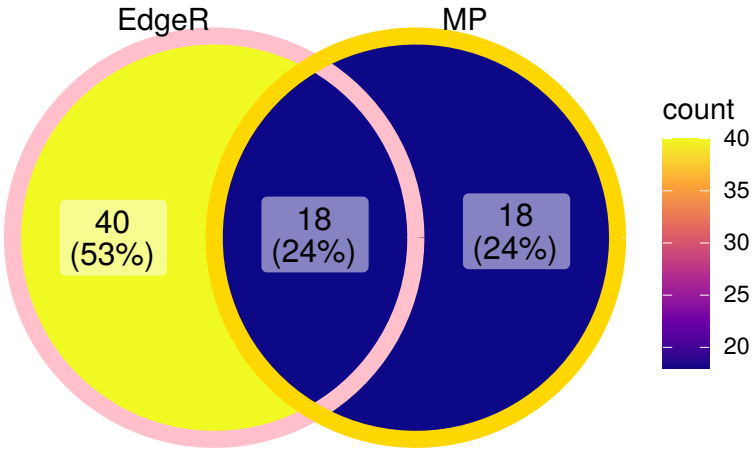

Fig. S16: **The comparison of the different analysis result between the edgeR and MicrobiotaProcess**

```
mpse2 %>%
    mp_extract_feature(addtaxa=T) %>%
```

```
    dplyr::filter(OTU %in% do.call(intersect, unname(DE.method)))
```

```
## # A tibble: 18 x 22
##    OTU    ggupsetOfGroup logFC logCPM     F  PValue     FDR Kingdom Phylum Class
##    <chr>  <list>         <dbl>  <dbl> <dbl>   <dbl>   <dbl> <chr>   <chr>  <chr>
##  1 OTU_2~ <chr [2]>      -4.44   9.41  9.00 3.50e-3 1.79e-2 k__Bac~ p__Ac~ c__A~
##  2 OTU_42 <chr [1]>       8.97  12.1  29.0  5.79e-7 1.48e-5 k__Bac~ p__Ac~ c__A~
##  3 OTU_86 <chr [1]>      10.4   11.1  36.2  8.10e-8 4.66e-6 k__Bac~ p__Ba~ c__B~
##  4 OTU_13 <chr [1]>      13.9   14.5  48.9  1.48e-9 3.41e-7 k__Bac~ p__Ba~ c__B~
##  5 OTU_1~ <chr [1]>       9.78  10.5  33.9  1.74e-7 6.42e-6 k__Bac~ p__Fi~ c__C~
##  6 OTU_1~ <chr [2]>       6.70  10.6  19.8  2.50e-5 2.21e-4 k__Bac~ p__Fi~ c__C~
##  7 OTU_17 <chr [2]>       3.48  13.6   8.95 3.59e-3 1.79e-2 k__Bac~ p__Fi~ c__C~
##  8 OTU_1~ <chr [2]>       6.85  10.4  20.2  2.05e-5 1.96e-4 k__Bac~ p__Fi~ c__C~
##  9 OTU_40 <chr [2]>       6.29  12.7  17.8  5.74e-5 4.89e-4 k__Bac~ p__Fi~ c__C~
## 10 OTU_85 <chr [2]>      -3.88  11.0   7.85 6.23e-3 2.71e-2 k__Bac~ p__Fi~ c__C~
## 11 OTU_58 <chr [2]>       3.61  11.9   7.84 6.25e-3 2.71e-2 k__Bac~ p__Fi~ c__C~
## 12 OTU_1~ <chr [1]>       8.90   9.65 33.6  1.95e-7 6.42e-6 k__Bac~ p__Fi~ c__C~
## 13 OTU_47 <chr [1]>      -7.37  11.8  20.7  1.69e-5 1.69e-4 k__Bac~ p__Fi~ c__C~
## 14 OTU_1~ <chr [1]>      -9.73  10.6  28.2  1.20e-6 2.13e-5 k__Bac~ p__Fi~ c__C~
## 15 OTU_1~ <chr [2]>       3.84  10.6   9.83 2.32e-3 1.27e-2 k__Bac~ p__Fi~ c__C~
## 16 OTU_77 <chr [1]>      10.8   11.5  37.4  5.50e-8 4.21e-6 k__Bac~ p__Fi~ c__C~
## 17 OTU_3  <chr [2]>      -5.35  18.2  15.8  1.40e-4 1.07e-3 k__Bac~ p__Pr~ c__G~
## 18 OTU_93 <chr [1]>      -9.39  10.3  27.5  1.56e-6 2.57e-5 k__Bac~ p__Pr~ c__G~
## # ... with 12 more variables: Order <chr>, Family <chr>, Genus <chr>,
## #   Species <chr>, RareAbundanceBySample <list>, RareAbundanceByGroup <list>,
## #   LDAupper <dbl>, LDAmean <dbl>, LDAlower <dbl>, Sign_Group <chr>,
## #   pvalue <dbl>, fdr <dbl>
```

3 the analysis of the other published pediatric CD stool samples

In the previous chapter, we described how to use *MicrobiotaProcess* to do the analysis of the 16s rDNA data. However, it also can be applied to metagenome or metatranscriptome species community data and the function data analysis. In this chapter, we used the example datasets about the other published pediatric CD stool microbial study (Douglas et al. 2018) to show how to used *MicrobiotaProcess* to do the related analysis. The datasets were obtained from the github[2]. However, to avoid duplication, we only show how to import the 16s dataset, we focused on the analysis of metagenomics and KEGG gene datasets.

3.1 Loading the 16s data and construction of MPSE class

The chapter is similar with the 3, so some operations can refer to the previous chapter 3.

```
cols <- c("orange", "deepskyblue")
cols2 <- c("deepskyblue", "yellow", "#FF9933")
sample.da <- read.table("./data/CD_RF_microbiome/biscuit_metadata.txt", header=TRUE, check.names=FALSE, sep="\t
sample.da %<>% dplyr::select(1:5)
biom <- biomformat::read_biom("./data/CD_RF_microbiome/otu_table_w_tax_BISCUIT.biom")
mpse16s <- biom %>% as.MPSE
mpse16s
```

```
## # A MPSE-tibble (MPSE object) abstraction: 37,392 x 10
## # OTU=984 | Samples=38 | Assays=Abundance | Taxanomy=Kingdom, Phylum, Class, Order, Family, Genus, Speies
##    OTU     Sample Abundance Kingdom     Phylum  Class  Order Family Genus Speies
##    <chr>   <chr>      <dbl> <chr>       <chr>   <chr>  <chr> <chr>  <chr> <chr>
##  1 358030  S15            5 k__Bacteria p__Fir~ c__Cl~ o__C~ f__Ru~ g__u~ s__un~
##  2 196271  S15            0 k__Bacteria p__Fir~ c__Cl~ o__C~ f__La~ g__u~ s__un~
##  3 196270  S15            2 k__Bacteria p__Fir~ c__Cl~ o__C~ f__un~ g__u~ s__un~
##  4 297149  S15            0 k__Bacteria p__Fir~ c__Cl~ o__C~ f__La~ g__u~ s__un~
##  5 3604981 S15            0 k__Bacteria p__Fir~ c__Cl~ o__C~ f__La~ g__B~ s__un~
##  6 240755  S15            0 k__Bacteria p__Pro~ c__Ga~ o__P~ f__Pa~ g__H~ s__in~
##  7 326482  S15            0 k__Bacteria p__Bac~ c__Ba~ o__B~ f__Pr~ g__P~ s__co~
##  8 4393540 S15            0 k__Bacteria p__Bac~ c__Ba~ o__B~ f__[B~ g__u~ s__un~
##  9 4339144 S15            0 k__Bacteria p__Bac~ c__Ba~ o__B~ f__[O~ g__B~ s__un~
## 10 4369050 S15            0 k__Bacteria p__Fus~ c__Fu~ o__F~ f__Fu~ g__F~ s__un~
## # ... with 37,382 more rows
```

```
mpse16s %<>% dplyr::left_join(sample.da, by=c("Sample"="sample_id"))
mpse16s
```

```
## # A MPSE-tibble (MPSE object) abstraction: 37,392 x 14
## # OTU=984 | Samples=38 | Assays=Abundance | Taxanomy=Kingdom, Phylum, Class, Order, Family, Genus, Speies
##    OTU     Sample Abundance disease response sex     age Kingdom  Phylum  Class
##    <chr>   <chr>      <dbl> <chr>   <chr>    <chr> <dbl> <chr>    <chr>   <chr>
##  1 358030  S15            5 CN      CN       Male   15.4 k__Bact~ p__Fir~ c__Cl~
##  2 196271  S15            0 CN      CN       Male   15.4 k__Bact~ p__Fir~ c__Cl~
##  3 196270  S15            2 CN      CN       Male   15.4 k__Bact~ p__Fir~ c__Cl~
##  4 297149  S15            0 CN      CN       Male   15.4 k__Bact~ p__Fir~ c__Cl~
##  5 3604981 S15            0 CN      CN       Male   15.4 k__Bact~ p__Fir~ c__Cl~
##  6 240755  S15            0 CN      CN       Male   15.4 k__Bact~ p__Pro~ c__Ga~
##  7 326482  S15            0 CN      CN       Male   15.4 k__Bact~ p__Bac~ c__Ba~
##  8 4393540 S15            0 CN      CN       Male   15.4 k__Bact~ p__Bac~ c__Ba~
##  9 4339144 S15            0 CN      CN       Male   15.4 k__Bact~ p__Bac~ c__Ba~
## 10 4369050 S15            0 CN      CN       Male   15.4 k__Bact~ p__Fus~ c__Fu~
## # ... with 37,382 more rows, and 4 more variables: Order <chr>, Family <chr>,
## #   Genus <chr>, Speies <chr>
```

[2]https://github.com/LangilleLab/CD_RF_microbiome

3.2 Loading the KEGG data

The KEGG gene abundances were annotated based on the MGS data. It can also be imported as MPSE, and further analyzed using *MicrobiotaProcess*. Here, we only show how to identify the different gene using the *mp_diff_analysis* of *MicrobiotaProcess* (refer to chapter 3.3). Other operations are similar with the analysis of 16s rDNA data (refer to chapter 3).

```
KO.da <- read.table("./data/CD_RF_microbiome/biscuit_mgs_KOs.tsv", header=TRUE, sep = "\t", row.names=1, check.
# Building the MPSE object.
mpseKO <- MPSE(assays=list(Abundance = KO.da))
# merge the sample metadata information.
mpseKO %<>% left_join(sample.da, by=c("Sample"="sample_id"))
```

3.2.1 Different analysis for KEGG genes abundance

The metrics of the KEGG genes is the relative abundance, here we used *mp_diff_analysis* to identify the difference KEGG genes with 'force = TRUE and relative = FALSE', meaning the relative abundance will be used directly.

```
mpseKO %<>%
   mp_diff_analysis(
     .abundance = Abundance,
     force = TRUE,
     relative = FALSE,
     .group = disease,
     filter.p = "pvalue"
   )
```

Then we can perform the KEGG pathway enrichment analysis using clusterProfiler (Wu et al. 2021) and MicrobiomeProfiler (Chen and Yu 2021) developed by our team.

```
# perform KEGG pathway analysis with clusterProfiler and MicrobiomeProfiler
com.xx <- mpseKO %>%
    mp_extract_feature() %>% # Extracting the feature metadata information
    dplyr::filter(pvalue <= 0.05) %>% # Extracting the features pvalue <= 0.05
    compareCluster(OTU~Sign_disease, data=., fun=enrichKO)

# visualizing the enriched pathway with dotplot
p.dot <- dotplot(com.xx) +
         scale_color_gradientn(
           colours = c("#b3eebe", "#46bac2", "#371ea3"),
           guide = guide_colorbar(reverse=TRUE, order=1)
         ) +
         labs(x = NULL) +
         guides(size = guide_legend(override.aes=list(shape=1))) +
         theme(
           panel.grid.major.y = element_line(linetype='dotted', color='#808080'),
           panel.grid.major.x = element_blank()
         )
```

```
# with network plot
set.seed(1024)
p.net <- cnetplot(
           com.xx,
           layout = "fr",
           cex_label_category = 1.8
         ) +
         scale_fill_manual(
           values = cols
         )

p <- aplot::plot_list(p.net, p.dot, widths = c(3, 1), tag_levels="A")
```

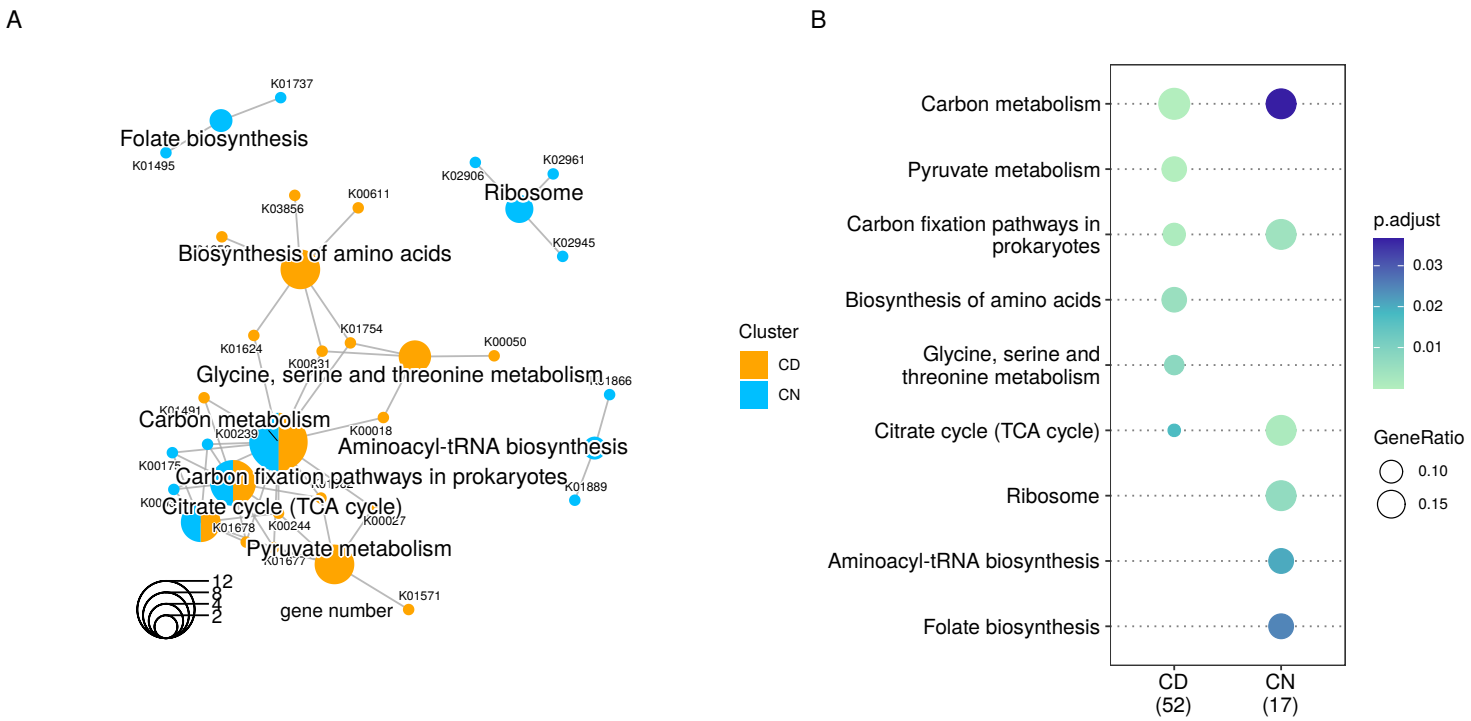

Fig. S17: **The result of KEGG pathway enrichment analysis**

3.3 Loading the MGS data

The taxa abundance data also can be analyzed by *MicrobiotaProcess*, Here we used the example data from output of *MetaPhlAn* (Segata et al. 2012) to show how to perform the related analysis using *MicrobiotaProcess*. The output of other taxa abundance can also be imported and converted to the *MPSE* object, and be further analyzed using *MicrobiotaProcess*, which can refer to chapter3.2 and chapter4.

```
# This is the output of MetaPhlAn2, which might need to specific the 'linenum'
# base on the first several rows whether to contain the metadata information
mpseMGS <- mp_import_metaphlan("./data/CD_RF_microbiome/metaphlan2_out_merged_species.tsv", linenum=1)
# rename the column names of MPSE.
colnames(mpseMGS) <- mpseMGS %>% mp_extract_sample %>% pull(2)
mpseMGS %<>% left_join(sample.da, by=c("Sample"="sample_id"))
mpseMGS
```

```
## # A MPSE-tibble (MPSE object) abstraction: 4,370 x 14
## # OTU=115 | Samples=38 | Assays=Abundance | Taxanomy=Kingdom, Phylum, Class, Order, Family, Genus
##    OTU     Sample Abundance unknown1 disease response sex     age Kingdom Phylum
##    <chr>   <chr>      <dbl> <chr>    <chr>   <chr>    <chr> <dbl> <chr>   <chr>
##  1 s__un_~ S12          0    S12      CN      CN       Fema~   8.6 k__Arc~ p__Eu~
##  2 s__Bif~ S12          0    S12      CN      CN       Fema~   8.6 k__Bac~ p__Ac~
##  3 s__Bif~ S12          0    S12      CN      CN       Fema~   8.6 k__Bac~ p__Ac~
##  4 s__Bif~ S12          0    S12      CN      CN       Fema~   8.6 k__Bac~ p__Ac~
##  5 s__Col~ S12          0    S12      CN      CN       Fema~   8.6 k__Bac~ p__Ac~
##  6 s__Col~ S12          0    S12      CN      CN       Fema~   8.6 k__Bac~ p__Ac~
##  7 s__un_~ S12          0    S12      CN      CN       Fema~   8.6 k__Bac~ p__Ac~
##  8 s__un_~ S12          0    S12      CN      CN       Fema~   8.6 k__Bac~ p__Ac~
##  9 s__Bac~ S12          6.34 S12      CN      CN       Fema~   8.6 k__Bac~ p__Ba~
## 10 s__Bac~ S12          0    S12      CN      CN       Fema~   8.6 k__Bac~ p__Ba~
## # ... with 4,360 more rows, and 4 more variables: Class <chr>, Order <chr>,
## #   Family <chr>, Genus <chr>
```

3.3.1 Alpha diversity analysis in MGS level

The metric of Metagenomics data usually is relative abundance. But the some functions of `MicrobiotaProcess` need to require the abundance is count (in default). To process the relative abundance (not integer), We can specific 'force = TRUE', which meaning the corresponding functions will be calculated directly without rarefied.

```
mpseMGS %<>%
    mp_cal_alpha(
      .abundance = Abundance,
      force = TRUE
    )
p <- mpseMGS %>%
     mp_plot_alpha(
       .group = disease,
       .alpha = c(Observe, Shannon, Pielou)
     ) +
     scale_color_manual(values = cols) +
     scale_fill_manual(values = cols) +
     theme(legend.position = "none")
p
```

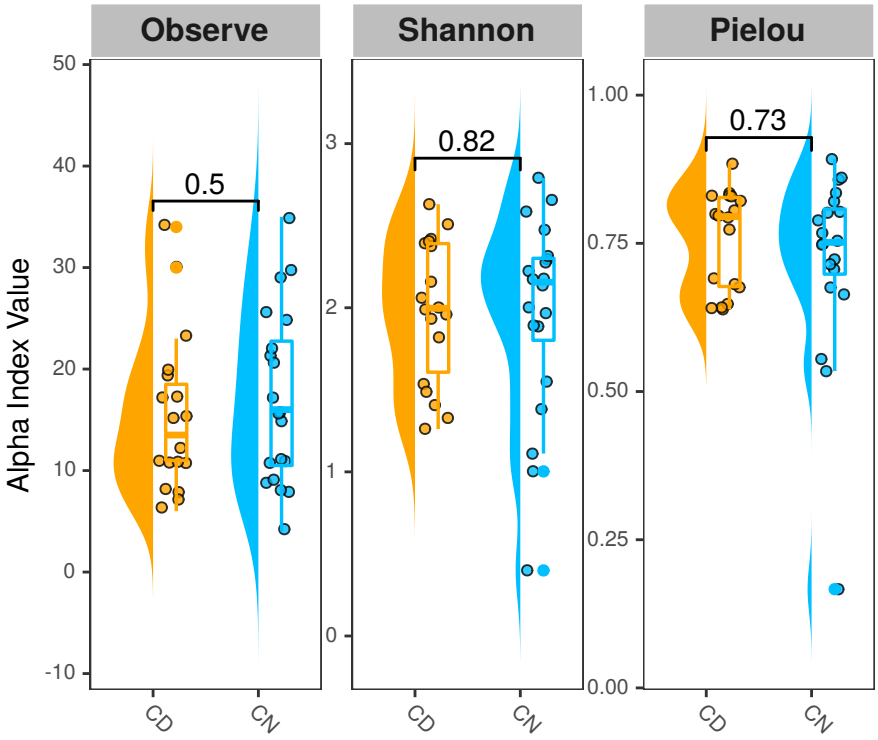

Fig. S18: **The alpha diversity boxplot based on MGS data**

3.3.2 Beta diversity analysis in MGS level

We used *mp_cal_dist* to calculated the distance between the samples, then used *mp_plot_dist* to display the distance with heatmap (Fig.S19.A) and boxplot (Fig.S19.B), then the distance was used to perform the PCoA analysis (Fig.S19.C).

```
mpseMGS %<>%
   mp_decostand(
      .abundance = Abundance,
      method = "hellinger"
    )

mpseMGS %<>%
    mp_cal_dist(
      .abundance = hellinger,
      distmethod = "bray"
    )

mpseMGS %<>%
    mp_cal_pcoa(
      .abundance = hellinger,
      distmethod = "bray"
    )

p1 <- mpseMGS %>%
      mp_plot_dist(
        .distmethod = bray,
        .group=c(disease, response)
      ) %>%
      set_scale_theme(
        x = scale_fill_manual(
              values = cols,
              guide = guide_legend(
                          keywidth = 0.5,
                          keyheight = 0.5,
                          label.theme=element_text(size=6)
                      )
            ),
        aes_var = disease
      ) %>%
      set_scale_theme(
        x = scale_fill_manual(
              values=cols2,
              guide = guide_legend(
                          keywidth = 0.5,
                          keyheight = 0.5,
                          label.theme=element_text(size=6)
                      )
            ),
        aes_var = response
      ) %>%
      set_scale_theme(
        x = scale_size_continuous(
              range = c(1, 3),
              guide = guide_legend(
                          keywidth = 0.5,
                          keyheight = 0.5,
                          label.theme=element_text(size=6)
                      )
            ),
        aes_var = bray
      )
```

```
p2 <- mpseMGS %>%
      mp_plot_dist(
        .distmethod = bray,
        .group = disease,
        group.test = TRUE
      ) +
      scale_color_manual(
        values = c("orange", "#00A08A", "deepskyblue")
      ) +
      scale_fill_manual(
        values = c("orange", "#00A08A", "deepskyblue")
      )

p3 <- mpseMGS %>%
      mp_plot_ord(
        .ord = pcoa,
        .group = disease,
        .size = Observe,
        .starshape = response,
        show.side = FALSE
      ) +
      scale_starshape_manual(values = c(1, 13, 15)) +
      scale_fill_manual(
        values=cols,
        guide=guide_legend(
          keywidth = 0.3,
          keyheight = 0.3,
          label.element = element_text(size = 6),
          override.aes = list(size = 2, starshape = 15)
        )
      ) +
      scale_size_continuous(
        range = c(1, 3),
        guide = guide_legend(
          keywidth = 0.3,
          keyheight = 0.3,
          label.element = element_text(size = 6),
          override.aes = list(starshape = 15)
        )
      )

design <- "\n111\n111\n111\n233\n233\n"
aplot::plot_list(p1, p2, p3, design = design, tag_levels = "A")
```

Then we used *mp_adonis* to perform the Permutational Multivariate Analysis of Variance based on the distance.

```
mpseMGS %<>%
    mp_adonis(
       .abundance = Abundance,
       .formula = ~ disease + response,
       distmethod = "bray",
       permutation = 9999,
       action = "add"
     )
# the result can be extracted with mp_extract_internal_attr
# mpseMGS %>% mp_extract_internal_attr(name = adonis)
```

3.3.3 Different analysis in MGS level

Here, we also used *mp_diff_analysis* to detect the difference taxa, we also specified the 'force = TRUE' and 'relative = FALSE', meaning the metric of abundance (.abundance) was used to perform the analysis directly without rarefied and calculated the relative abundance (Fig.S20).

```
mpseMGS %<>%
    mp_diff_analysis(
       .abundance = Abundance,
       force = TRUE,
       relative = FALSE,
       .group = disease,
       filter.p = "pvalue"
    )

library(forcats)
trda <- mpseMGS %>% mp_extract_tree()

p <- ggtree(trda, layout = 'radial') +
     geom_tiplab(size = 1.8, offset = 11) +
     geom_hilight(
         mapping = aes(
           subset = nodeClass == "Phylum",
           node = node,
           fill = label
         )
     )
p2 <- p +
      ggnewscale::new_scale_fill() +
      geom_fruit(
         data = td_unnest(AbundanceBySample, names_repair=tidyr::tidyr_legacy),
         geom = geom_star,
         mapping = aes(
            x = fct_reorder(Sample, disease, .fun=min),
            size = Abundance,
            fill = disease,
            subset = Abundance > 0
         ),
         starshape = 13,
         offset = 0.02,
         pwidth = 1,
         grid.params = list(linetype=2)
      ) +
      scale_size_continuous(name="Relative Abundance (%)",range = c(1, 3)) +
      scale_fill_manual(values = cols)

p3 <- p2 +
      ggnewscale::new_scale("fill") +
      geom_fruit(
         geom = geom_col,
         mapping = aes(
                       x = LDAmean,
                       fill = Sign_disease,
                       subset = !is.na(LDAmean)
                       ),
         orientation = "y",
         offset = .05,
         pwidth = 0.5,
         width = 0.5, # the parameter of geom_col
         axis.params = list(axis = "x",
                            title = "Log10(LDA)",
```

```
                           title.height = 0.001,
                           title.size = 2,
                           text.size = 1.8,
                           vjust = 1),
         grid.params = list(linetype = 1)
      ) +
      ggnewscale::new_scale("size") +
      geom_point(
         data=td_filter(!is.na(pvalue)),
         mapping = aes(size = -log10(pvalue),
                       fill = Sign_disease
                  ),
         shape = 21
      ) +
      scale_size_continuous(range=c(0.5, 3)) +
      scale_fill_manual(values=cols) +
      theme(
           legend.key.height = unit(0.3, "cm"),
           legend.key.width = unit(0.3, "cm"),
           legend.spacing.y = unit(0.02, "cm"),
           legend.text = element_text(size = 7),
           legend.title = element_text(size = 9),
      )
p3
```

Next, we extract the abundance of the different species, then using ggplot2 (Wickham 2011) to visualize them (Fig.S21).

```
deT <- mpseMGS %>% mp_extract_tree() %>% dplyr::filter(!is.na(fdr) & isTip, keep.td=F) %>% dplyr::pull(label)
mpseMGS %>%
    mp_extract_abundance(taxa.class="OTU") %>%
    dplyr::filter(label %in% deT) %>%
    tidyr::unnest(AbundanceBySample) %>%
    ggplot(mapping=aes(x=disease, y=Abundance, fill=disease)) +
    geom_boxplot() +
    facet_wrap(facets = vars(label), nrow = 1, scales = "free", strip.position = "right") +
    ggsignif::geom_signif(comparisons=list(c("CD", "CN"))) +
    scale_fill_manual(values=cols, guide="none") +
    labs(x=NULL, y="relative abundance (%)")
```

4 The analysis of the mosquito ecology data using MicrobiotaProcess

MicrobiotaProcess also can be used to perform the other related ecology data analysis, besides the microbial community data. Here, we used an example data about a Mosquito ecology study (REISKIND et al. 2017) to show how to use MicrobiotaProcess to perform the analysis of the related ecology study. The data was obtained from the github[3].

4.1 Loading data and Construction of MPSE object

The 1 to 14 columns are the sample metadata including the study site, and habitat, etc. and the others columns represent the abundance of mosquito species the in each sample.

```
data <- read.csv("./data/Mosquito_ecology/data.csv", row.names=1)
abun.d <- data[, 14:36]
sample.d <- data[, 1:13]
# We implements `MPSE` function to build the `MPSE` object, which requires the abundance table (matrix-like).
mpse <- MPSE(assays=list(Abundance=t(abun.d)), colData=sample.d)
mpse
```

```
## # A MPSE-tibble (MPSE object) abstraction: 1,035 x 16
## # OTU=23 | Samples=45 | Assays=Abundance | Taxanomy=NULL
##    OTU      Sample Abundance Region Transect Habitat DeciduousForest
##    <chr>    <chr>      <int> <chr>  <chr>    <chr>             <dbl>
##  1 Cx.sal   DU1.1         19 Durham DU1      Field              125.
##  2 Ae.albo  DU1.1          0 Durham DU1      Field              125.
##  3 Ae.cin   DU1.1          1 Durham DU1      Field              125.
##  4 Ae.vex   DU1.1         16 Durham DU1      Field              125.
##  5 Ps.fer   DU1.1          1 Durham DU1      Field              125.
##  6 Cx.err   DU1.1        372 Durham DU1      Field              125.
##  7 Ps.col   DU1.1        104 Durham DU1      Field              125.
##  8 Ae.tris  DU1.1          0 Durham DU1      Field              125.
##  9 Cx.pip.q DU1.1          2 Durham DU1      Field              125.
## 10 Ae.can   DU1.1          0 Durham DU1      Field              125.
## # ... with 1,025 more rows, and 9 more variables: EvergreenForest <dbl>,
## #   Grassland <dbl>, MixedForest <dbl>, ShrubScrub <dbl>, BarrenLand <dbl>,
## #   Building <dbl>, Pavement <dbl>, CultivatedCrops <dbl>, TrapNights <int>
```

4.2 Alpha diversity analysis of the Mosquito ecology study

The `MicrobiotaProcess` provides some verbs of `dplyr`, which allows user to explore the `MPSE` class effectively and develop reproducible and human-readable pipelines

```
cols = c("lightgoldenrod1", "orange","chartreuse2", "chartreuse4", "darkgreen")
# Adjusting the order of Habitat
mpse %<>%
   dplyr::mutate(
     Habitat = factor(
       Habitat,
       levels = c("Field", "NearField", "Edge", "NearForest", "Forest")
    )
   )
mpse
```

```
## # A MPSE-tibble (MPSE object) abstraction: 1,035 x 16
## # OTU=23 | Samples=45 | Assays=Abundance | Taxanomy=NULL
##    OTU      Sample Abundance Region Transect Habitat DeciduousForest
##    <chr>    <chr>      <int> <chr>  <chr>    <fct>             <dbl>
##  1 Cx.sal   DU1.1         19 Durham DU1      Field              125.
##  2 Ae.albo  DU1.1          0 Durham DU1      Field              125.
```

[3] https://github.com/rgriff23/Mosquito_ecology

```
##  3 Ae.cin   DU1.1          1 Durham DU1      Field              125.
##  4 Ae.vex   DU1.1         16 Durham DU1      Field              125.
##  5 Ps.fer   DU1.1          1 Durham DU1      Field              125.
##  6 Cx.err   DU1.1        372 Durham DU1      Field              125.
##  7 Ps.col   DU1.1        104 Durham DU1      Field              125.
##  8 Ae.tris  DU1.1          0 Durham DU1      Field              125.
##  9 Cx.pip.q DU1.1          2 Durham DU1      Field              125.
## 10 Ae.can   DU1.1          0 Durham DU1      Field              125.
## # ... with 1,025 more rows, and 9 more variables: EvergreenForest <dbl>,
## #   Grassland <dbl>, MixedForest <dbl>, ShrubScrub <dbl>, BarrenLand <dbl>,
## #   Building <dbl>, Pavement <dbl>, CultivatedCrops <dbl>, TrapNights <int>
```

```
# force=TRUE meaning the Abundance will be used to calculate the alpha index without rarefaction
mpse %<>% mp_cal_alpha(.abundance=Abundance, force=TRUE)
# test the relationship between the Observe Species and Habitat or Shannon and Habitat.
tb1 <- mpse %>% mp_extract_sample() %>% lm(formula=Observe ~ Habitat, data=.) %>% anova() %>% broom::tidy()
tb2 <- mpse %>% mp_extract_sample() %>% lm(formula=Shannon ~ Habitat, data=.) %>% anova() %>% broom::tidy()
```

The result of ANOVA test revealed that the richness of the mosquito species was significantly associated with the **habitat**. Then the result was visualized by *mp_plot_alpha* (Fig.S22).

```
p.alpha <- mpse %>%
     mp_plot_alpha(.group = Habitat, .alpha = c(Observe, Shannon), test = NULL) +
     scale_fill_manual(values = cols) +
     scale_color_manual(values = cols) +
     theme(legend.position = "none")
library(ggpp)
# building the table layer
tb1 %<>% dplyr::slice(1) %>% select(statistic, p.value) %>% round(3)
tb2 %<>% dplyr::slice(1) %>% select(statistic, p.value) %>% round(3)
df <- tibble(npcx=c(0.9, 0.9), npcy=c(0.05, 0.05), tb=list(tb1, tb2), Measure=c("Observe", "Shannon"))

p.alpha <- p.alpha +
           geom_table_npc(
             data = df,
             mapping = aes(
               npcx = npcx,
               npcy = npcy,
               label = tb
             ),
             table.theme = ttheme_gtminimal
           )
p.alpha
```

4.3 Beta Diversity Analysis of the Mosquito ecology study

Here, we use the cca (constrained correspondence analysis) to test which environment factor is related to the Mosquito species in the habitat (Fig.S23).

```
mpse %<>%
    mutate(NormAbun=sqrt(Abundance)/TrapNights) %>%
    mp_cal_cca(
       .abundance  = NormAbun,
       .formula = ~DeciduousForest+
           EvergreenForest+
           Grassland+
           MixedForest+
           ShrubScrub+
           Condition(
             BarrenLand+
             Building+
```

```
            Pavement+
            CultivatedCrops
          )
    )
mpse
```

```
## # A MPSE-tibble (MPSE object) abstraction: 1,035 x 26
## # OTU=23 | Samples=45 | Assays=Abundance, NormAbun | Taxanomy=NULL
##    OTU      Sample Abundance NormAbun Region Transect Habitat DeciduousForest
##    <chr>    <chr>      <int>    <dbl> <chr>  <chr>    <fct>             <dbl>
##  1 Cx.sal   DU1.1         19    0.436 Durham DU1      Field              125.
##  2 Ae.albo  DU1.1          0    0     Durham DU1      Field              125.
##  3 Ae.cin   DU1.1          1    0.1   Durham DU1      Field              125.
##  4 Ae.vex   DU1.1         16    0.4   Durham DU1      Field              125.
##  5 Ps.fer   DU1.1          1    0.1   Durham DU1      Field              125.
##  6 Cx.err   DU1.1        372    1.93  Durham DU1      Field              125.
##  7 Ps.col   DU1.1        104    1.02  Durham DU1      Field              125.
##  8 Ae.tris  DU1.1          0    0     Durham DU1      Field              125.
##  9 Cx.pip.q DU1.1          2    0.141 Durham DU1      Field              125.
## 10 Ae.can   DU1.1          0    0     Durham DU1      Field              125.
## # ... with 1,025 more rows, and 18 more variables: EvergreenForest <dbl>,
## #   Grassland <dbl>, MixedForest <dbl>, ShrubScrub <dbl>, BarrenLand <dbl>,
## #   Building <dbl>, Pavement <dbl>, CultivatedCrops <dbl>, TrapNights <int>,
## #   Observe <dbl>, Chao1 <dbl>, ACE <dbl>, Shannon <dbl>, Simpson <dbl>,
## #   Pielou <dbl>, CCA1 (25.28%) <dbl>, CCA2 (7.34%) <dbl>, CCA3 (3.39%) <dbl>
```

The raw result of pCCA was added the *internal_attr*, which can be extract by *mp_extract_internal_attr* with specific *name=cca*. Then it can be performed the significance test using the functions of *vegan* (Oksanen et al. 2020), such as *anova.cca*, *permutest*.

```
# Extract the raw result of cca analysis
# And significance test with anova

mpse %>%
    mp_extract_internal_attr(name=cca) %>%
    anova()
```

```
## Permutation test for cca under reduced model
## Permutation: free
## Number of permutations: 999
##
## Model: cca(formula = x ~ DeciduousForest + EvergreenForest + Grassland + MixedForest + ShrubScrub + Conditio
##          Df ChiSquare      F Pr(>F)
## Model     5   0.38999 4.4365  0.001 ***
## Residual 35   0.61534
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Further we used *mp_envfit* to identity the environment variables that were significantly associated with the mosquito communities.

```
# fits environmental vectors onto cca
mpse %<>%
    mp_envfit(
       .ord = cca,
       .env = c(
          DeciduousForest,
          EvergreenForest,
          Grassland,
          MixedForest,
          ShrubScrub
        ),
       action = "add",
```

```
      permutation = 9999
    )

# Extract the raw result of envfit analysis
mpse %>% mp_extract_internal_attr(name=cca_envfit)
```

```
##
## ***VECTORS
##
##                     CCA1     CCA2     CCA3     r2 Pr(>r)
## DeciduousForest  0.42979  0.90272 -0.01945 0.3804 0.0020 **
## EvergreenForest  0.91539 -0.34612 -0.20559 0.5557 0.0001 ***
## Grassland       -0.97679 -0.21356  0.01639 0.7216 0.0001 ***
## MixedForest      0.77120 -0.25826  0.58186 0.1936 0.0929 .
## ShrubScrub      -0.73942  0.22976 -0.63283 0.2595 0.0537 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## Permutation: free
## Number of permutations: 9999
```

Then we used *mp_plot_ord* to visualize the result of pCCA (Fig.S23).

```
# visualization only pCCA
f <- mpse %>%
     mp_plot_ord(
       .ord = cca,
       .group = Habitat,
       .size = Observe,
       .starshape = Region,
       show.side = FALSE,
       show.envfit = FALSE,
       colour = 'black',
       bg.colour = 'white'
     ) +
     scale_starshape_manual(values=c(1, 13, 15)) +
     scale_fill_manual(
        values = cols,
        guide = guide_legend(
          override.aes = list(starshape=15)
        )
     ) +
     scale_size_continuous(
       range = c(1, 3),
       guide = guide_legend(override.aes = list(starshape=15))
     ) +
     theme(
        legend.key.height = unit(0.3, "cm"),
        legend.key.width = unit(0.3, "cm"),
        legend.spacing.y = unit(0.02, "cm"),
        legend.text = element_text(size = 7),
        legend.title = element_text(size = 9),
     )

# visualization with envfit result
p <- mpse %>%
     mp_plot_ord(
       .ord = cca,
       .group = Habitat,
       .size = Observe,
       .starshape = Region,
       show.side = FALSE,
```

```
        show.envfit = TRUE,
        colour = "black",
        bg.colour = "white"
      ) +
      scale_starshape_manual(values=c(1, 13, 15)) +
      scale_fill_manual(
         values = cols,
         guide = guide_legend(
           override.aes = list(starshape=15)
         )
      ) +
      scale_size_continuous(
        range = c(1, 3),
        guide = guide_legend(override.aes = list(starshape=15))
      ) +
      theme(
         legend.key.height = unit(0.3, "cm"),
         legend.key.width = unit(0.3, "cm"),
         legend.spacing.y = unit(0.02, "cm"),
         legend.text = element_text(size = 7),
         legend.title = element_text(size = 9),
      )

aplot::plot_list(f, p, tag_levels="A")
```

4.4 The distribution of Mosquito species in the study.

We used *mp_cal_abundance* and *mp_plot_abundance* to calculated and visualized the abundance of the Mosquito species in the study (Fig.S24).

```
cols2 <- c("deepskyblue", "yellow", "#FF9933")

# The theme and scale of fill of heatmap
Abund.char <- list(
            scale_fill_viridis_c(option = "H"),
            theme(
              axis.text.x = element_text(size = 6),
              axis.text.y = element_text(size = 8),
              legend.title = element_text(size = 7),
              legend.text = element_text(size = 5),
              legend.key.width = unit(0.3, "cm"),
              legend.key.height = unit(0.3, "cm")
            )
      )

# The theme and legend of annotate bar of 'Habitat' variable
Habitat.char <- list(
            scale_fill_manual(values = cols),
            theme(
              legend.key.height = unit(0.3, "cm"),
              legend.key.width = unit(0.3, "cm"),
              legend.spacing.y = unit(0.02, "cm"),
              legend.text = element_text(size = 7),
              legend.title = element_text(size = 9)
            )
      )

# The theme and legend of annotate bar of 'Region' variable
Region.char <- list(
```

```
            scale_fill_manual(values = cols2),
            theme(
              legend.key.height = unit(0.3, "cm"),
              legend.key.width = unit(0.3, "cm"),
              legend.spacing.y = unit(0.02, "cm"),
              legend.text = element_text(size = 7),
              legend.title = element_text(size = 9)
            )
      )

# visualization of the count abundance.
p.count <- mpse %>%
    mp_cal_abundance(
      .abundance = Abundance,
      force = T,
      relative = F
    ) %>%
    mp_plot_abundance(
      .abundance = Abundance,
      force = T,
      relative = F,
      geom = "heatmap",
      topn = "all",
      .group = c(Habitat, Region)
    ) %>%
    set_scale_theme(
      x = Abund.char,
      aes_var = Abundance
    ) %>%
    set_scale_theme(
      x = Habitat.char,
      aes_var = Habitat
    ) %>%
    set_scale_theme(
      x = Region.char,
      aes_var = Region
    )

# visualization of the relative abundance
p.rel <- mpse %>%
    mp_cal_abundance(
      .abundance = Abundance,
      force = T,
      relative = T
    ) %>%
    mp_plot_abundance(
      .abundance = Abundance,
      force = T,
      relative = T,
      geom = "heatmap",
      topn = "all",
      .group = c(Habitat, Region)
    ) %>%
    set_scale_theme(
      x = Abund.char,
      aes_var = RelAbundance
    ) %>%
    set_scale_theme(
      x = Habitat.char,
      aes_var = Habitat
```

```
    ) %>%
    set_scale_theme(
      x = Region.char,
      aes_var = Region
    )

aplot::plot_list(p.count, p.rel, tag_levels="A")
```

Then We can use *mp_diff_analysis* to identify the species that have significant difference abundance between the **field** and **forest**. We found the Cx.sal (*Culex salinarius*) and Ps.col (*Psorophora columbiae*) were significantly enriched in **field**, However, the Ae.albo (*Aedes albopicta*), Ae.cin (*Aedes cinereus*), Ps.fer (*Psorophora ferox*), Ae.tris (*Aedes triseriatus*), Ae.can (*Aedes canadensis*), Ae.hen (*Aedes hendersoni*), Ae.atl (*Aedes atlanticus*) and Ae.dup (*Aedes dupreei*) were significantly enriched in the **forest**

```
mpse %>%
    dplyr::filter(Habitat %in% c("Field", "Forest")) %>%
    dplyr::mutate(Habitat = as.vector(Habitat)) %>%
    mp_diff_analysis(.abundance=Abundance, force=T, relative=T, .group=Habitat) %>%
    mp_extract_feature() %>%
    dplyr::filter(fdr<=0.05) %>%
    print(width=200)
```

```
## # A tibble: 10 x 8
##    OTU     AbundanceBySample LDAupper LDAmean LDAlower Sign_Habitat   pvalue
##    <chr>   <list>                <dbl>   <dbl>    <dbl> <chr>           <dbl>
##  1 Cx.sal  <tibble [18 x 6]>     4.96    4.92     4.87 Field        0.00705
##  2 Ae.albo <tibble [18 x 6]>     4.83    4.79     4.75 Forest       0.000229
##  3 Ae.cin  <tibble [18 x 6]>     4.36    4.31     4.25 Forest       0.0159
##  4 Ps.fer  <tibble [18 x 6]>     4.94    4.90     4.87 Forest       0.00122
##  5 Ps.col  <tibble [18 x 6]>     5.26    5.24     5.22 Field        0.000327
##  6 Ae.tris <tibble [18 x 6]>     4.49    4.46     4.43 Forest       0.000530
##  7 Ae.can  <tibble [18 x 6]>     4.28    4.24     4.19 Forest       0.0119
##  8 Ae.hen  <tibble [18 x 6]>     4.28    4.23     4.18 Forest       0.000483
##  9 Ae.atl  <tibble [18 x 6]>     4.59    4.56     4.52 Forest       0.00311
## 10 Ae.dup  <tibble [18 x 6]>     4.03    3.96     3.88 Forest       0.0119
##        fdr
##      <dbl>
##  1 0.0211
##  2 0.00278
##  3 0.0334
##  4 0.00513
##  5 0.00278
##  6 0.00278
##  7 0.0278
##  8 0.00278
##  9 0.0109
## 10 0.0278
```

5 Session information

Here is the output of sessionInfo() on the system on which this document was compiled:

```
## - Session info ---------------------------------------------------------------------------------------------
##  setting  value
##  version  R version 4.1.1 (2021-08-10)
##  os       Ubuntu 18.04.4 LTS
##  system   x86_64, linux-gnu
##  ui       X11
##  language (EN)
##  collate  en_US.UTF-8
```

```
##  ctype    en_US.UTF-8
##  tz       Asia/Shanghai
##  date     2022-01-05
##
## - Packages -------------------------------------------------------------------------------------------
##  package              * version   date       lib source
##  AnnotationDbi          1.56.1    2021-10-29 [1] Bioconductor
##  ape                    5.6       2021-12-21 [1] CRAN (R 4.1.1)
##  aplot                * 0.1.1     2021-09-22 [1] CRAN (R 4.1.1)
##  assertthat             0.2.1     2019-03-21 [1] CRAN (R 4.1.1)
##  attempt                0.3.1     2020-05-03 [1] CRAN (R 4.1.1)
##  backports              1.3.0     2021-10-27 [1] CRAN (R 4.1.1)
##  Biobase              * 2.54.0    2021-10-26 [1] Bioconductor
##  BiocGenerics         * 0.40.0    2021-10-26 [1] Bioconductor
##  BiocManager            1.30.16   2021-06-15 [1] CRAN (R 4.1.1)
##  BiocParallel           1.28.0    2021-10-26 [1] Bioconductor
##  biomformat             1.22.0    2021-10-26 [1] Bioconductor
##  Biostrings             2.62.0    2021-10-26 [1] Bioconductor
##  bit                    4.0.4     2020-08-04 [1] CRAN (R 4.1.1)
##  bit64                  4.0.5     2020-08-30 [1] CRAN (R 4.1.1)
##  bitops                 1.0-7     2021-04-24 [1] CRAN (R 4.1.1)
##  blob                   1.2.2     2021-07-23 [1] CRAN (R 4.1.1)
##  bookdown               0.24      2021-09-02 [1] CRAN (R 4.1.1)
##  broom                  0.7.10    2021-10-31 [1] CRAN (R 4.1.1)
##  bslib                  0.3.1     2021-10-06 [1] CRAN (R 4.1.1)
##  cachem                 1.0.6     2021-08-19 [1] CRAN (R 4.1.1)
##  callr                  3.7.0     2021-04-20 [1] CRAN (R 4.1.1)
##  class                  7.3-19    2021-05-03 [1] CRAN (R 4.1.1)
##  classInt               0.4-3     2020-04-07 [1] CRAN (R 4.1.1)
##  cli                    3.1.0     2021-10-27 [1] CRAN (R 4.1.1)
##  cluster                2.1.2     2021-04-17 [1] CRAN (R 4.1.1)
##  clusterProfiler      * 4.2.0     2021-10-26 [1] Bioconductor
##  codetools              0.2-18    2020-11-04 [1] CRAN (R 4.1.1)
##  coin                 * 1.4-2     2021-10-08 [1] CRAN (R 4.1.1)
##  colorspace             2.0-2     2021-06-24 [1] CRAN (R 4.1.1)
##  config                 0.3.1     2020-12-17 [1] CRAN (R 4.1.1)
##  conflicted           * 1.0.4     2019-06-21 [1] CRAN (R 4.1.1)
##  corrr                  0.4.3     2020-11-24 [1] CRAN (R 4.1.1)
##  crayon                 1.4.2     2021-10-29 [1] CRAN (R 4.1.1)
##  data.table             1.14.2    2021-09-27 [1] CRAN (R 4.1.1)
##  DBI                    1.1.1     2021-01-15 [1] CRAN (R 4.1.1)
##  DelayedArray           0.20.0    2021-10-26 [1] Bioconductor
##  desc                   1.4.0     2021-09-28 [1] CRAN (R 4.1.1)
##  digest                 0.6.28    2021-09-23 [1] CRAN (R 4.1.1)
##  DO.db                  2.9       2021-12-13 [1] Bioconductor
##  dockerfiler            0.1.4     2021-09-03 [1] CRAN (R 4.1.1)
##  DOSE                   3.20.1    2021-11-18 [1] Bioconductor
##  downloader             0.4       2015-07-09 [1] CRAN (R 4.1.1)
##  dplyr                  1.0.7     2021-06-18 [1] CRAN (R 4.1.1)
##  DT                     0.19      2021-09-02 [1] CRAN (R 4.1.1)
##  dtplyr                 1.1.0     2021-02-20 [1] CRAN (R 4.1.1)
##  e1071                  1.7-9     2021-09-16 [1] CRAN (R 4.1.1)
##  edgeR                * 3.36.0    2021-10-26 [1] Bioconductor
##  ellipsis               0.3.2     2021-04-29 [1] CRAN (R 4.1.1)
##  enrichplot           * 1.14.1    2021-10-31 [1] Bioconductor
##  evaluate               0.14      2019-05-28 [1] CRAN (R 4.1.1)
##  fansi                  0.5.0     2021-05-25 [1] CRAN (R 4.1.1)
##  farver                 2.1.0     2021-02-28 [1] CRAN (R 4.1.1)
##  fastmap                1.1.0     2021-01-25 [1] CRAN (R 4.1.1)
##  fastmatch              1.1-3     2021-07-23 [1] CRAN (R 4.1.1)
```

```
##  fgsea                1.20.0    2021-10-26 [1] Bioconductor
##  forcats            * 0.5.1     2021-01-27 [1] CRAN (R 4.1.1)
##  foreach              1.5.1     2020-10-15 [1] CRAN (R 4.1.1)
##  fs                   1.5.0     2020-07-31 [1] CRAN (R 4.1.1)
##  generics             0.1.1     2021-10-25 [1] CRAN (R 4.1.1)
##  GenomeInfoDb       * 1.30.0    2021-10-26 [1] Bioconductor
##  GenomeInfoDbData     1.2.7     2021-10-29 [1] Bioconductor
##  GenomicRanges      * 1.46.0    2021-10-26 [1] Bioconductor
##  ggalluvial           0.12.3    2020-12-05 [1] CRAN (R 4.1.1)
##  ggforce              0.3.3     2021-03-05 [1] CRAN (R 4.1.1)
##  ggfun                0.0.4     2021-09-17 [1] CRAN (R 4.1.1)
##  ggh4x                0.2.0     2021-08-21 [1] CRAN (R 4.1.1)
##  gghalves             0.1.1     2020-11-08 [1] CRAN (R 4.1.1)
##  ggnewscale         * 0.4.5     2021-01-11 [1] CRAN (R 4.1.1)
##  ggplot2            * 3.3.5     2021-06-25 [1] CRAN (R 4.1.1)
##  ggplotify            0.1.0     2021-09-02 [1] CRAN (R 4.1.1)
##  ggpp               * 0.4.2     2021-07-31 [1] CRAN (R 4.1.1)
##  ggraph               2.0.5     2021-02-23 [1] CRAN (R 4.1.1)
##  ggrepel            * 0.9.1     2021-01-15 [1] CRAN (R 4.1.1)
##  ggside               0.2.0     2021-12-11 [1] CRAN (R 4.1.1)
##  ggsignif             0.6.3     2021-09-09 [1] CRAN (R 4.1.1)
##  ggstar             * 1.0.3     2021-12-03 [1] Github (xiangpin/ggstar@5e1f018)
##  ggtree             * 3.3.1     2021-12-31 [1] Bioconductor
##  ggtreeExtra        * 1.5.1     2021-11-24 [1] Bioconductor
##  ggupset              0.3.0     2020-05-05 [1] CRAN (R 4.1.1)
##  ggVennDiagram      * 1.1.4     2021-07-07 [1] CRAN (R 4.1.1)
##  glue                 1.5.0     2021-11-07 [1] CRAN (R 4.1.1)
##  GO.db                3.14.0    2021-11-30 [1] Bioconductor
##  golem                0.3.1     2021-04-17 [1] CRAN (R 4.1.1)
##  GOSemSim             2.20.0    2021-10-26 [1] Bioconductor
##  graphlayouts         0.7.1     2020-10-26 [1] CRAN (R 4.1.1)
##  gridExtra            2.3       2017-09-09 [1] CRAN (R 4.1.1)
##  gridGraphics         0.5-1     2020-12-13 [1] CRAN (R 4.1.1)
##  gtable               0.3.0     2019-03-25 [1] CRAN (R 4.1.1)
##  hms                  1.1.1     2021-09-26 [1] CRAN (R 4.1.1)
##  htmltools            0.5.2     2021-08-25 [1] CRAN (R 4.1.1)
##  htmlwidgets          1.5.4     2021-09-08 [1] CRAN (R 4.1.1)
##  httpuv               1.6.3     2021-09-09 [1] CRAN (R 4.1.1)
##  httr                 1.4.2     2020-07-20 [1] CRAN (R 4.1.1)
##  igraph               1.2.7     2021-10-15 [1] CRAN (R 4.1.1)
##  IRanges            * 2.28.0    2021-10-26 [1] Bioconductor
##  iterators            1.0.13    2020-10-15 [1] CRAN (R 4.1.1)
##  jquerylib            0.1.4     2021-04-26 [1] CRAN (R 4.1.1)
##  jsonlite             1.7.2     2020-12-09 [1] CRAN (R 4.1.1)
##  kableExtra         * 1.3.4     2021-02-20 [1] CRAN (R 4.1.1)
##  KEGGREST             1.34.0    2021-10-26 [1] Bioconductor
##  KernSmooth           2.23-20   2021-05-03 [1] CRAN (R 4.1.1)
##  knitr                1.36      2021-09-29 [1] CRAN (R 4.1.1)
##  labeling             0.4.2     2020-10-20 [1] CRAN (R 4.1.1)
##  later                1.3.0     2021-08-18 [1] CRAN (R 4.1.1)
##  lattice              0.20-45   2021-09-22 [1] CRAN (R 4.1.1)
##  lazyeval             0.2.2     2019-03-15 [1] CRAN (R 4.1.1)
##  libcoin              1.0-9     2021-09-27 [1] CRAN (R 4.1.1)
##  lifecycle            1.0.1     2021-09-24 [1] CRAN (R 4.1.1)
##  limma              * 3.50.0    2021-10-26 [1] Bioconductor
##  locfit               1.5-9.4   2020-03-25 [1] CRAN (R 4.1.1)
##  magrittr           * 2.0.1     2020-11-17 [1] CRAN (R 4.1.1)
##  MASS                 7.3-54    2021-05-03 [1] CRAN (R 4.1.1)
##  Matrix               1.3-4     2021-06-01 [1] CRAN (R 4.1.1)
##  MatrixGenerics     * 1.6.0     2021-10-26 [1] Bioconductor
```

```
##  matrixStats          * 0.61.0    2021-09-17 [1] CRAN (R 4.1.1)
##  memoise                2.0.0     2021-01-26 [1] CRAN (R 4.1.1)
##  mgcv                   1.8-38    2021-10-06 [1] CRAN (R 4.1.1)
##  MicrobiomeProfiler   * 1.0.0     2021-10-26 [1] Bioconductor
##  MicrobiotaProcess    * 1.7.5     2021-12-31 [1] Bioconductor
##  mime                   0.12      2021-09-28 [1] CRAN (R 4.1.1)
##  modeltools             0.2-23    2020-03-05 [1] CRAN (R 4.1.1)
##  multcomp               1.4-17    2021-04-29 [1] CRAN (R 4.1.1)
##  munsell                0.5.0     2018-06-12 [1] CRAN (R 4.1.1)
##  mvtnorm                1.1-3     2021-10-08 [1] CRAN (R 4.1.1)
##  nlme                   3.1-153   2021-09-07 [1] CRAN (R 4.1.1)
##  patchwork            * 1.1.1     2020-12-17 [1] CRAN (R 4.1.1)
##  permute                0.9-5     2019-03-12 [1] CRAN (R 4.1.1)
##  pillar                 1.6.4     2021-10-18 [1] CRAN (R 4.1.1)
##  pkgbuild               1.2.0     2020-12-15 [1] CRAN (R 4.1.1)
##  pkgconfig              2.0.3     2019-09-22 [1] CRAN (R 4.1.1)
##  pkgload                1.2.2     2021-09-11 [1] CRAN (R 4.1.1)
##  plyr                   1.8.6     2020-03-03 [1] CRAN (R 4.1.1)
##  png                    0.1-7     2013-12-03 [1] CRAN (R 4.1.1)
##  polyclip               1.10-0    2019-03-14 [1] CRAN (R 4.1.1)
##  preprocessCore         1.56.0    2021-10-26 [1] Bioconductor
##  prettyunits            1.1.1     2020-01-24 [1] CRAN (R 4.1.1)
##  processx               3.5.2     2021-04-30 [1] CRAN (R 4.1.1)
##  promises               1.2.0.1   2021-02-11 [1] CRAN (R 4.1.1)
##  proxy                  0.4-26    2021-06-07 [1] CRAN (R 4.1.1)
##  ps                     1.6.0     2021-02-28 [1] CRAN (R 4.1.1)
##  purrr                  0.3.4     2020-04-17 [1] CRAN (R 4.1.1)
##  qvalue                 2.26.0    2021-10-26 [1] Bioconductor
##  R6                     2.5.1     2021-08-19 [1] CRAN (R 4.1.1)
##  RColorBrewer           1.1-2     2014-12-07 [1] CRAN (R 4.1.1)
##  Rcpp                   1.0.7     2021-07-07 [1] CRAN (R 4.1.1)
##  RCurl                  1.98-1.5  2021-09-17 [1] CRAN (R 4.1.1)
##  readr                  2.0.2     2021-09-27 [1] CRAN (R 4.1.1)
##  remotes                2.4.1     2021-09-29 [1] CRAN (R 4.1.1)
##  reshape2               1.4.4     2020-04-09 [1] CRAN (R 4.1.1)
##  rhdf5                  2.38.0    2021-10-26 [1] Bioconductor
##  rhdf5filters           1.6.0     2021-10-26 [1] Bioconductor
##  Rhdf5lib               1.16.0    2021-10-26 [1] Bioconductor
##  rlang                  0.4.12    2021-10-18 [1] CRAN (R 4.1.1)
##  rmarkdown              2.11      2021-09-14 [1] CRAN (R 4.1.1)
##  roxygen2               7.1.2     2021-09-08 [1] CRAN (R 4.1.1)
##  rprojroot              2.0.2     2020-11-15 [1] CRAN (R 4.1.1)
##  RSQLite                2.2.8     2021-08-21 [1] CRAN (R 4.1.1)
##  rstudioapi             0.13      2020-11-12 [1] CRAN (R 4.1.1)
##  rvcheck              * 0.2.0     2021-09-14 [1] CRAN (R 4.1.1)
##  RVenn                  1.1.0     2019-07-18 [1] CRAN (R 4.1.1)
##  rvest                  1.0.2     2021-10-16 [1] CRAN (R 4.1.1)
##  S4Vectors            * 0.32.0    2021-10-26 [1] Bioconductor
##  sandwich               3.0-1     2021-05-18 [1] CRAN (R 4.1.1)
##  sass                   0.4.0     2021-05-12 [1] CRAN (R 4.1.1)
##  scales                 1.1.1     2020-05-11 [1] CRAN (R 4.1.1)
##  scatterpie             0.1.7     2021-08-20 [1] CRAN (R 4.1.1)
##  sessioninfo            1.1.1     2018-11-05 [1] CRAN (R 4.1.1)
##  sf                     1.0-3     2021-10-07 [1] CRAN (R 4.1.1)
##  shadowtext           * 0.1.0     2021-12-28 [1] local
##  shiny                  1.7.1     2021-10-02 [1] CRAN (R 4.1.1)
##  shinycustomloader      0.9.0     2018-03-27 [1] CRAN (R 4.1.1)
##  shinyWidgets           0.6.2     2021-09-17 [1] CRAN (R 4.1.1)
##  stringi                1.7.5     2021-10-04 [1] CRAN (R 4.1.1)
##  stringr                1.4.0     2019-02-10 [1] CRAN (R 4.1.1)
```

```
##  SummarizedExperiment * 1.24.0    2021-10-26 [1] Bioconductor
##  survival             * 3.2-13    2021-08-24 [1] CRAN (R 4.1.1)
##  svglite                2.0.0     2021-02-20 [1] CRAN (R 4.1.1)
##  systemfonts            1.0.3     2021-10-13 [1] CRAN (R 4.1.1)
##  testthat               3.1.0     2021-10-04 [1] CRAN (R 4.1.1)
##  TH.data                1.1-0     2021-09-27 [1] CRAN (R 4.1.1)
##  tibble                 3.1.6     2021-11-07 [1] CRAN (R 4.1.1)
##  tidybulk             * 1.6.1     2021-10-31 [1] Bioconductor
##  tidygraph              1.2.0     2020-05-12 [1] CRAN (R 4.1.1)
##  tidyr                  1.1.4     2021-09-27 [1] CRAN (R 4.1.1)
##  tidyselect             1.1.1     2021-04-30 [1] CRAN (R 4.1.1)
##  tidytree               0.3.6     2021-11-12 [1] CRAN (R 4.1.1)
##  treeio                 1.18.0    2021-10-26 [1] Bioconductor
##  tweenr                 1.0.2     2021-03-23 [1] CRAN (R 4.1.1)
##  tzdb                   0.1.2     2021-07-20 [1] CRAN (R 4.1.1)
##  units                  0.7-2     2021-06-08 [1] CRAN (R 4.1.1)
##  usethis                2.0.1     2021-02-10 [1] CRAN (R 4.1.1)
##  utf8                   1.2.2     2021-07-24 [1] CRAN (R 4.1.1)
##  vctrs                  0.3.8     2021-04-29 [1] CRAN (R 4.1.1)
##  vegan                  2.5-7     2020-11-28 [1] CRAN (R 4.1.1)
##  viridis                0.6.2     2021-10-13 [1] CRAN (R 4.1.1)
##  viridisLite            0.4.0     2021-04-13 [1] CRAN (R 4.1.1)
##  webshot                0.5.2     2019-11-22 [1] CRAN (R 4.1.1)
##  wget                 * 0.0.1     2021-12-06 [1] local
##  withr                  2.4.2     2021-04-18 [1] CRAN (R 4.1.1)
##  xfun                   0.28      2021-11-04 [1] CRAN (R 4.1.1)
##  xml2                   1.3.2     2020-04-23 [1] CRAN (R 4.1.1)
##  xtable                 1.8-4     2019-04-21 [1] CRAN (R 4.1.1)
##  XVector                0.34.0    2021-10-26 [1] Bioconductor
##  yaml                   2.2.1     2020-02-01 [1] CRAN (R 4.1.1)
##  yulab.utils            0.0.4.901 2021-11-17 [1] Github (YuLab-SMU/yulab.utils@98ec2db)
##  zlibbioc               1.40.0    2021-10-26 [1] Bioconductor
##  zoo                    1.8-9     2021-03-09 [1] CRAN (R 4.1.1)
##
## [1] /mnt/d/UbuntuApps/R/4.1.1/lib/R/library
```

References

Chen, Meijun, and Guangchuang Yu. 2021. *MicrobiomeProfiler: An R/Shiny Package for Microbiome Functional Enrichment Analysis*. https://github.com/YuLab-SMU/MicrobiomeProfiler/.

Douglas, Gavin M., Richard Hansen, Casey M. A. Jones, Katherine A. Dunn, AndrÃ©M. Comeau, Joseph P. Bielawski, Rachel Tayler, et al. 2018. "Multi-Omics Differentially Classify Disease State and Treatment Outcome in Pediatric Crohnâ??S Disease." *Microbiome* 6 (1): 13. https://doi.org/10.1186/s40168-018-0398-3.

Huang, Ruizhu, Charlotte Soneson, Felix G. M. Ernst, Kevin C. Rue-Albrecht, Guangchuang Yu, Stephanie C. Hicks, and Mark D. Robinson. 2021. "TreeSummarizedExperiment: A S4 Class for Data with Hierarchical Structure." *F1000Research* 9: 1246. https://f1000research.com/articles/9-1246.

Mangiola, Stefano, Ramyar Molania, Ruining Dong, Maria A. Doyle, and Anthony T. Papenfuss. 2021. "Tidybulk: An R Tidy Framework for Modular Transcriptomic Data Analysis." *Genome Biology* 22 (1): 42. https://doi.org/10.1186/s13059-020-02233-7.

McMurdie, Paul J., and Joseph N Paulson. 2021. *Biomformat: An Interface Package for the Biom File Format*.

McMurdie, Susan, Paul J. AND Holmes. 2013. "Phyloseq: An R Package for Reproducible Interactive Analysis and Graphics of Microbiome Census Data." *PLOS ONE* 8 (4): 1–11. https://doi.org/10.1371/journal.pone.0061217.

Morgan, Martin, Valerie Obenchain, Jim Hester, and Hervé Pagès. 2021. *SummarizedExperiment: SummarizedExperiment Container*. https://bioconductor.org/packages/SummarizedExperiment.

Oksanen, Jari, F. Guillaume Blanchet, Michael Friendly, Roeland Kindt, Pierre Legendre, Dan McGlinn, Peter R. Minchin, et al. 2020. *Vegan: Community Ecology Package*. https://CRAN.R-project.org/package=vegan.

REISKIND, M. H., R. H. GRIFFIN, M. S. JANAIRO, and K. A. HOPPERSTAD. 2017. "Mosquitoes of Field and Forest: The Scale of Habitat Segregation in a Diverse Mosquito Assemblage." *Medical and Veterinary Entomology* 31 (1): 44–54. https://doi.org/https://doi.org/10.1111/mve.12193.

Research Network Consortium, Integrative HMP (iHMP). 2014. "The Integrative Human Microbiome Project: Dynamic Analysis of Microbiome-Host Omics Profiles During Periods of Human Health and Disease." *Cell Host & Microbe* 16 (3): 276–89. https://doi.org/10.1016/j.chom.2014.08.014.

Robinson, Mark D, Davis J McCarthy, and Gordon K Smyth. 2010. "EdgeR: A Bioconductor Package for Differential Expression Analysis of Digital Gene Expression Data." *Bioinformatics* 26 (1): 139–40. https://doi.org/10.1093/bioinformatics/btp616.

Segata, Nicola, Levi Waldron, Annalisa Ballarini, Vagheesh Narasimhan, Olivier Jousson, and Curtis Huttenhower. 2012. "Metagenomic Microbial Community Profiling Using Unique Clade-Specific Marker Genes." *Nature Methods* 9 (8): 811–14. https://doi.org/10.1038/nmeth.2066.

Wickham, Hadley. 2011. "Ggplot2." *WIREs Computational Statistics* 3 (2): 180–85. https://doi.org/https://doi.org/10.1002/wics.147.

Wu, Tianzhi, Erqiang Hu, Shuangbin Xu, Meijun Chen, Pingfan Guo, Zehan Dai, Tingze Feng, et al. 2021. "ClusterProfiler 4.0: A Universal Enrichment Tool for Interpreting Omics Data." *The Innovation* 2 (3): 100141. https://doi.org/10.1016/j.xinn.2021.100141.

Xu, Shuangbin, Zehan Dai, Pingfan Guo, Xiaocong Fu, Shanshan Liu, Lang Zhou, Wenli Tang, et al. 2021. "ggtreeExtra: Compact Visualization of Richly Annotated Phylogenetic Data." *Molecular Biology and Evolution* 38 (9): 4039–42. https://doi.org/10.1093/molbev/msab166.

Yu, Guangchuang, David K. Smith, Huachen Zhu, Yi Guan, and Tommy Tsan-Yuk Lam. 2017. "Ggtree: An R Package for Visualization and Annotation of Phylogenetic Trees with Their Covariates and Other Associated Data." *Methods in Ecology and Evolution* 8 (1): 28–36. https://doi.org/https://doi.org/10.1111/2041-210X.12628.

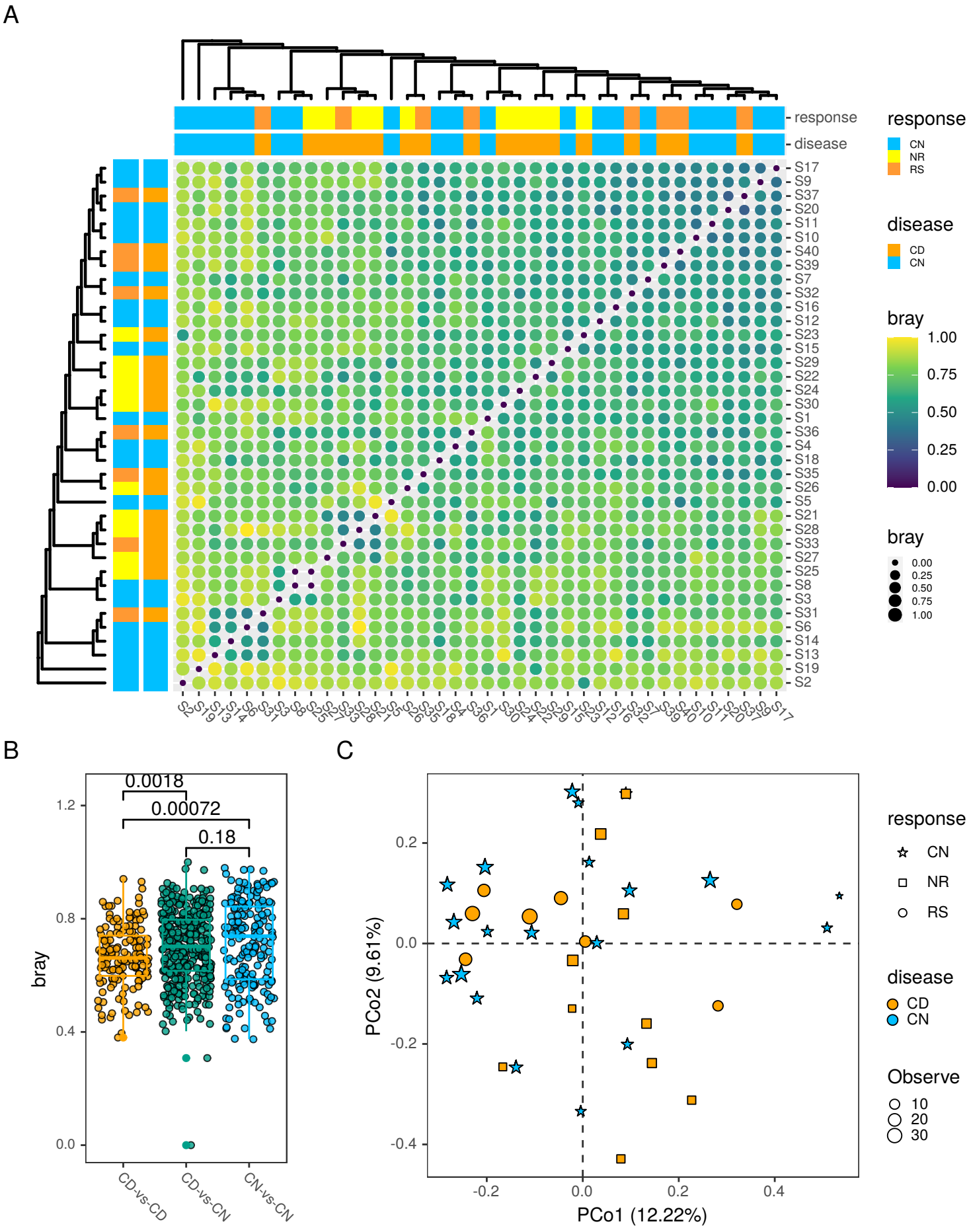

Fig. S19: **The distance heatmap and boxplot and the PCoA plot based on MGS data**

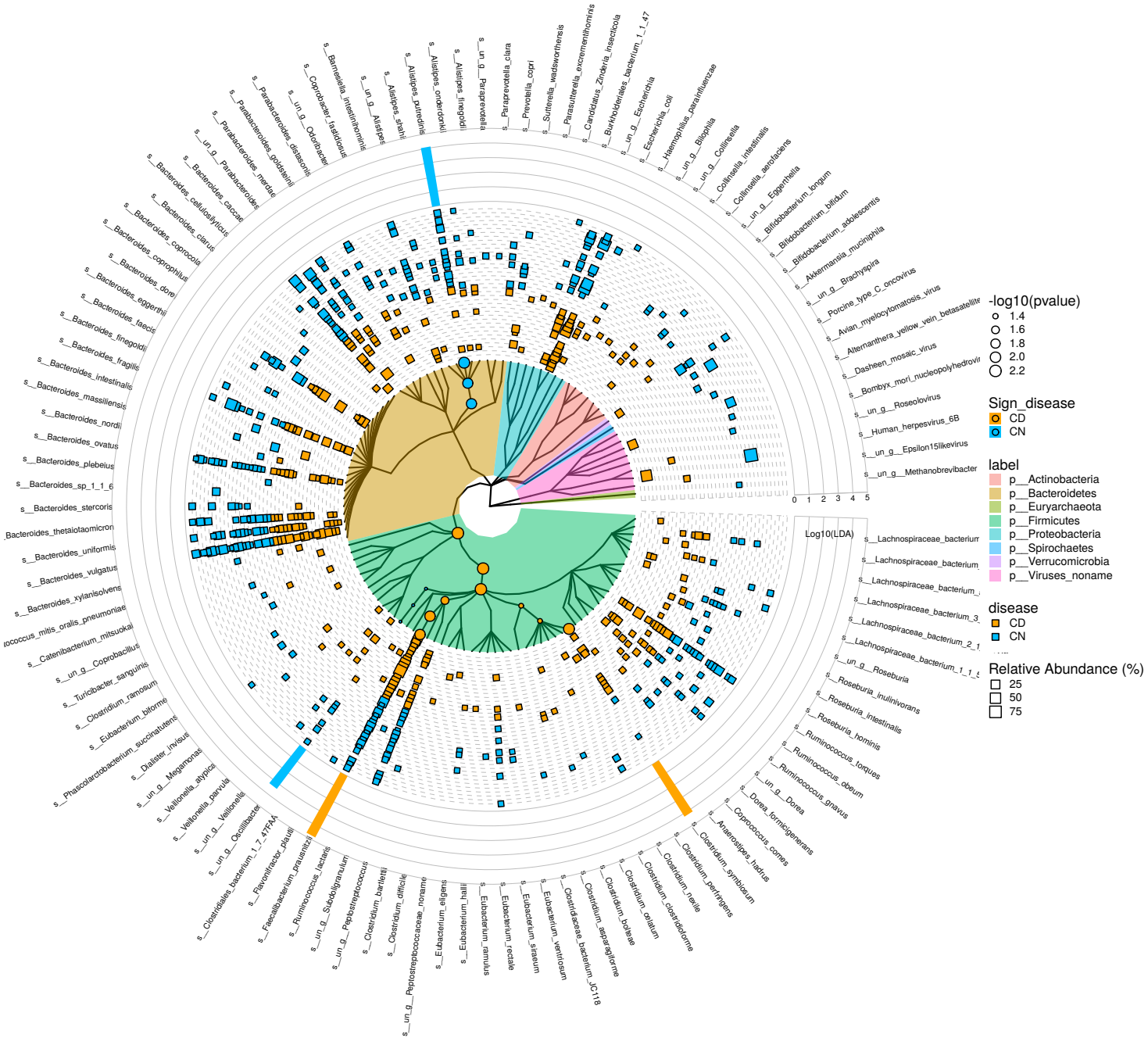

Fig. S20: **The result of different analysis based on MGS data**

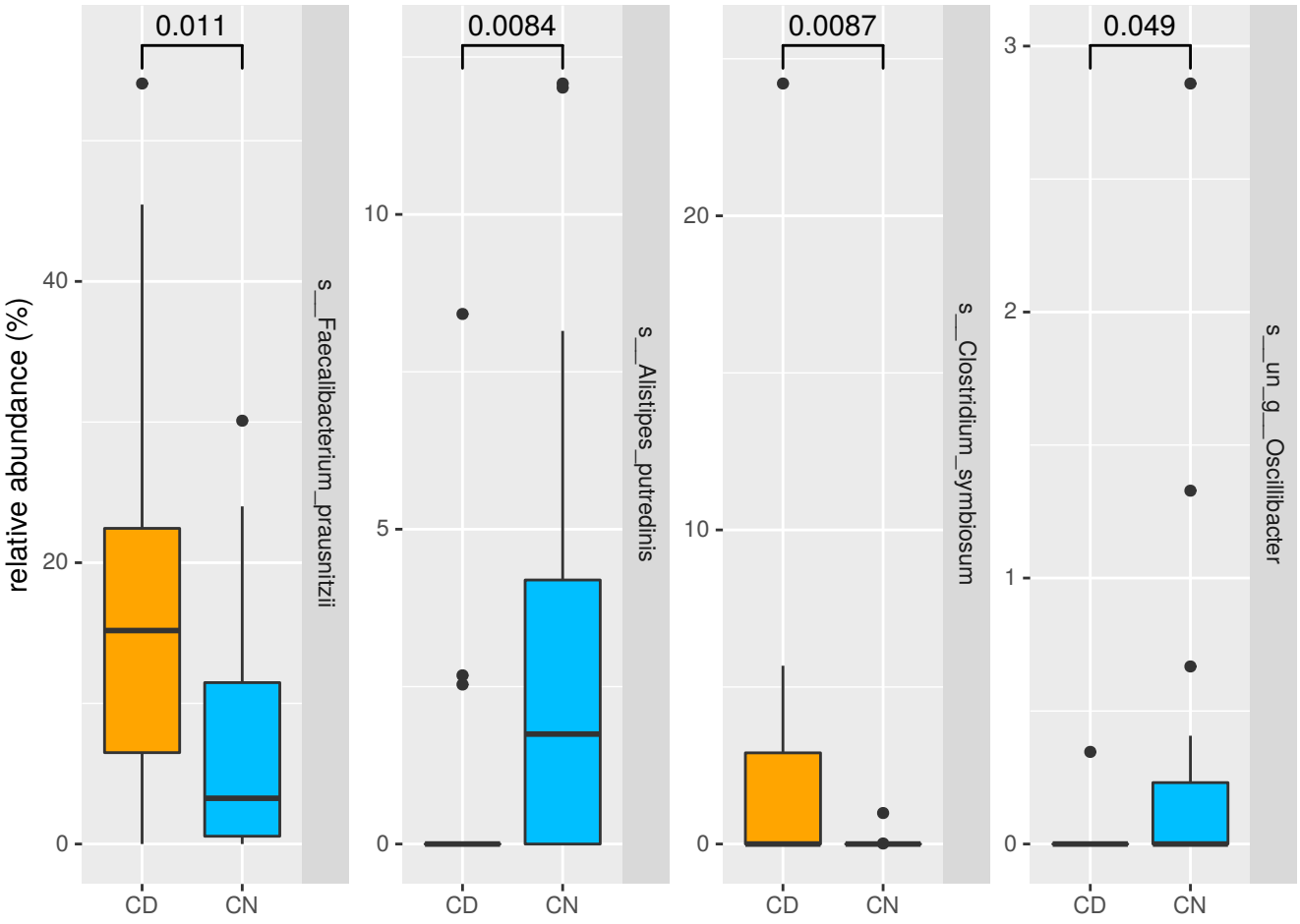

Fig. S21: **The abundance boxplot of the different species between the CD and control group**

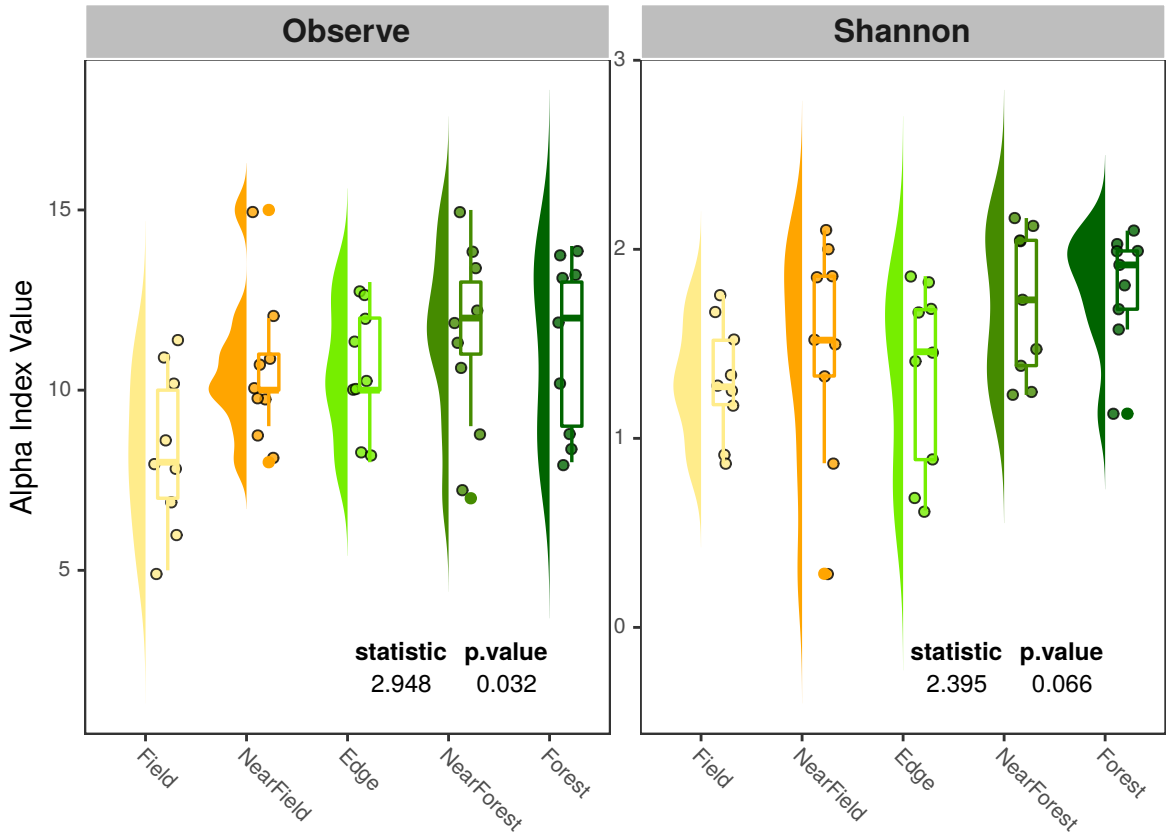

Fig. S22: **The raincloud plot of the alpha diversity of the Mosquito ecology community.** The result of the alpha diversity analysis about the Mosquito ecology study showed that the Mosquito species richness gradually increases from field to forest (field --> near field --> edge --> near field --> forest).

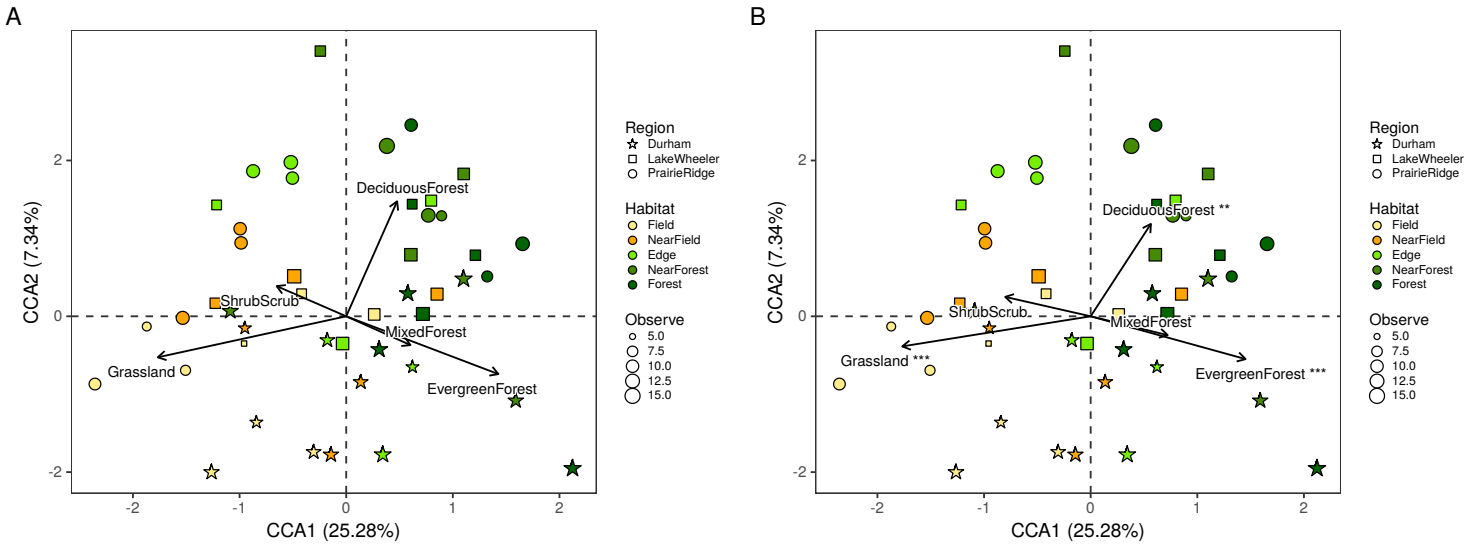

Fig. S23: **The CCA plot of the Mosquito ecology study**. Each point represents one sample, the size of the points represents the observe species of the corresponding sample, the color of the points represents the habitat of the corresponding sample, the shape of points represents the Region of the corresponding sample. And the arrows represent the environment factors, the marked ones by star represent significant related to the Mosquito communities in the study (* 0.05, ** 0.01, *** 0.001).

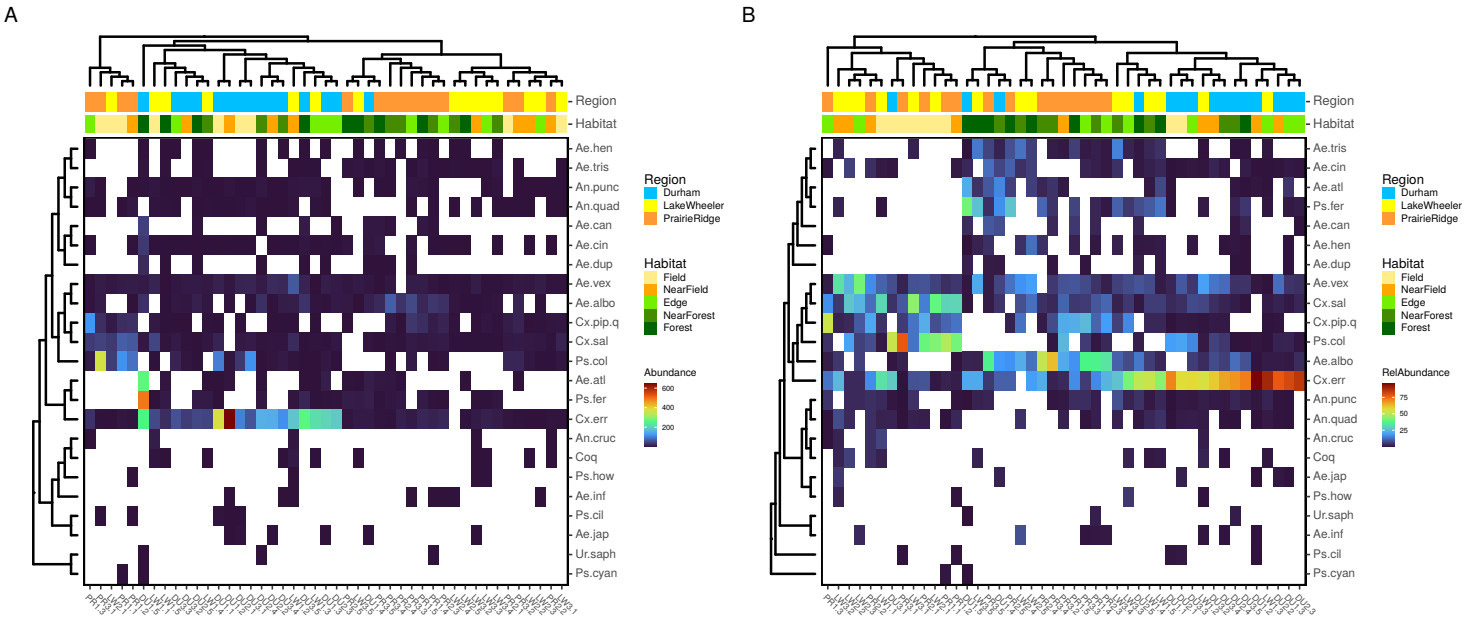

Fig. S24: **The heatmap of the abundance and relative abundance of the Mosquito species.**