Supplemental File A of

MicrobiotaProcess: A comprehensive R package for deep mining microbiome

Shuangbin Xu, Li Zhan, Wenli Tang, Qianwen Wang, Zehan Dai, Lang Zhou, Tingze Feng, Meijun Chen, 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:

```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.

```r
cranpkgs <- c("aplot", "ggpp", "igraph",
              "broom", "forcats", 'pROC',
              "ggrepel", "ggVennDiagram",
              "patchwork", "shadowtext",
              "ggupset", "ggnewscale",
              "GUniFrac", "matrixStats")

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

Biocpkgs <- c("SummarizedExperiment", "clusterProfiler",
              "edgeR", "enrichplot", "tidybulk", "curatedMetagenomicData",
              "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 re-analyzed the 16s rDNA dataset of 43 pediatric IBD stool samples, 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 the web[1]. These datasets contain `ibd_asv_table.txt` (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 used *mp_import_dada2* of *MicrobiotaProcess* to import the dataset, and returned an *MPSE* object.

```r
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)),]
ref.tree <- treeio::read.tree('./data/IBD_data/ibd_repseq.tree')
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)
otutree(mpse2) <- ref.tree
mpse2
```

```
## # A MPSE-tibble (MPSE object) abstraction: 9,890 x 11
## # OTU=230 | Samples=43 | Assays=Abundance | Taxonomy=Kingdom, Phylum, Class, Order, Family, Genus, Species
##    OTU     Sample  Abund~1 Group Kingdom Phylum Class Order Family Genus Species
##    <chr>   <chr>     <int> <chr> <chr>   <chr>  <chr> <chr> <chr>  <chr> <chr>
##  1 OTU_215 S206700       0 CD    k__Bac~ p__Ac~ c__A~ o__A~ f__Ac~ g__A~ s__un_~
##  2 OTU_522 S206700       0 CD    k__Bac~ p__Ac~ c__A~ o__A~ f__Ac~ g__A~ s__un_~
##  3 OTU_719 S206700       0 CD    k__Bac~ p__Ac~ c__A~ o__A~ f__Mi~ g__R~ s__muc~
##  4 OTU_42  S206700       0 CD    k__Bac~ p__Ac~ c__A~ o__B~ f__Bi~ g__B~ s__ado~
##  5 OTU_120 S206700       0 CD    k__Bac~ p__Ac~ c__A~ o__B~ f__Bi~ g__B~ s__un_~
##  6 OTU_138 S206700       0 CD    k__Bac~ p__Ac~ c__A~ o__B~ f__Bi~ g__B~ s__un_~
##  7 OTU_333 S206700       0 CD    k__Bac~ p__Ac~ c__C~ o__C~ f__Co~ g__A~ s__un_~
##  8 OTU_141 S206700       0 CD    k__Bac~ p__Ac~ c__C~ o__C~ f__Co~ g__C~ s__aer~
##  9 OTU_322 S206700       0 CD    k__Bac~ p__Ac~ c__C~ o__C~ f__Co~ g__E~ s__len~
## 10 OTU_117 S206700       0 CD    k__Bac~ p__Ba~ c__B~ o__B~ f__[O~ g__O~ s__un_~
## # ... with 9,880 more rows, and abbreviated variable name 1: Abundance
```

2.2 Other import functions

MicrobiotaProcess also presents some other functions SA.1 to parse the output of the upstream pipelines. In addition, some common objects 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*) referring to session 3.1.

Table SA.1: List of import functions provided by MicrobiotaProcess

Package	Import Function	Description
	mp_import_qiime2	Import function to load the output of qiime2
MicrobiotaProcess	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 the sampling technique was used to compensate for the effect of sample size on the number of units observed in a sample. *MicrobiotaProcess* provides *mp_cal_rarecurve* and *mp_plot_rarecurve* to calculate and plot the curves.

```r
library(MicrobiotaProcess)
library(patchwork)
cols <- c('#fcc751ff', '#00c7bfff')
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")
```

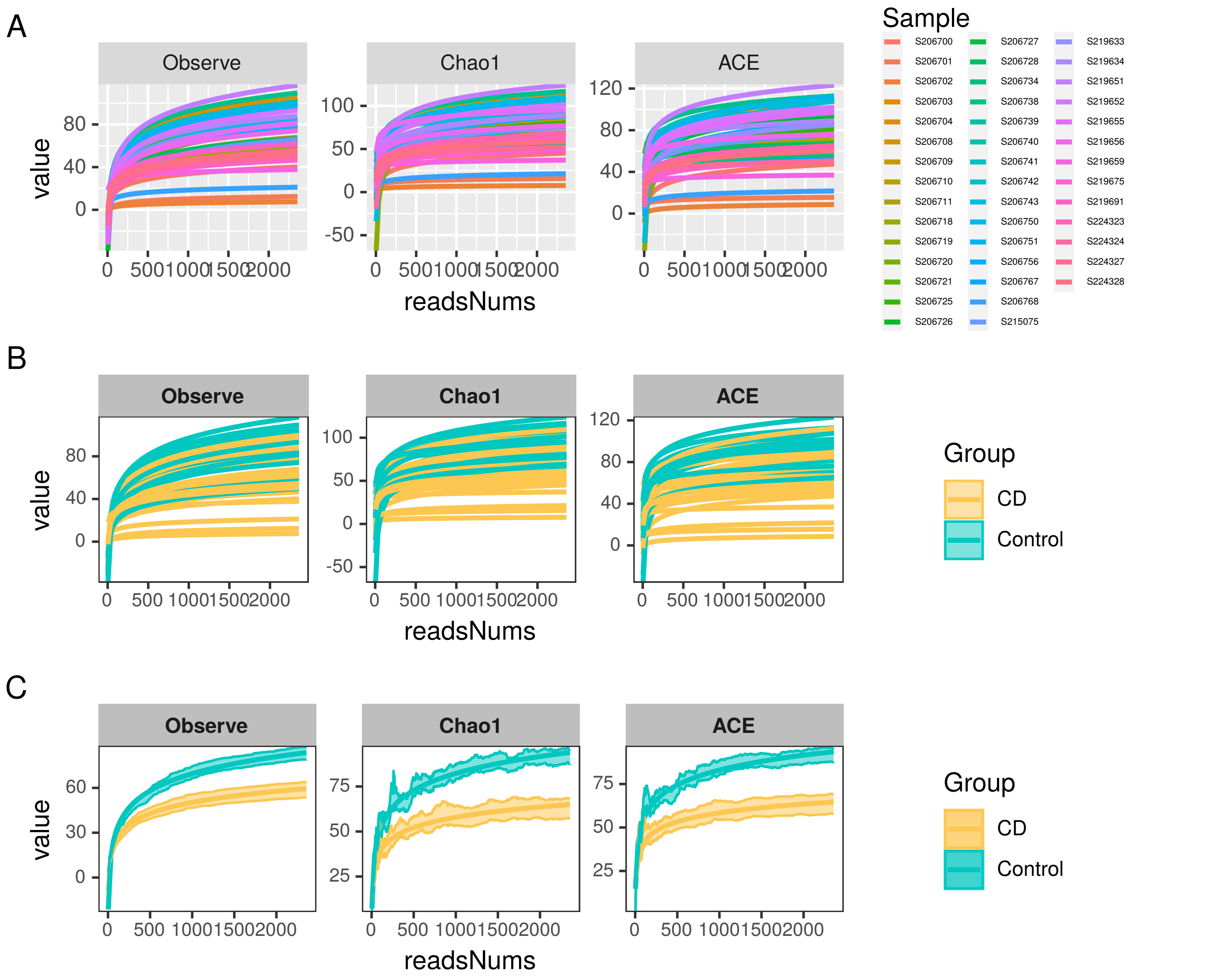


Fig. SA.1: **This example shows *mp_cal_rarecurve* and *mp_plot_rarecurve* provided by *MicrobiotaProcess* to calculate and visualize the rarefaction curve.** The horizontal coordinate represents the sequencing depth of samples, the vertical coordinate shows the Alpha diversity index (such as Observe 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*)

2.3.2 Calculation and different analysis of alpha diversity

Alpha diversity can evaluate the richness and evenness of microbial communities. *MicrobiotaProcess* provides *mp_cal_alpha* to calculate alpha index. Six common diversity measures (*Observe*, *Chao1*, *ACE*, *Shannon*, *Simpson*, *Pielou*) are supported. In addition, *MicrobiotaProcess* also provided *mp_cal_pd_metric* to calculate some phylogenetic community structure metrics, such as PD (Faith's Phylogenetic Diversity), NRI (Nearest Relative Index), NTI (Nearest Taxon Index), IAC (Relative deviation from null expectation of phylogenetically balanced abundances), PAE (Phylogenetic evenness of the abundance distribution scaled by branch lengths), HAED (Entropic measure of diversity of evolutionary distinctiveness among individuals), EAED (Equitability of HAED) (Webb 2000; Cadotte et al. 2010). These phylogenetic metrics can help us to explore the process of microbiota community assembly (Cadotte et al. 2010). The result can be visualized by *mp_plot_alpha*. The following example showed how to use *mp_cal_alpha* and *mp_plot_alpha* of *MicrobiotaProcess* to analyze the alpha diversity of the community. The *RareAbundance* is rarefied (default), which will be used to calculate the alpha diversity index, users can specify the *force=TRUE* of *mp_cal_alpha* to calculated the alpha diversity if the abundance can not be rarefied (referring to session 3.3.1).

```
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
```

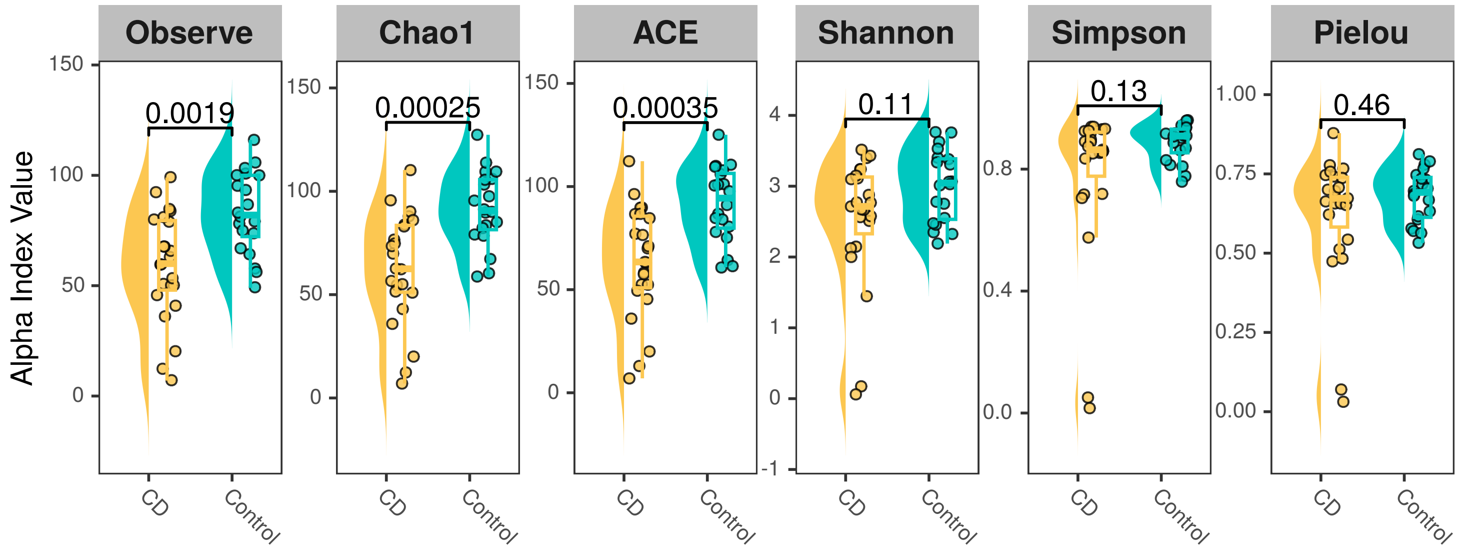


Fig. SA.2: **The raincloud plot of alpha diversity index** The horizontal coordinate represents each group (by *.group* argument of *mp_plot_alpha*), the vertical coordinate represents the alpha diversity index.

```
mpse2 %<>% mp_cal_pd_metric(.abundance = RareAbundance, metric = all)
p.pd_alpha <- mpse2 %>%
         mp_plot_alpha(
            .alpha = c("PAE", "NRI", "NTI", "PD", "HAED", "EAED", "IAC"),
            .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.pd_alpha
```

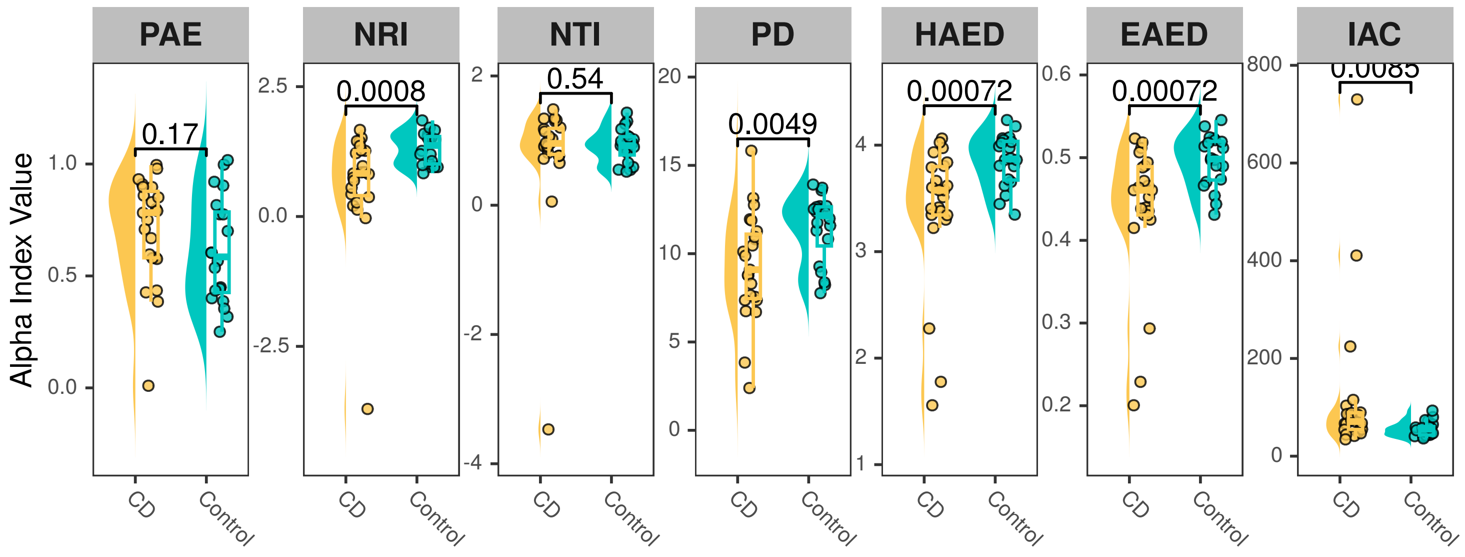


Fig. SA.3: The raincloud plot of phylogenetic diversity index. The horizontal coordinate represents each group (by *.group* argument of *mp_plot_alpha*), the vertical coordinate represents the phylogenetic diversity index.

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 the composition of microbial communities. After the *mp_cal_abundance* is done, you can get the abundance of specific levels of the class by *mp_extract_abundance* (referring to session 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
```

The relative abundance of different 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

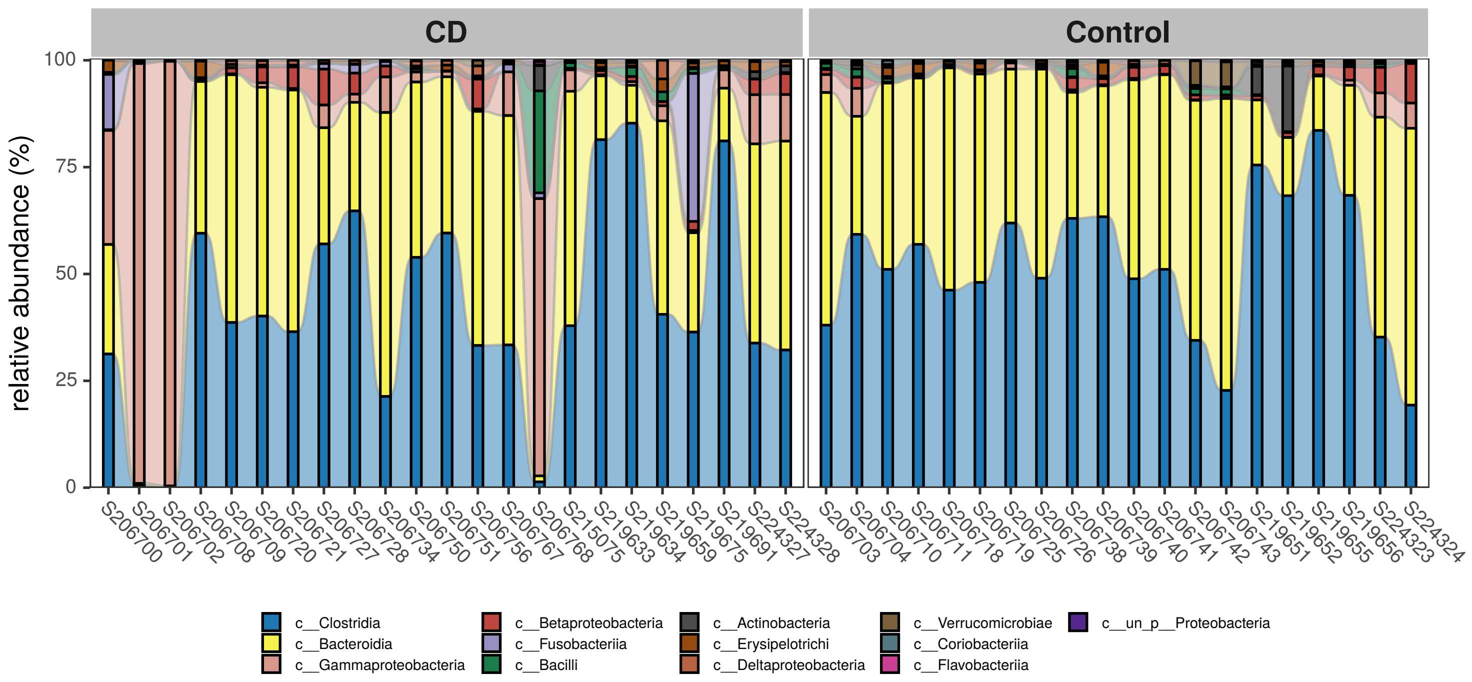


Fig. SA.4: **The relative abundance of each sample in *class* level**

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")
```

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

```
hclass1 <- mpse2 %>%
          mp_plot_abundance(
```

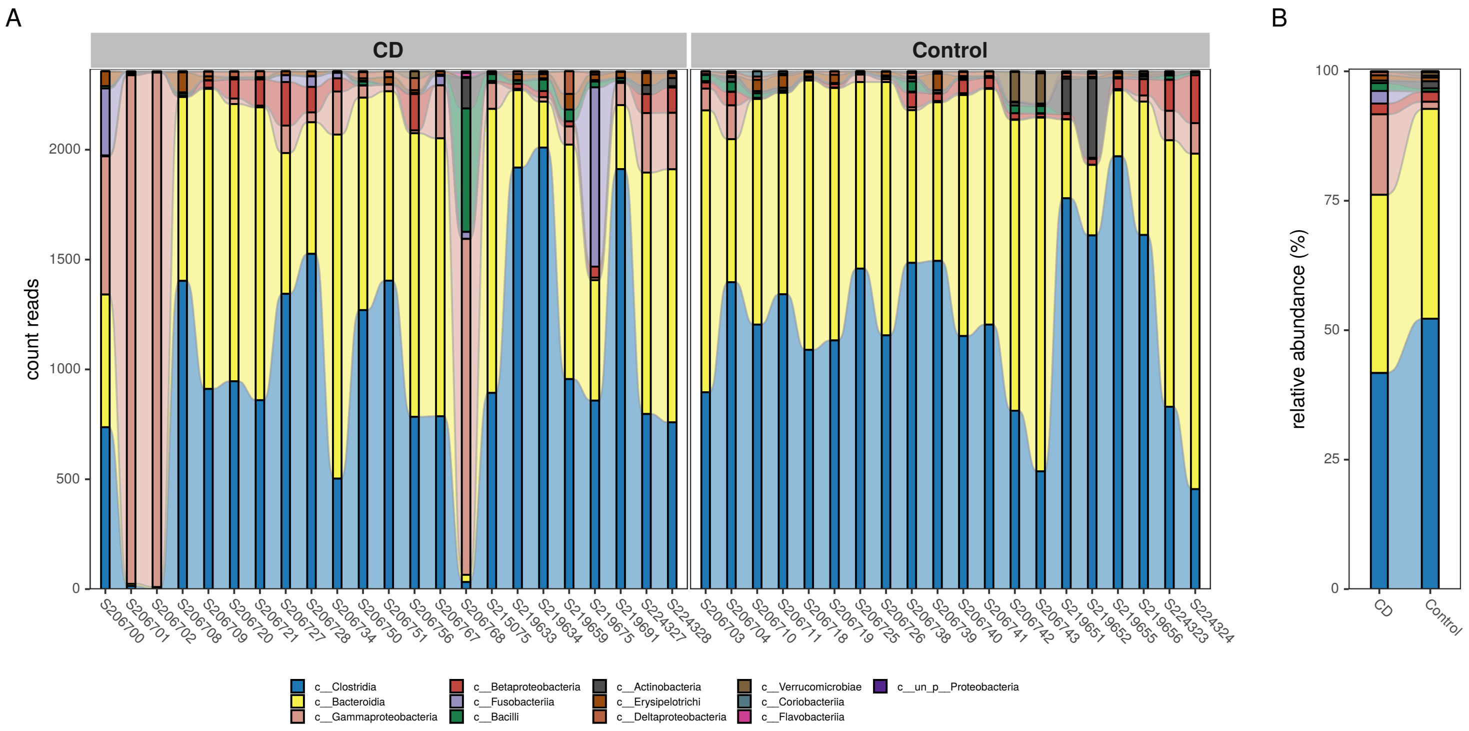


Fig. SA.5: **This example show how to displayed the abundance (count or other) of sample and the relative abundance of groups**. The relative abundance of group (A) and the abundance (count by rarefied) of each sample (B), these results show the *Gammaproteobacteria* of *CD* group might be more abundant than the *control* group.

```
            .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
```

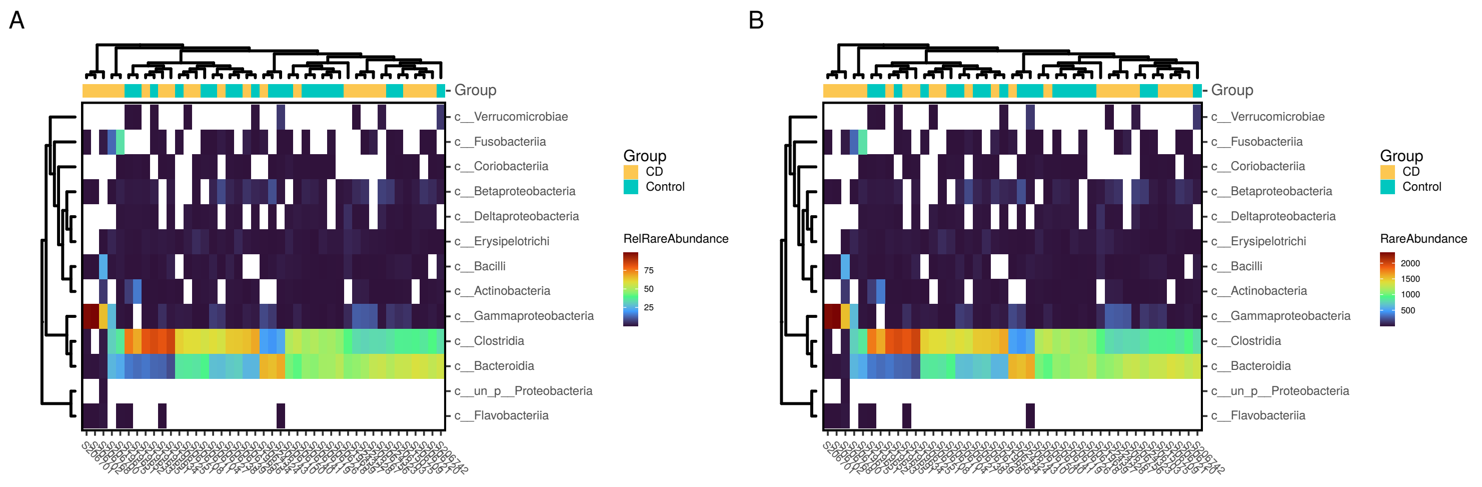


Fig. SA.6: **The heatmap of abundance for each sample at *class* level.** The color (continuous) of heatmap represents the abundance of different classes, the color of bar plot represents the group name of sample, the horizontal coordinate represents the sample, and the vertical coordinate represents the different classes.

2.4.2 Venn or Upset plot

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

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

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

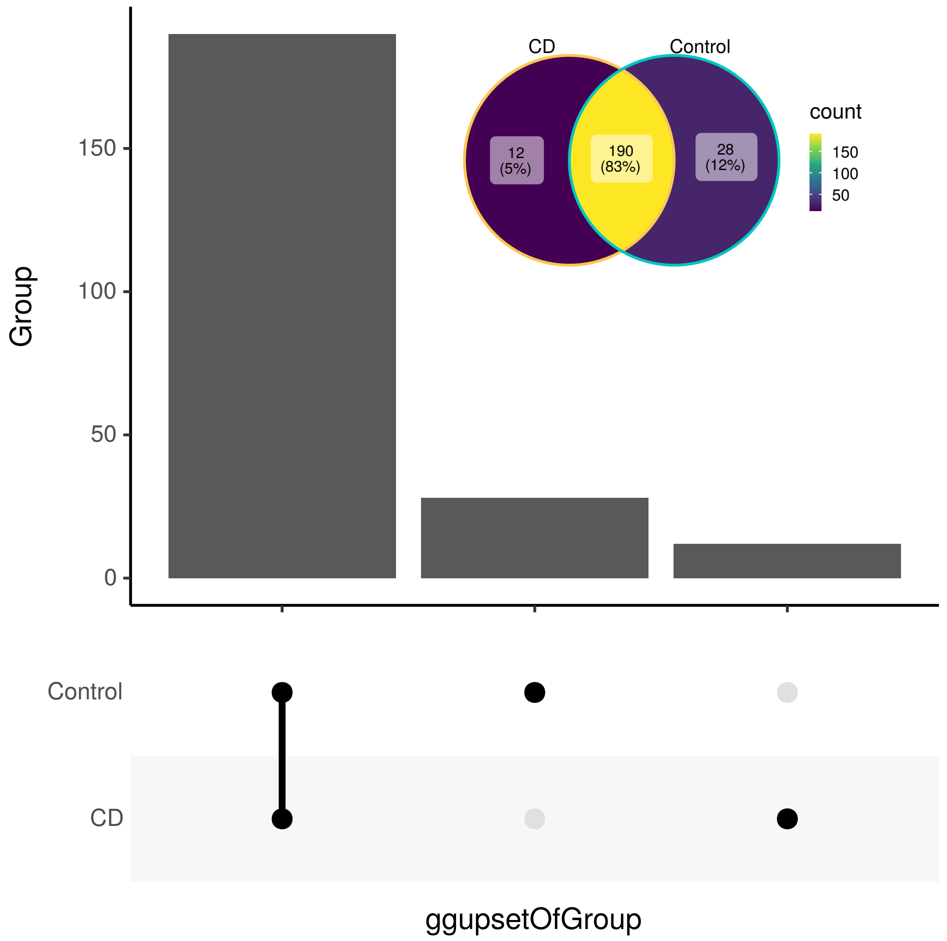


Fig. SA.7: **The Venn diagram and upset plot for groups in OTU/ASV level**

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 related 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")
```

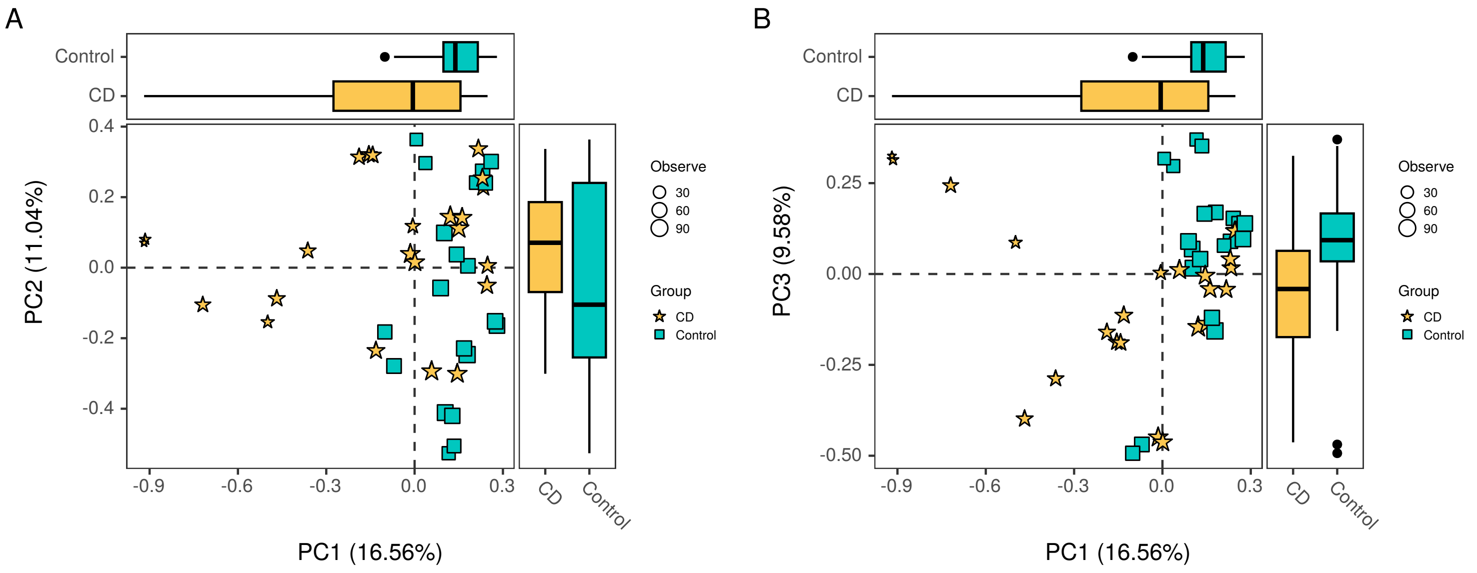


Fig. SA.8: **The PCA plot of the community**. Each point represents one sample, the size of point represents the observe OTU of the sample. The color of point represents the group name of the sample, based on the first and second component (A), based on the first and third component (B).

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,
               .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")
```

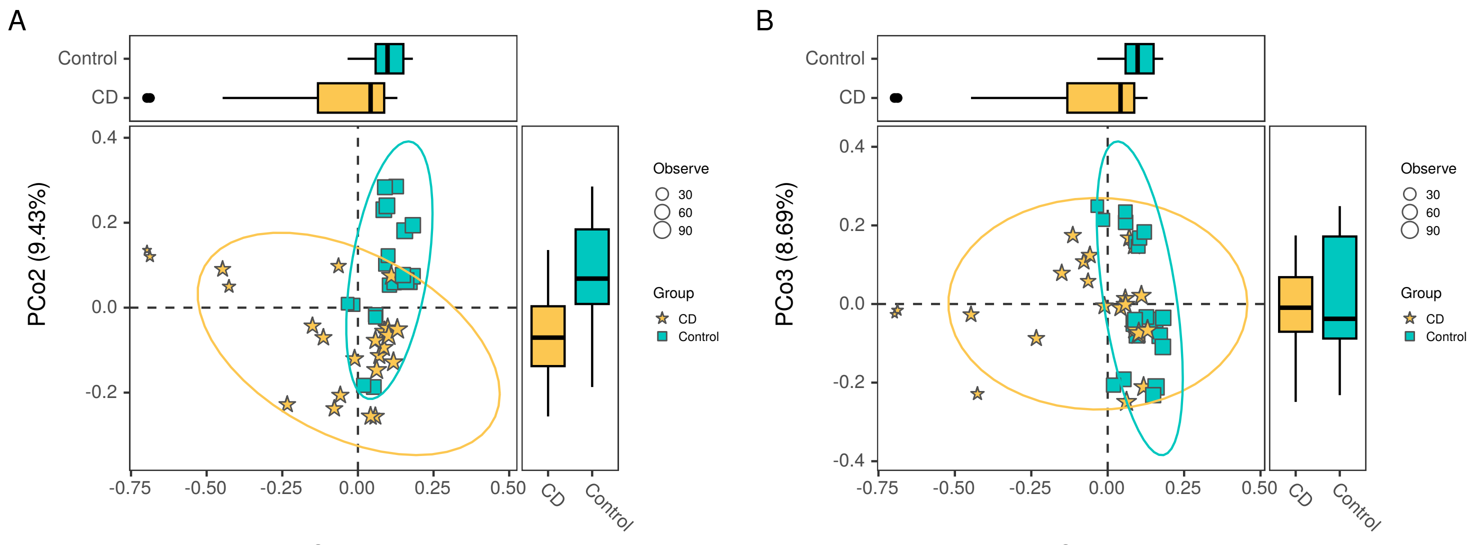


Fig. SA.9: **The PCoA plot based on Bray-Curtis distance**.

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
          )
```

```r
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")
```

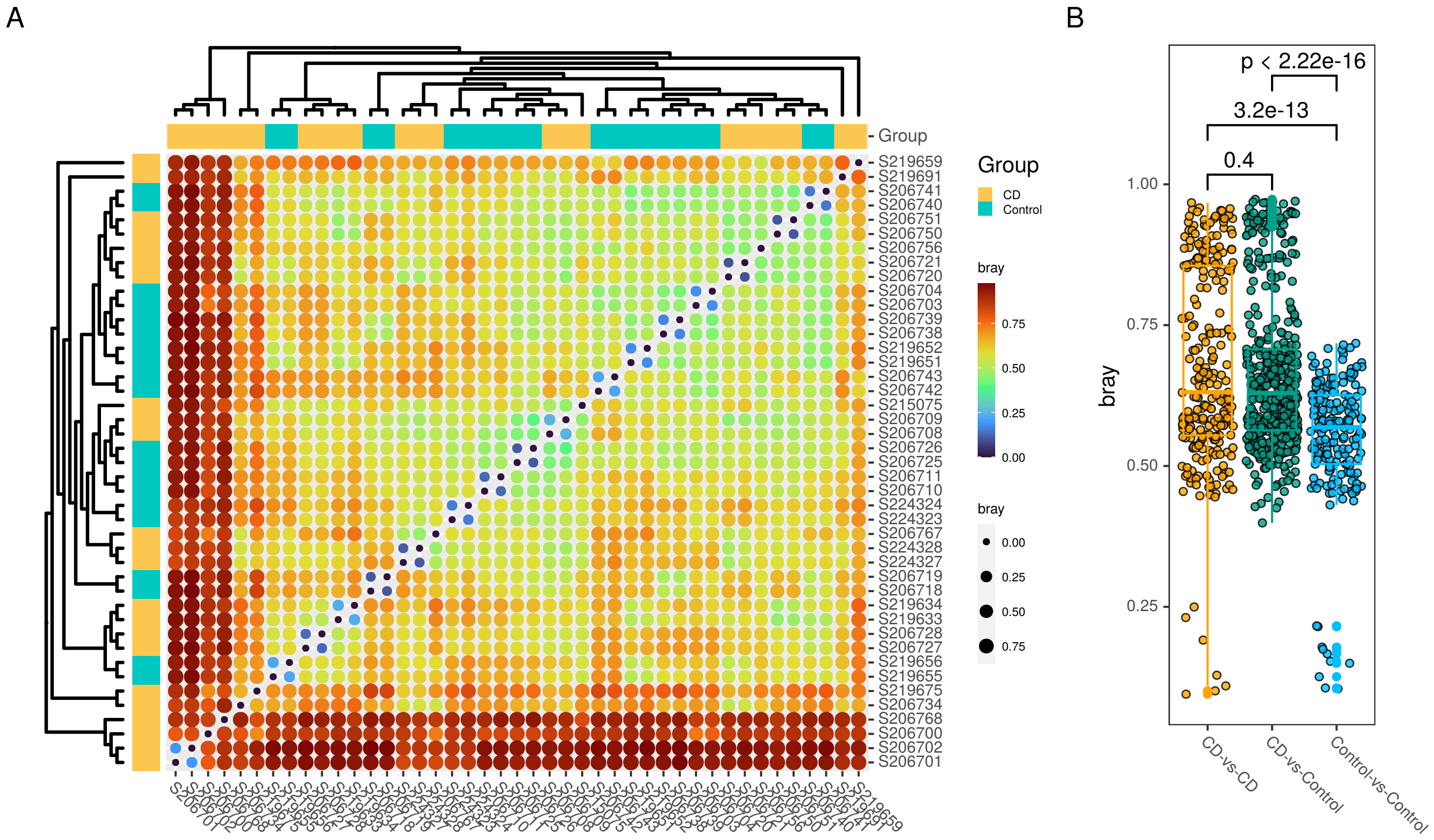


Fig. SA.10: **The distance heatmap and the boxplot for each sample**. The size and color of the heatmap represent the distance of each sample, and color of bar plot represents the group of sample (A). The boxplot represents the distance pairs of sample among the group, it shows the dissimilarity of the sample between the *control* and *CD* is significant, which is consistent with the result of the Permutational Multivariate Analysis of Variance in session 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 6
##   factors     Df SumOfSqs     R2     F `Pr(>F)`
##   <chr>    <dbl>    <dbl>  <dbl> <dbl>    <dbl>
## 1 Group        1    0.789 0.0864  3.88   0.0001
## 2 Residual    41    8.34  0.914  NA     NA
## 3 Total       42    9.12  1      NA     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`, it also can 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
```

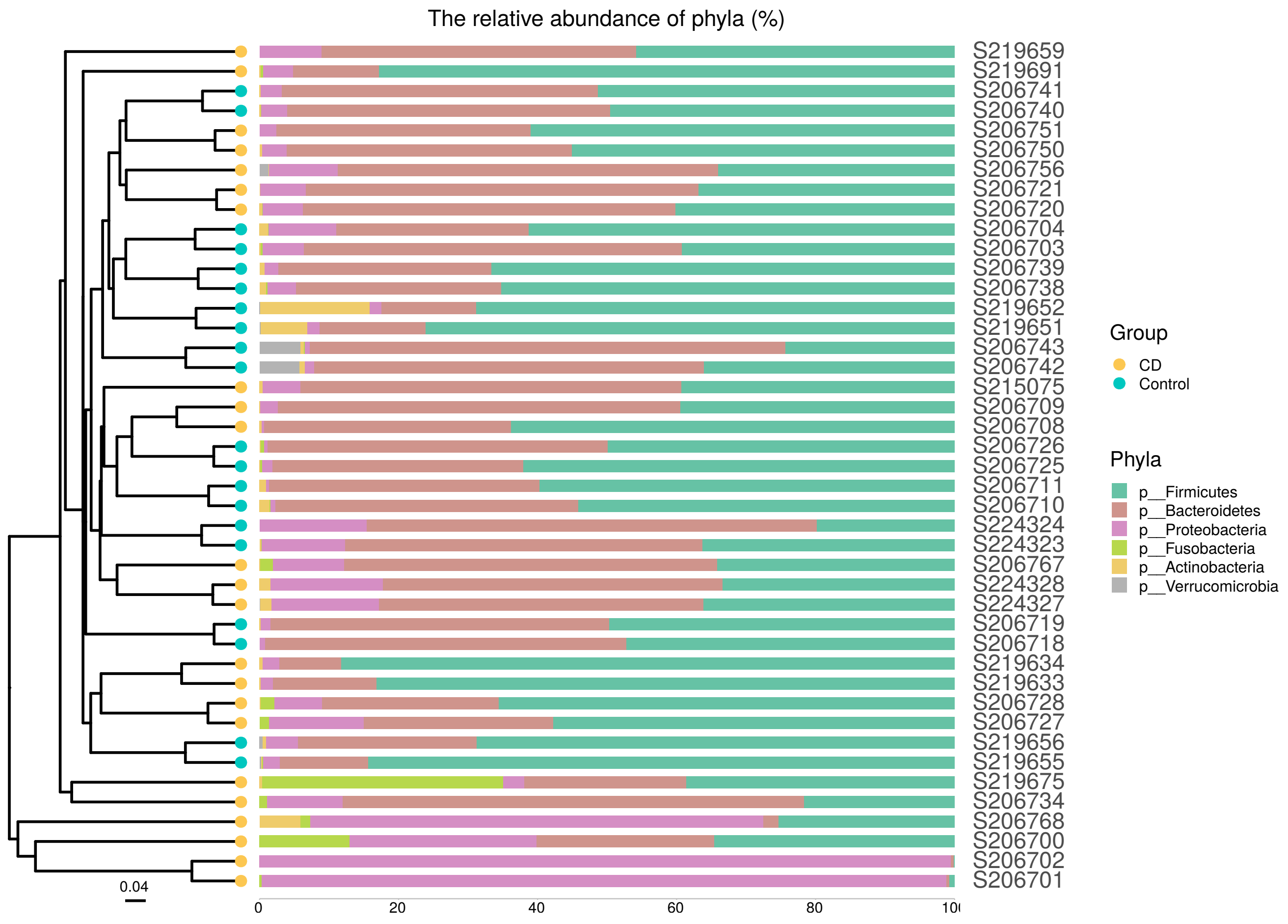


Fig. SA.11: **The hierarchical clustering plot of samples based on Bray-Curtis distance calculated with abundance of OTU/ASV and the relative Abundance of phyla for samples**

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 mp_plot_diff_cladogram* (with `action = "add"`), or displayed manually using `ggtree` (Yu et al. 2017) and `ggtreeExtra` (Xu et al. 2021).

```r
# 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"
         )
# The result of different analysis was added to the taxatree with action = 'add'
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"
         )

p.clado <- mpse2 %>%
   mp_plot_diff_cladogram(
     taxa.class = Order,
     removeUnknown = TRUE,
     as.tiplab = TRUE,
     tip.annot = TRUE,
     label.size = 2.2
   ) +
   scale_fill_diff_cladogram(values=cols)
p.clado
```

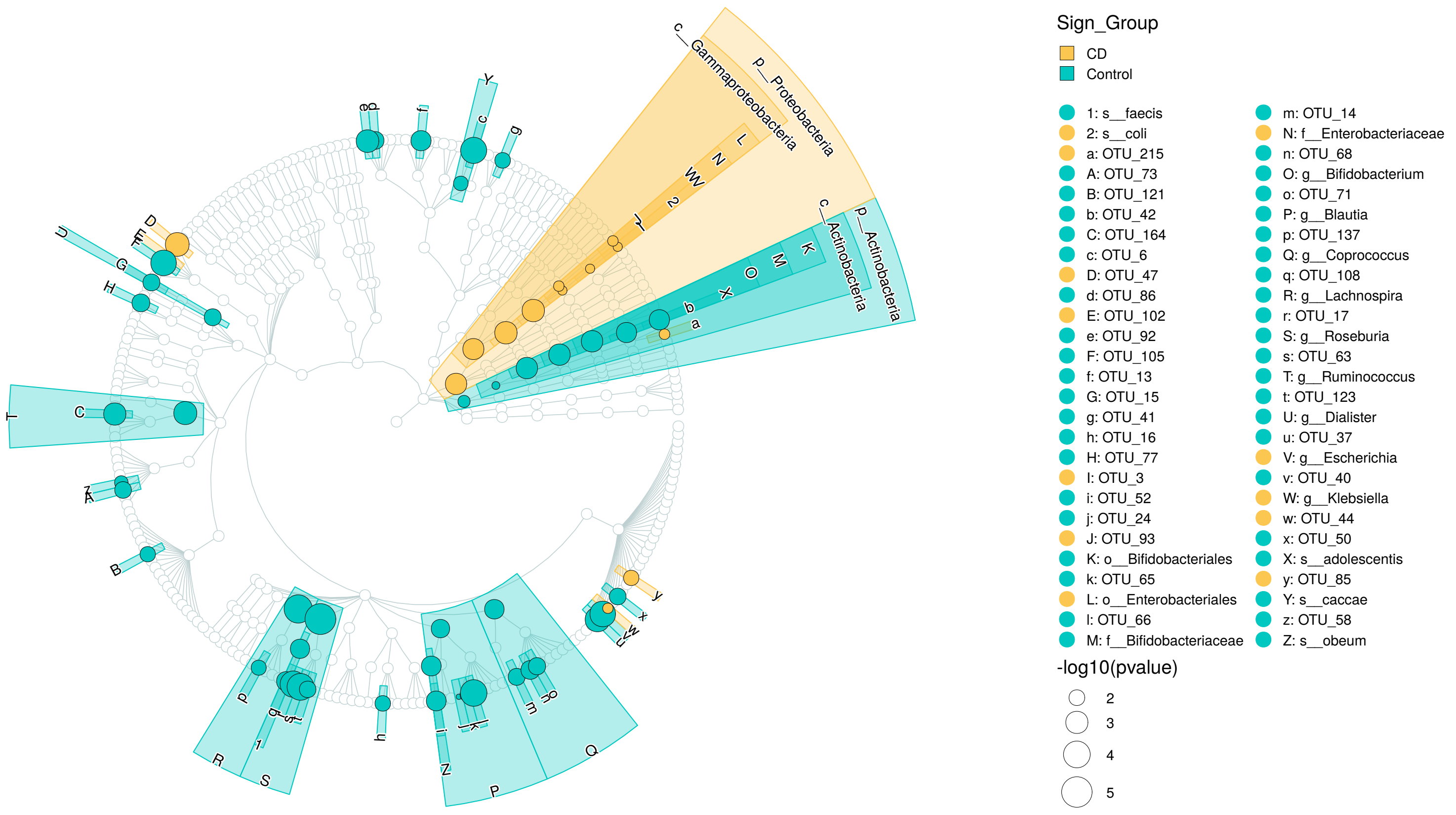


Fig. SA.12: **The cladogram of significant differential taxa between CD group and Control group** The hight light clades represent the differential taxa is enriched in the corresponding group. We found the species from Proteobacteria were enriched in the CD group, the species (OTU_42) from Actinobacteria were enriched in Control group.

2.6.1 visualization of different results by `ggdiffclade`

The color of discriminative taxa represents the taxa is more abundant in the corresponding group. The point size shows the negative logarithms (base 10) of the pvalue. The bigger size of the 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 = ggplot2::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
```

We also can visualized the result (default, with `action = 'add'`) via `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 = 0.01,
         offset = 0.04,
         pwidth = 1.5,
         grid.params = list(vline = TRUE, size = 0.001, color="snow2", linetype = 1)
      ) +
      scale_size_continuous(
         name="Relative Abundance (%)",
         range = c(0.5, 3),
         guide = guide_legend(override.aes = list(starstroke = 0.25))
      ) +
      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(
```

```r
                       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(Sign_Group)),
         mapping = aes(size = -log10(fdr),
                       fill = Sign_Group,
                       ),
         stroke = 0.01,
         shape = 21,
      ) +
      scale_size_continuous(range=c(1, 3), guide = guide_legend(override.aes = list(stroke = .25))) +
      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
```

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

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

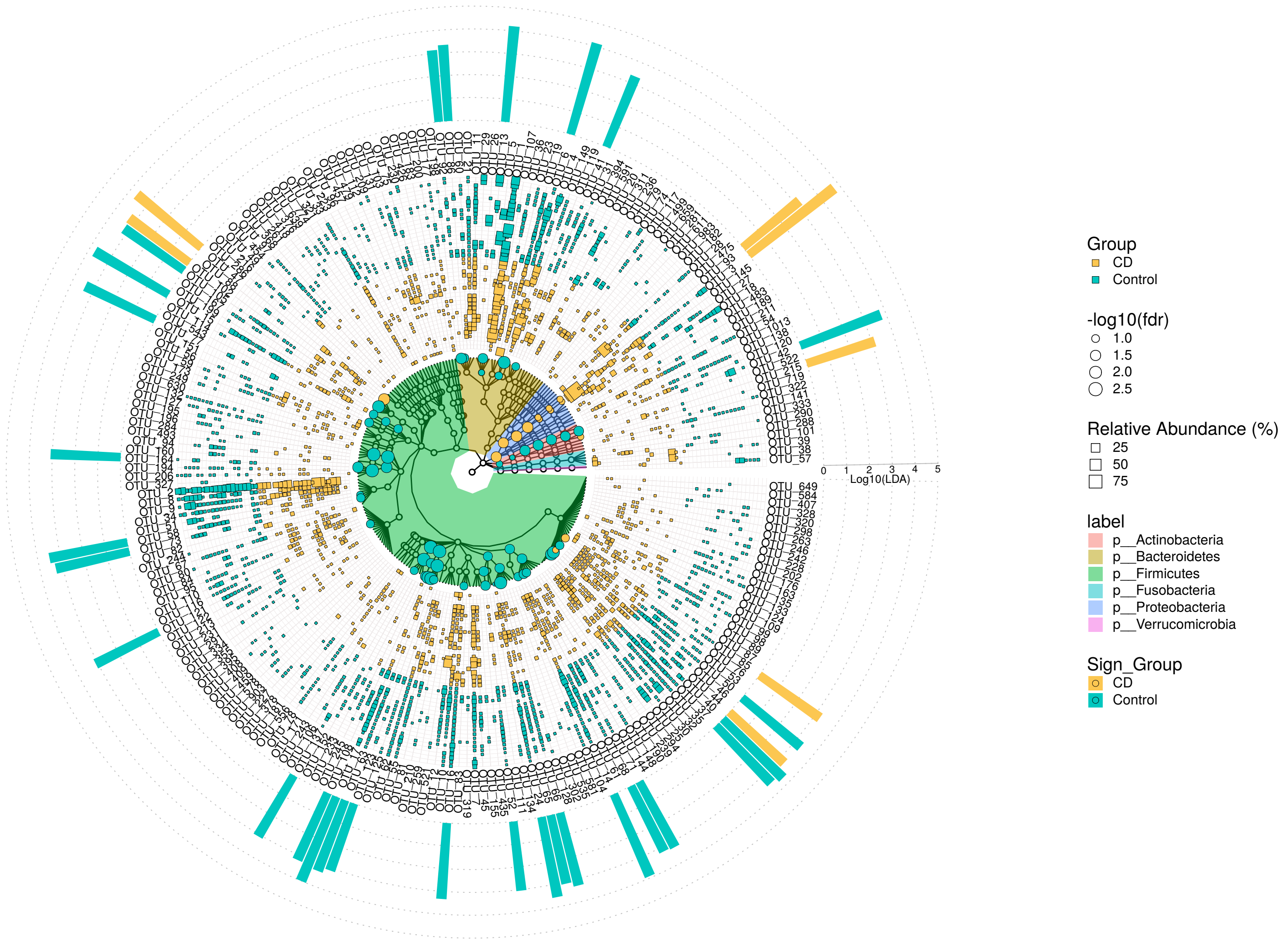


Fig. SA.13: **The taxa tree of the community with the relative abundance of each OTU/ASV on sample and the LDA of different OTU/ASV**. The taxa tree was built with the taxa of all samples. The high light clades of taxa tree represented the phyla. The external point layer represented the relative abundance of each OTU on sample. The external bar plot represented the LDA of the different OTU. The colored points represented the different taxa, the size of colored point represented the *pvalue* or *fdr*.

2.6.2 visualization of differential results (with action = "get") by `ggdiffbox`

The left panel represented the relative abundance or abundance (according the standard_method) of biomarker, the right panel represented 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 sampling times.

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

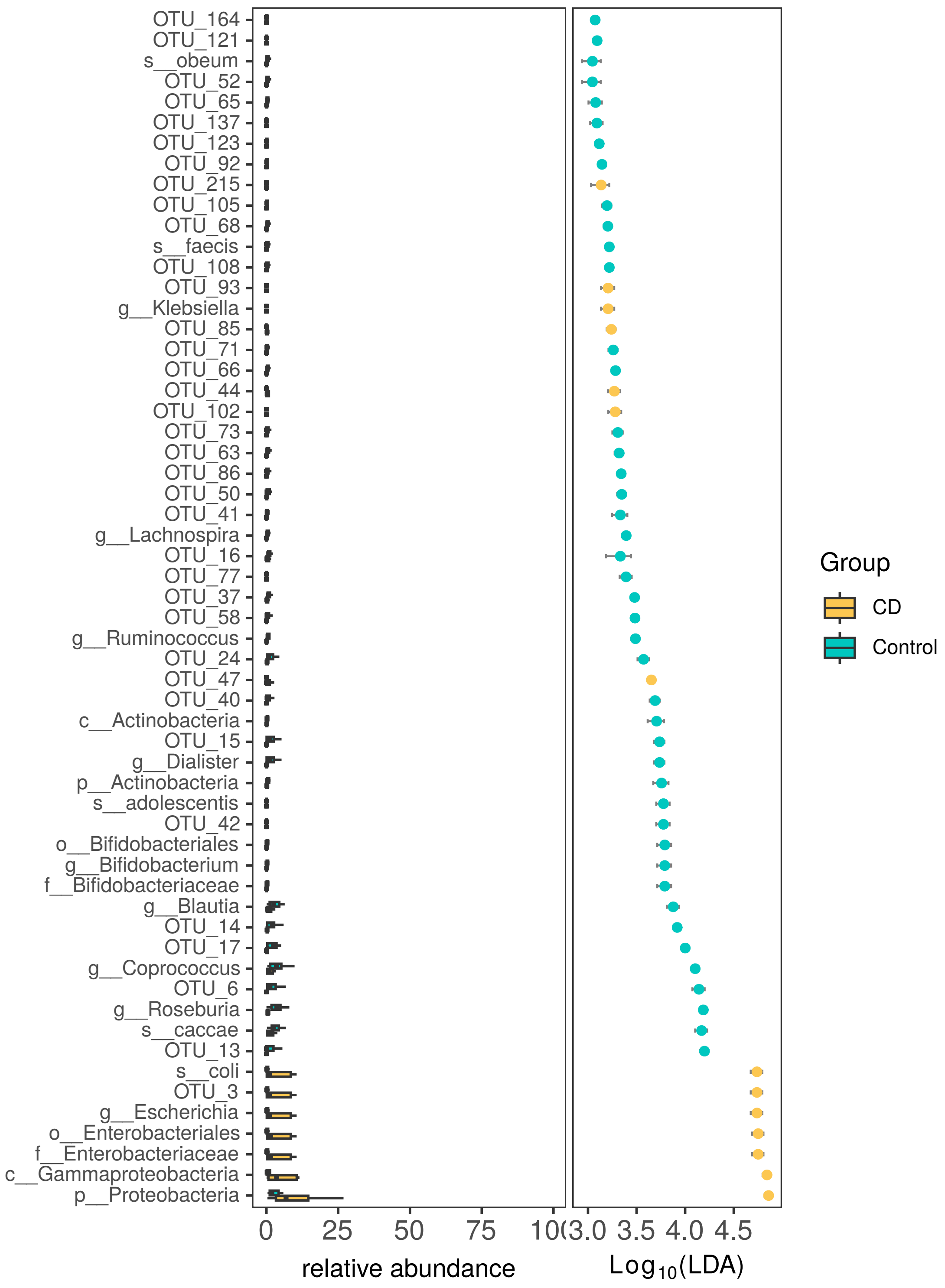


Fig. SA.14: **The boxplot and the LDA score of different taxa.** The left panel represented the relative abundance of the different taxa, the right panel represented the LDA effect size (95% confidence interval) of different taxa.

2.6.3 visualization of differential results (with action = "get") by `ggdifftaxbar`

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

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

2.7 Significant differential clades for the diagnosis of some related diseases

MicrobiotaProcess provided *mp_balance_clade* to calculate the balance of clades of phylogenetic tree with the abundance (geometric mean, mean or median) of tips. Then we can use *mp_diff_analysis* to identify the significantly differential clades.

```r
library(ggplot2)
library(ggsci)
library(ggtree)
library(forcats)

mpse3 <- mpse2 %>% dplyr::filter(Class != 'c__un_p__Proteobacteria')
mpse3 %>%
    mp_balance_clade(
      .abundance = Abundance,
      force = TRUE,
      relative = FALSE,
      pseudonum = 1,
      balance_fun='geometric.mean'
    ) -> mpse.balance.node

mpse.balance.node %<>%
    mp_diff_analysis(
      .abundance = Abundance,
      force = TRUE,
      relative = FALSE,
      .group = Group,
      fc.method = 'compare_mean'
    )

mpse.balance.node %>%
    mp_extract_feature() %>%
    dplyr::filter(!is.na(Sign_Group)) -> ba.node.sign

ba.node.sign %>%
    dplyr::filter(node %in% c(434, 426, 343, 388)) %>%
    tidyr::unnest(Balance_offspring) %>%
    tidyr::unnest(offspringTiplabel) %>%
    select(offspringTiplabel, node) %>%
    dplyr::mutate_at('node', as.character) %>%
    dplyr::rename(BalanceNode = 'node') -> Hight.BalanceNode

p1 <- mpse3 %>% mp_extract_otutree() %>%
      ggtree(
        layout = 'circular',
        size = .25,
        color = '#bed0d1'
      ) %<+% Hight.BalanceNode +
      geom_tiplab(
        data = td_filter(!is.na(BalanceNode)),
        size = 1.2,
        mapping = aes(color=BalanceNode),
        align = TRUE,
        linesize = .5,
        linetype = 3,
        offset = 1.45
      ) +
      scale_color_npg(guide=guide_legend(overide.aes=list(size = 2.6))) +
      geom_tiplab(
        data = td_filter(is.na(BalanceNode)),
        size = 1.2,
        align = TRUE,
        linesize = .05,
        linetype = 3,
        offset = .9
      ) +
      geom_point(
        data = td_filter(node %in% ba.node.sign$node),
        size = .3,
        color = 'red'
      ) +
      ggrepel::geom_text_repel(
        data = td_filter(node %in% ba.node.sign$node),
        mapping = aes(label = node),
        bg.color = 'white',
        size = 2,
        segment.size = .1,
        min.segment.length = 0,
        max.overlaps = 24,
      )

ba.node.sign2 <- ba.node.sign %>%
                 tidyr::unnest(Balance_offspring) %>%
                 tidyr::unnest(offspringTiplabel)

bla.sign.da <- ba.node.sign %>%
    select(OTU, AbundanceBySample) %>%
    tidyr::unnest(AbundanceBySample) %>%
    select(OTU, Sample, Abundance, Group) %>%
    tidyr::pivot_wider(id_cols=c('Sample', 'Group'), values_from=Abundance, names_from=OTU) %>%
    dplyr::mutate_at('Group', as.factor)

otu.sign.da <- mpse3 %>% mp_extract_feature() %>%
    filter(!is.na(Sign_Group)) %>%
    tidyr::unnest(RareAbundanceBySample) %>%
    select(OTU, RelRareAbundanceBySample, Sample, Group) %>%
    tidyr::pivot_wider(id_cols=c('Sample', 'Group'), names_from='OTU', values_from=RelRareAbundanceBySample) %>
    dplyr::mutate_at('Group', as.factor)

p2 <- p1 +
    geom_fruit(
      data = ba.node.sign2,
      geom = geom_tile,
      mapping = aes(
        x = OTU,
        y = offspringTiplabel,
        fill = Clade
      ),
      axis.params = list(axis='none', text.angle=-45, vjust=1, hjust=0, text.size=2),
      grid.params = list(),
      pwidth = .5,
      offset = .01
    ) +
    scale_fill_manual(values = c('#00D617', '#E6A519')) +
    scale_y_continuous(limits=c(-1, NA))

p3 <- p2 +
   ggnewscale::new_scale_fill() +
   geom_fruit(
     data = td_filter(RelRareAbundanceBySample > 0, .f=td_unnest(RareAbundanceBySample)),
     geom = geom_star,
     mapping = aes(
       x = fct_reorder(Sample, Group, .fun=min),
       fill = Group,
       size = RelRareAbundanceBySample
     ),
     offset = .15,
     pwidth = 1.5,
     starshape = 13,
     starstroke = .05,
     grid.params = list(vline=TRUE, size = 0.1, color="snow2", linetype = 1)
   ) +
   scale_fill_manual(values = cols) +
   scale_size_continuous(
     name = 'Relative Abundance(%)',
     range = c(.5, 4),
     guide = guide_legend(overide.aes = list(starstroke = .5))
   )

sign.otu <- mpse3 %>%
    mp_extract_feature() %>%
    filter(!is.na(Sign_Group)) %>%
    select(OTU, LDAmean, Sign_Group) %>%
    dplyr::left_join(
      mpse3 %>% mp_extract_taxonomy(),
      by = 'OTU'
    )

p4 <- p3 %<+% sign.otu +
   ggnewscale::new_scale_fill() +
   geom_fruit(
      data = td_filter(!is.na(Sign_Group)),
      geom = geom_tile,
      mapping = aes(fill=Phylum),
      width = .1,
      offset = .1
   ) +
   ggnewscale::new_scale_fill() +
   geom_fruit(
      data = td_filter(!is.na(Sign_Group)),
      geom = geom_col,
      mapping = aes(x = LDAmean, fill = Sign_Group),
      orientation = "y",
      offset = 0.05,
      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) ,
      show.legend = FALSE
   ) +
   scale_fill_manual(values = cols) +
   theme(
     legend.key.width = unit(.3, 'cm'),
     legend.key.height = unit(.3, 'cm'),
     legend.text = element_text(size=6),
     legend.title = element_text(size=8),
     legend.margin = ggplot2::margin(-.25, 0, 0, 0, 'cm')
   )
```

```
p4
```

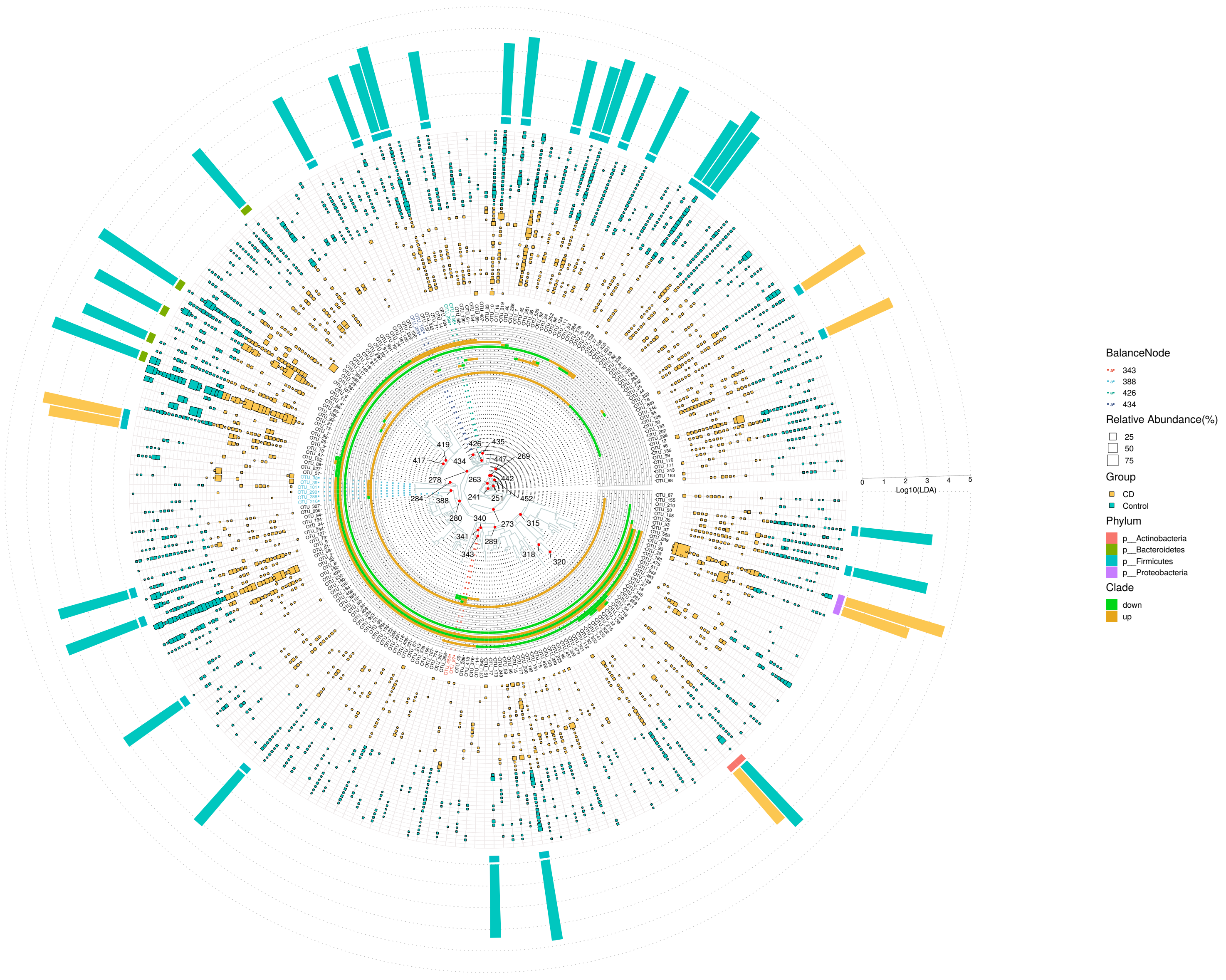


Fig. SA.15: **The cladogram of significant differential clades between the CD and Control group.** The external heatmap represents the differential clades (up and down). The external point layer represents the relative abundance of each OTU on each sample. The external bar plot represents the mean LDA of the differential OTUs.

We found some differential clades contain the closely related species that were not be detected in the previous differential analysis, such as OTU_454/OTU_97 (both belong to Clostridiaceae SMB53), OTU_152/OTU_233 (both belong to Lachnospira), which suggested the phylogenetic transform can improve the detection of differential signals by accumulating the small consistent differences at a broad resolution.

```r
no.sig.OTUs.da <- mpse.balance.node %>% mp_extract_feature() %>%
    dplyr::filter(!is.na(Sign_Group)) %>%
    select(OTU, node, Balance_offspring) %>%
    tidyr::unnest(Balance_offspring) %>%
    dplyr::filter(node %in% c(434, 426, 343, 388)) %>%
    tidyr::unnest(offspringTiplabel) %>%
    dplyr::arrange(node)

no.sig.OTUs <- no.sig.OTUs.da %>% dplyr::pull(offspringTiplabel)

no.sig.otu.genus <- mpse2 %>%
    mp_extract_taxonomy %>%
    dplyr::filter(OTU %in% no.sig.OTUs) %>%
    select(OTU, Genus) %>%
    dplyr::mutate(Genus=gsub("g__Clostridium_f__Clostridiaceae", "g__Clostridium", Genus))

theme_annot <- function(){
    th <- list(
        labs(x=NULL, y=NULL),
        theme_bw(),
        theme(
          axis.text = element_blank(),
          axis.ticks = element_blank(),
          panel.grid = element_blank(),
          panel.border = element_blank(),
          legend.key.height = unit(.3, 'cm'),
          legend.key.width = unit(.3, "cm"),
          legend.text = element_text(size=7),
          legend.title = element_text(size=9)
        )
    )
    return(th)
}

mpse2 %>%
    filter(OTU %in% no.sig.OTUs) %>%
    as_tibble() %>%
    ggplot(
      aes(
        y = fct_reorder(Sample, Group, .fun = min),
        x = fct_relevel(OTU, no.sig.OTUs),
        fill = RelRareAbundanceBySample,
        size = RelRareAbundanceBySample
      )
    ) +
    geom_tile(color='grey', size=.5, fill=NA) +
    geom_point(
      data = td_filter(RelRareAbundanceBySample!=0),
      shape=21
    ) +
    scale_fill_gradient2() +
    theme_bw() +
    theme(axis.text.x=element_text(angle=45, hjust=1), panel.grid=element_blank()) +
    labs(x=NULL, y=NULL, size="RelAbun", fill='RelAbun') -> f1

mpse2 %>%
    mp_extract_sample() %>%
    ggplot(aes(y=Sample, fill=Group, x="Group")) +
    geom_tile() +
    scale_fill_manual(values = cols) +
    theme_annot() +
    labs(x=NULL, y=NULL) -> f2

no.sig.OTUs.da %>%
    ggplot(aes(x=fct_reorder(offspringTiplabel, node, .fun=min),
               y='BalanceNode',
               fill=as.character(node))) +
    geom_tile() +
    scale_fill_uchicago() +
    theme_annot() +
    labs(x=NULL, y=NULL, fill='BalanceNode') -> f3

no.sig.otu.genus %>% ggplot(aes(x=OTU,y="Genus",fill=Genus)) +
    geom_tile() +
    labs(fill = 'Genus') +
    coord_cartesian(expand=F) +
    theme_annot() +
    scale_fill_npg() -> f4

ff <- f1 %>%
      aplot::insert_right(f2, width = .1) %>%
      aplot::insert_top(f3, height = .03) %>%
      aplot::insert_top(f4, height = .028)

f.box <- mpse.balance.node %>%
    dplyr::filter(node %in% c(434, 426, 343, 388)) %>%
    as_tibble() %>%
    tidyr::unnest(Balance_offspring) %>%
    dplyr::filter(Clade == 'up') %>%
    ggplot(aes(y = Group, x = Abundance, fill = Group)) +
    geom_boxplot(orientation = 'y') +
    geom_jitter(color = 'grey', height = .2) +
    facet_wrap(pseudolabel~., ncol = 1, strip.position = 'top', scales = 'free') +
    scale_fill_manual(values = cols) +
    ggsignif::geom_signif(comparisons = list(c('CD', 'Control')), orientation = 'y') +
    scale_y_discrete(position = 'right') +
    ylab(NULL) +
    xlab('Balance Score') +
    theme_bw() +
    theme(
      legend.position = 'none',
      panel.grid = element_blank(),
      strip.background = element_rect(fill='grey', color=NA),
      strip.text = element_text(face='bold')
    )

aplot::plot_list(ff, f.box, tag_levels = "A", widths=c(4.5, 5))
```

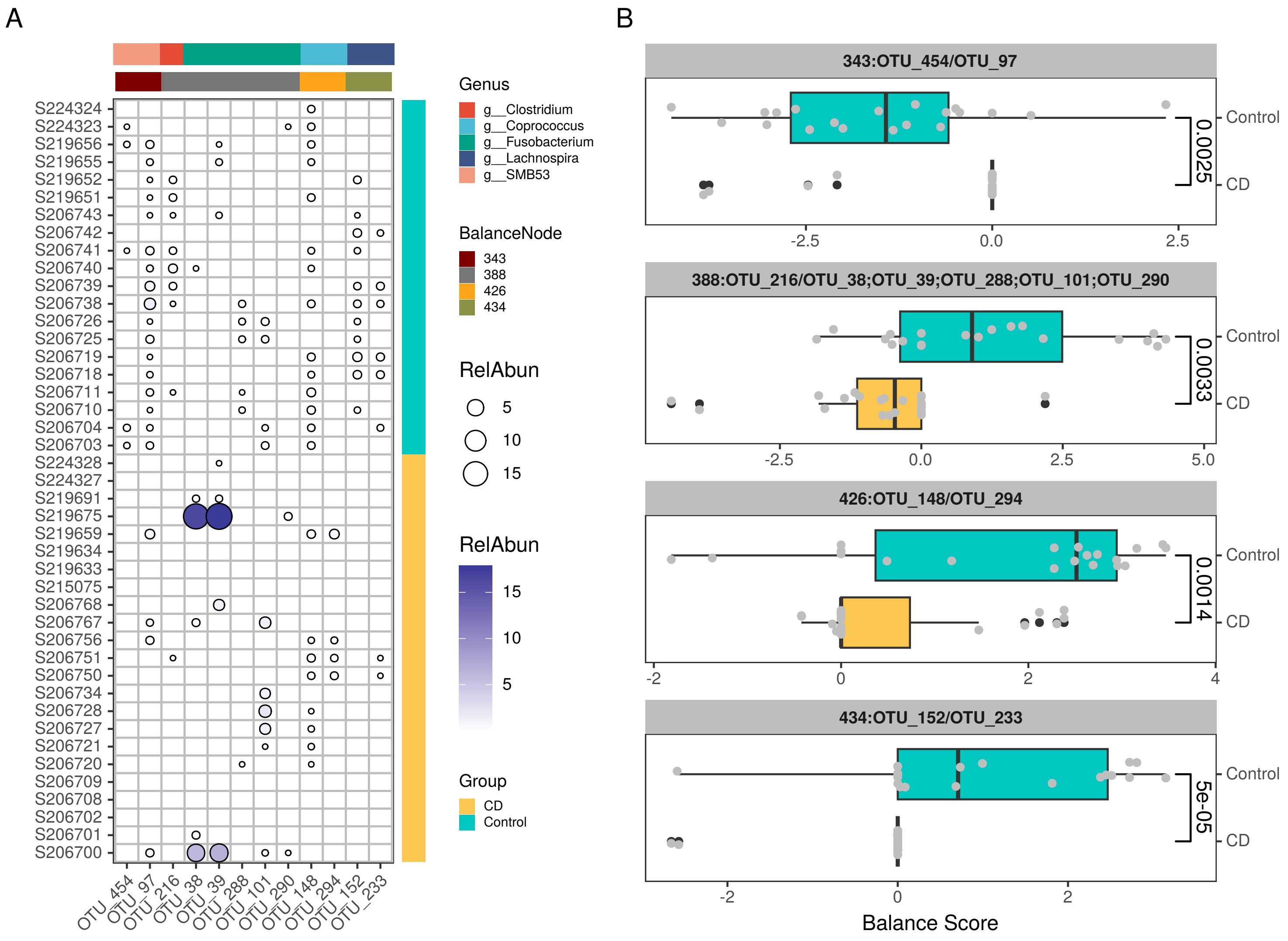


Fig. SA.16: **The balance scores of significantly differential clades and the relative abundance of their original OTUs** (A) the relative abundance, the taxonomy information and the compositional clades annotation of the original OTUs. (B) The balance scores of significantly differential clades.

2.8 Performing differential analysis among multiple groups

It is the same to perform the differential analysis between two groups using mp_diff_analysis. For example, we perform the following example to show this. The dataset was from a colorectal cancer study (Zeller et al. 2014), which was obtained with curatedMetagenomicData. The samples were from stools of the CRC, the Adenoma and the Control individuals. Through the analysis of *mp_diff_analysis*, we found *Fusobacterium gonidiaformans*, *Porphyromonas asaccharolytica*, *Parvimonas micra*, *Peptostreptococcus stomatis* and *Escherichia coli* were significantly enriched in CRC (colorectal cancer), *Ruminococcus lactaris* was significantly enriched in Adenoma (colorectal adenoma), but *Bifidobacterium longum*, *Bifidobacterium catenulatum*, *Blautia wexlerae* and *Anaerostipes hadrus* were significantly decreased in CRC and Adenoma.

```r
ExperimentHub::setExperimentHubOption('LOCAL', TRUE)
xx <- curatedMetagenomicData('ZellerG_2014.relative_abundance', dryrun=F)
xx[[1]] %>% as.mpse -> mpse.crc.ZellerG_2014

mpse.crc.ZellerG_2014 %<>% mp_diff_analysis(
    .abundance = Abundance,
    .group = disease,
    force = TRUE,
    relative = FALSE,
    first.test.alpha = 0.05,
    filter.p = "pvalue"
)
```

```
p.cladogram <- mpse.crc.ZellerG_2014 %>%
     mp_plot_diff_cladogram(
       .group = disease,
       .size = pvalue,
       taxa.class = Genus,
       hilight.alpha = .3,
       bg.tree.size = .15,
       bg.point.stroke = .1,
       bg.point.size = 1.5,
       label.size = 2.6,
       tip.annot = FALSE,
       as.tiplab = FALSE
     ) +
     scale_fill_diff_cladogram(
       values = c('red', 'orange', 'deepskyblue'),
     ) +
     scale_size_continuous(
       range = c(1, 4)
     )

p.cladogram
```

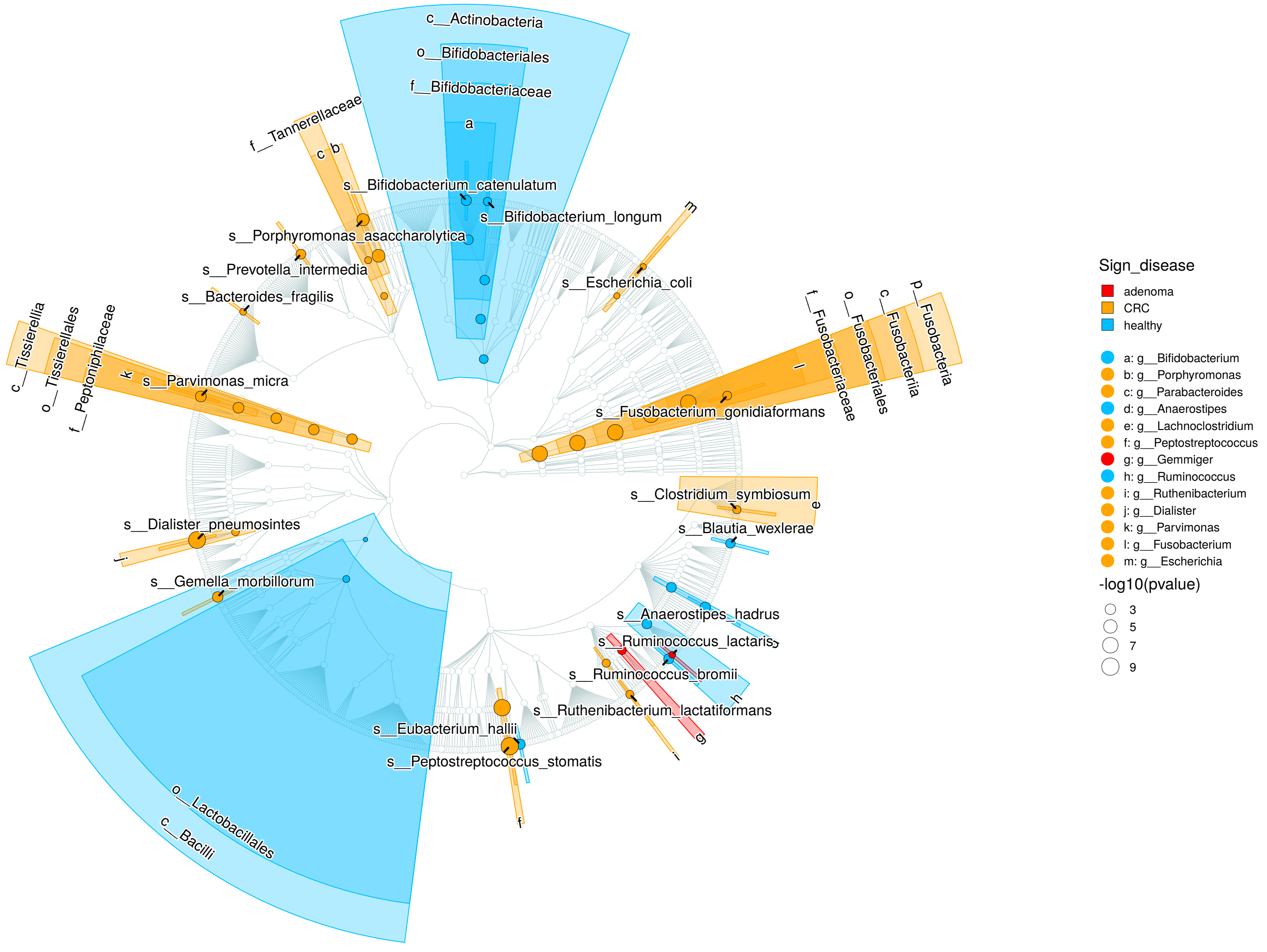


Fig. SA.17: **The cladogram of significant differential taxa** The hight light represented the differential taxa enriched in the corresponding group.

2.9 Interoperable with the existing computing ecosystem

Because the *MPSE* object of *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 is also 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
      )
pp[[1]] <- pp[[1]] +
           scale_fill_viridis_c(
             option='A',
             na.value = 0,
             trans = 'log10'
           ) +
           guides(
             fill = guide_colorbar(
               title = expression(log[10]("relative abundance")),
               title.position = "right",
               title.theme = element_text(angle=-90, size=9, vjust=.5, hjust=.5),
               label.theme = element_text(angle=-90, size=7, vjust=.5, hjust=.5),
               barwidth = unit(.3, 'cm'),
               barheight = unit(5, 'cm')
             )
           ) +
           theme(
             axis.text.x = element_blank(),
             axis.text.y = element_text(size = 6),
           )

pp[[2]] <- pp[[2]] +
           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)
           )

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 = 0.4)
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),
        )
      )
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
"
px <- plot_list(pp, sample.tree, p, design = design, tag_levels = "A")
px
```

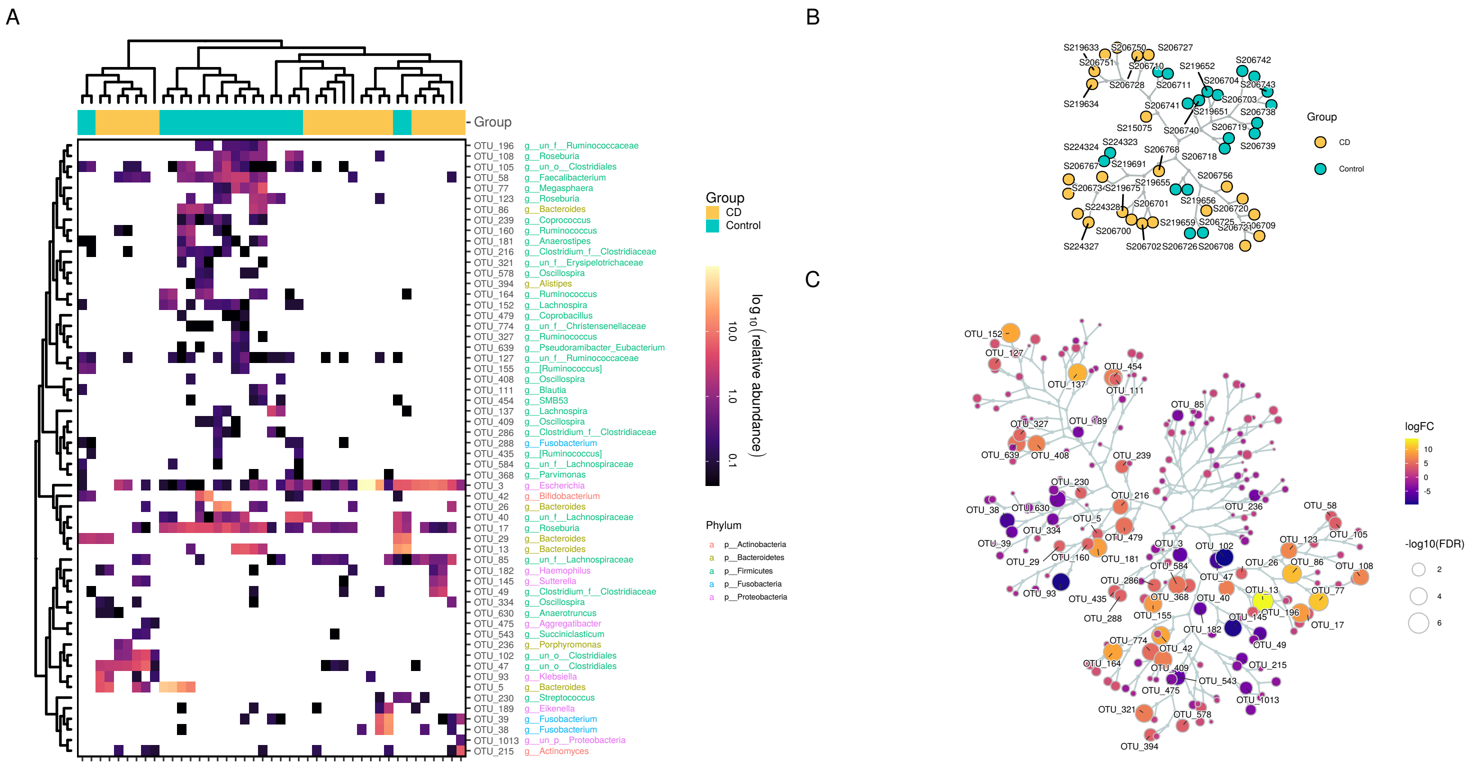


Fig. SA.18: **The results of different OTUs based on the edgeR__quasi__likelihood with tidybulk** (A). The relative abundance heatmap of the different OTUs. (B). The hierarchical cluster of samples based on the relative abundance of the different OTUs. (C). The hierarchical cluster of OTUs based on the 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).

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 in the analysis using *tidybulk*. This operation is generally needed in standard whole-transcriptome workflows. However, if it is performed 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 extracted the same different OTUs, we found the abundance of the same OTUs belonging 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.

```
mpse2 %>%
    mp_extract_feature(addtaxa=T) %>%
    dplyr::filter(OTU %in% do.call(intersect, base::unname(DE.method)))
```

```
## # A tibble: 36 x 22
```

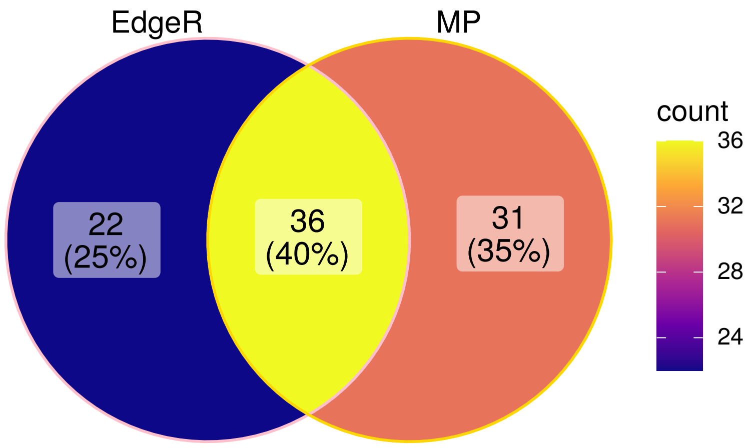


Fig. SA.19: **The comparison of the different analysis result between the edgeR and MicrobiotaProcess**

```
##    OTU     ggups~1 logFC logCPM     F  PValue     FDR Kingdom Phylum Class Order
##    <chr>   <list>  <dbl>  <dbl> <dbl>   <dbl>   <dbl> <chr>   <chr>  <chr> <chr>
##  1 OTU_215 <chr>   -4.44   9.41  9.00 3.50e-3 1.79e-2 k__Bac~ p__Ac~ c__A~ o__A~
##  2 OTU_42  <chr>    8.97  12.1  29.0  5.79e-7 1.48e-5 k__Bac~ p__Ac~ c__A~ o__B~
##  3 OTU_86  <chr>   10.4   11.1  36.2  8.10e-8 4.66e-6 k__Bac~ p__Ba~ c__B~ o__B~
##  4 OTU_13  <chr>   13.9   14.5  48.9  1.48e-9 3.41e-7 k__Bac~ p__Ba~ c__B~ o__B~
##  5 OTU_774 <chr>    4.93   6.65 24.5  5.30e-6 5.80e-5 k__Bac~ p__Fi~ c__C~ o__C~
##  6 OTU_216 <chr>    5.73   8.77 17.3  7.31e-5 6.00e-4 k__Bac~ p__Fi~ c__C~ o__C~
##  7 OTU_286 <chr>    4.55   8.20 11.3  1.15e-3 7.58e-3 k__Bac~ p__Fi~ c__C~ o__C~
##  8 OTU_454 <chr>    6.39   7.53 29.5  8.20e-7 1.89e-5 k__Bac~ p__Fi~ c__C~ o__C~
##  9 OTU_639 <chr>    5.54   6.97 25.2  3.95e-6 4.59e-5 k__Bac~ p__Fi~ c__C~ o__C~
## 10 OTU_155 <chr>    8.10   8.92 28.7  1.09e-6 2.13e-5 k__Bac~ p__Fi~ c__C~ o__C~
## # ... with 26 more rows, 11 more variables: Family <chr>, Genus <chr>,
## #   Species <chr>, RareAbundanceBySample <list>, RareAbundanceByGroup <list>,
## #   LDAupper <dbl>, LDAmean <dbl>, LDAlower <dbl>, Sign_Group <chr>,
## #   pvalue <dbl>, fdr <dbl>, and abbreviated variable name 1: ggupsetOfGroup
```

2.10 Interface to integrate external data

In addtion, because the *MPSE* used `treedata` class to store the taxonomy, phylogenetic and related information, the related results of other tools also can be integrated to it easily, we also developed *left_join* to cooperate. Then the new *MPSE* class can be further analyzed and visualized.

2.10.1 Integrating the results of other distance methods

The *mp_cal_dist* of *MicrobiotaProcess* had provided many distance methods, such as "bray", "aitchison", "jaccard", "gower", "altGower" etc. But if users want to use other methods that are not provided in *MicrobiotaProcess*. They can use *left_join* to add the result to *MPSE* class.

```
otu.da <- mpse2 %>% mp_extract_assays(.abundance=Abundance)
Aitchison.dist <- robCompositions::aDist(t(otu.da+1))
p1 <- mpse2 %>%
       left_join(y=list(Aitchison=Aitchison.dist)) %>%
       mp_plot_dist(.distmethod=Aitchison, .group=Group, group.test=T) +
       scale_fill_manual(values=c("orange", "#00A08A", "deepskyblue")) +
       scale_color_manual(values=c("orange", "#00A08A", "deepskyblue"))
p2 <- mpse2 %>%
      left_join(y=list(Aitchison=Aitchison.dist)) %>%
      mp_cal_pcoa(distmethod="Aitchison") %>%
      mp_plot_ord(.group=Group) +
      scale_fill_manual(values=cols)
aplot::plot_list(p1, p2, widths = c(0.5, 2))
```

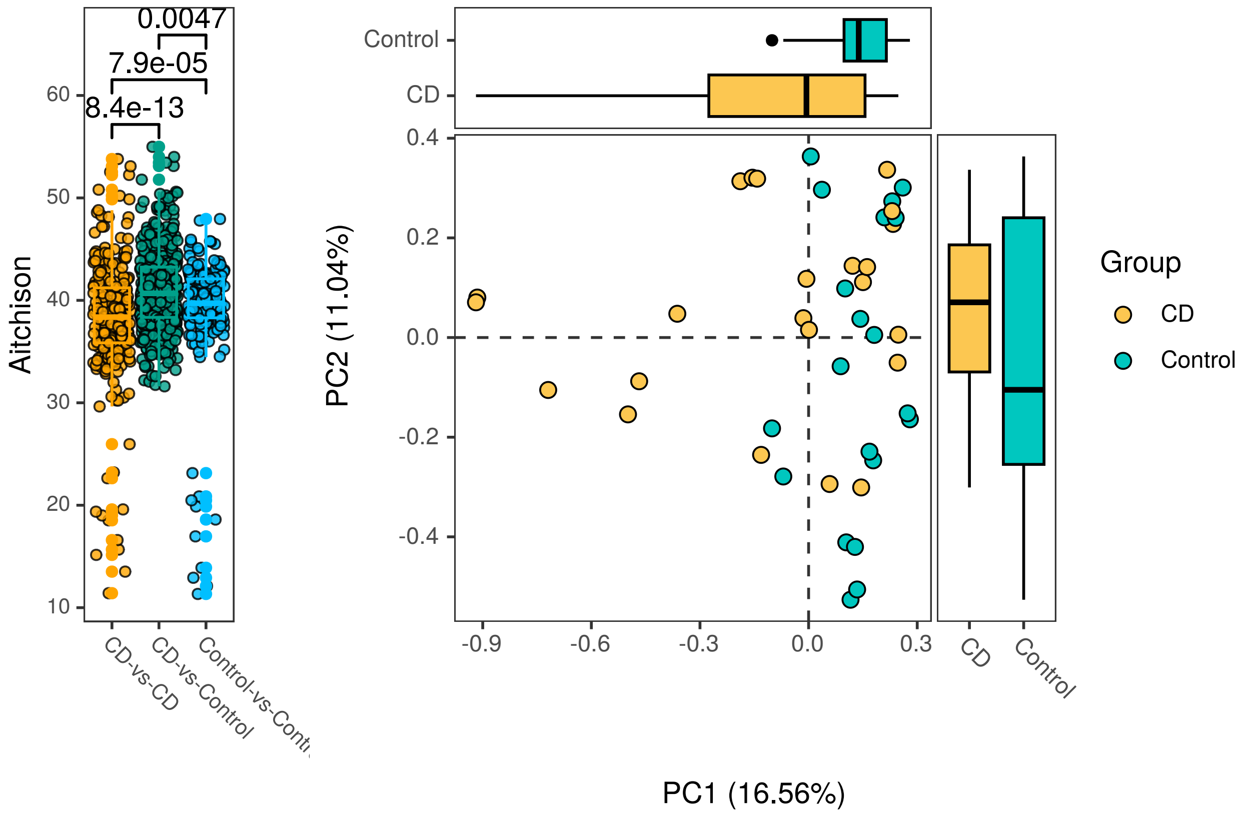


Fig. SA.20: **Integrating the other distance with left_join**

2.10.2 Integrating the results of other DAA tools

If your want to integrate the results of differential OTU and taxa. You can provided a data frame that contains a column of OTU and taxa names, a column of enriched group and other statistical results such like pvalue or FDR value, then using *left_join* to integrate them to *taxatree* slot in *MPSE* class.

```r
library(GUniFrac)
library(matrixStats)
# obtain the abundance on all taxonomy levels
all.abun <- mpse2 %>% mp_extract_abundance() %>%
  dplyr::select(label, RareAbundanceBySample)

# build the matrix input (longer format to wider format) of ZicoSeq
All.features <- all.abun %>%
  tidyr::unnest(RareAbundanceBySample) %>%
  dplyr::select(label, Sample, RelRareAbundanceBySample) %>%
  tidyr::pivot_wider(
    id_cols = label,
    names_from = 'Sample',
    values_from = RelRareAbundanceBySample
  ) %>%
  tibble::column_to_rownames(var='label') %>% as.matrix()

All.features <- All.features[!rowSds(All.features)==0, ]

sample.da <- mpse2 %>%
             mp_extract_sample() %>%
             dplyr::select(Sample, Group) %>%
             tibble::column_to_rownames(var='Sample')
set.seed(123)
zicoseq.res <- ZicoSeq(meta.dat=sample.da, feature.dat=All.features/100, grp.name='Group',
                       prev.filter=.1, perm.no=999, feature.dat.type='proportion', verbose=F)
```

```
## For proportion and other data types,  posterior sampling will not be performed!
## On average,  1  outlier counts will be replaced for each feature!
```

```r
res.df <- data.frame(zicoseq.res$p.adj.fdr)
colnames(res.df) <- 'FDR.zicoseq'

# build the enrich group information of the significant features.
res.sign <- all.abun %>% dplyr::filter(label %in% rownames(res.df[res.df$FDR.zicoseq <=0.05,,drop=FALSE])) %>%
    tidyr::unnest(RareAbundanceBySample) %>%
    dplyr::group_by(label, Group) %>%
    dplyr::summarize(MeanAbu=mean(RareAbundance)) %>%
    dplyr::slice_max(MeanAbu) %>%
    dplyr::ungroup() %>%
    dplyr::rename(Sign_Group=Group) %>%
    dplyr::select(label, Sign_Group)

res.df %<>% as_tibble(rownames='label') %>% dplyr::left_join(res.sign)

# remove the results of other DAA methods.
taxa.tree <- mpse2 %>% mp_extract_taxatree() %>%
  dplyr::select(-c('LDAupper', 'LDAmean', 'LDAlower', 'pvalue', 'fdr', 'Sign_Group'), keep.td=T)

# add the results of ZicoSeq to taxatree slot in MPSE
taxa.tree %<>% dplyr::left_join(res.df, by='label')
mpse4 <- mpse2
taxatree(mpse4) <- taxa.tree
zicoseq.p1 <- mpse4 %>% mp_plot_diff_cladogram(
                .group = Sign_Group,
                .size = FDR.zicoseq,
                removeUnknown = T,
                as.tiplab = F
             ) +
             scale_fill_diff_cladogram(values=cols)
zicoseq.p1
```

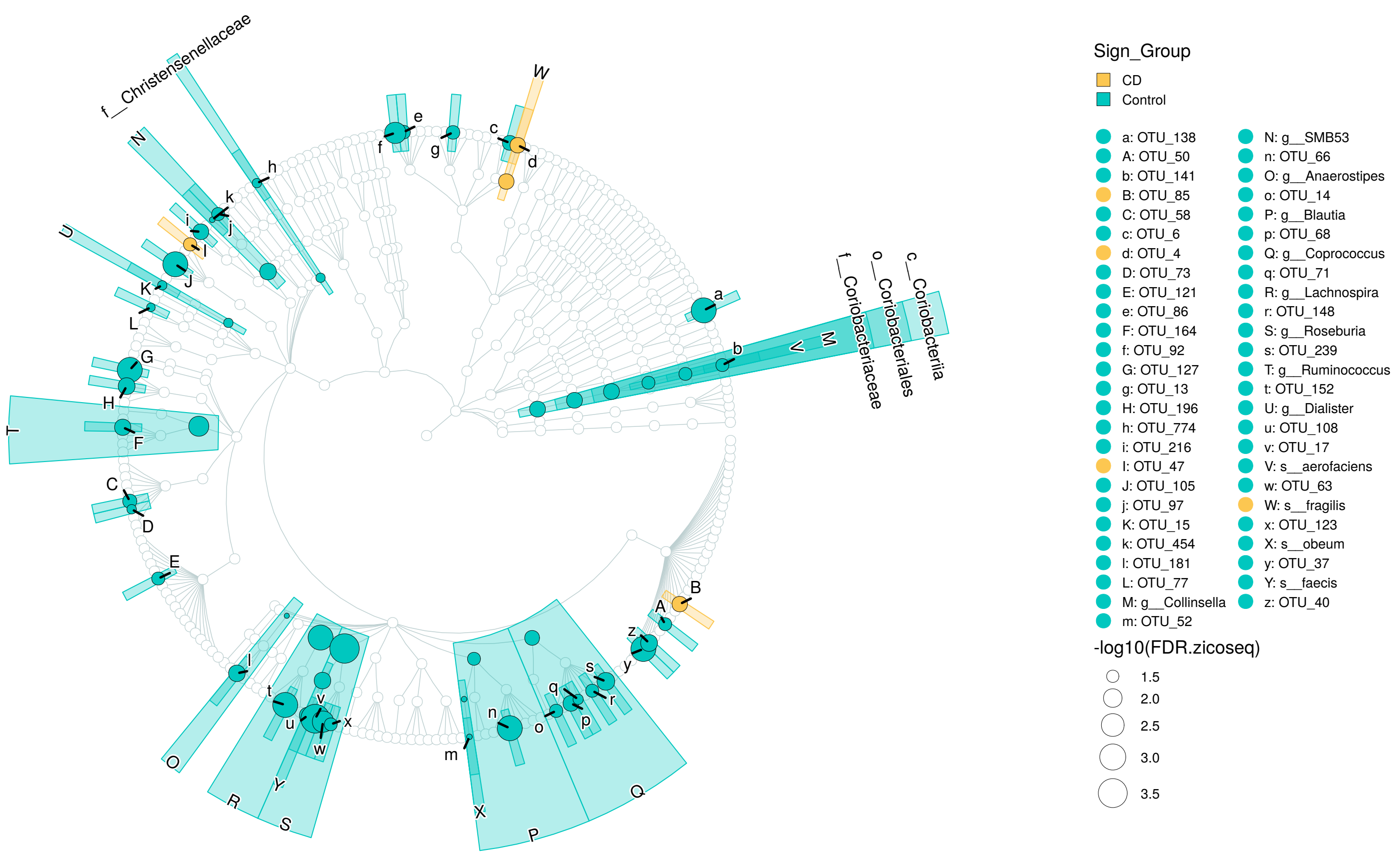


Fig. SA.21: **Integrating the different analysis results (contains differential OTU and taxa) of ZiCoSeq and visualizing using mp_plot_diff_cladogram**

```
zicoseq.p2 <- mpse4 %>% mp_plot_diff_manhattan(
                 .group = Sign_Group,
                 .y=-log10(FDR.zicoseq),
                 taxa.class = OTU,
                 anno.taxa.class=Phylum) +
              scale_shape_manual(
                 values = c(17, 25, 19)
              )
zicoseq.p2
```

But if you want integrate the results of differential OTU only, you can use *left_join* to integrate the result to *MPSE* class directly.

```
ps2 <- mpse2 %>% as.phyloseq(.abundance=RareAbundance)
library(MicrobiomeStat)
library(dplyr)
res.linda <- linda(phyloseq.obj=ps2, formula='~Group', p.adj.method='fdr', prev.filter=.1)
```

```
## 37  features are filtered!
## The filtered data has  43  samples and  193  features will be tested!
## Pseudo-count approach is used.
## Fit linear models ...
## Completed.
```

```
tbl.res <- res.linda$output$GroupControl

tbl.res %<>% tibble::as_tibble(rownames='OTU') %>%
            dplyr::mutate(Sign_Group = case_when(
```

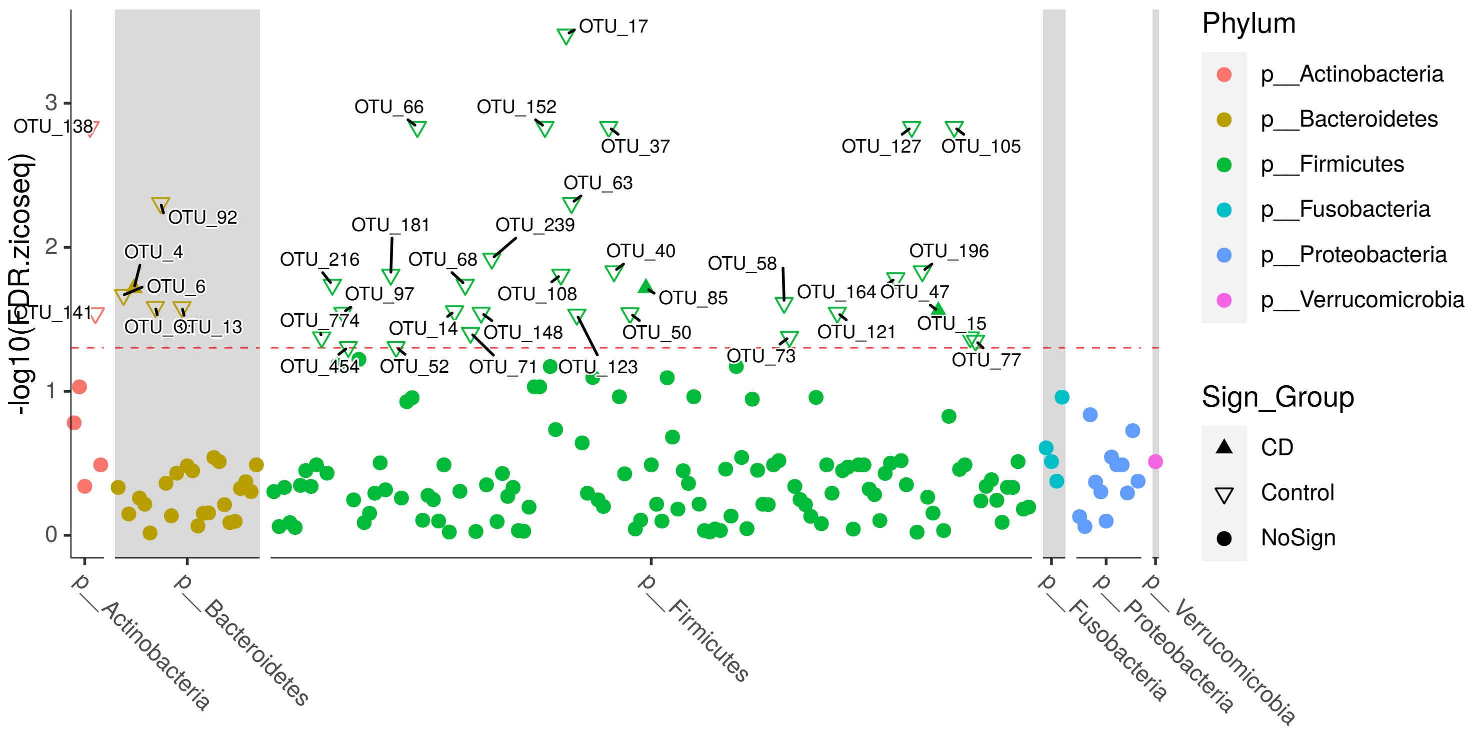


Fig. SA.22: **Integrating the different analysis results (contains differential OTU and taxa) of ZiCoSeq and visualizing using mp__plot__diff__manhattan**

```
                                 log2FoldChange < 0 & reject ~ "CD",
                                 log2FoldChange > 0 & reject ~ 'Control',
                                 TRUE ~ as.character(NA))
            )


mpse5 <- ps2 %>%
         as.mpse() %>%
         mp_rrarefy() %>%
         mp_cal_abundance(.abundance=RareAbundance)
mpse5 %<>% left_join(tbl.res, by='OTU')

# visualizing the results with mp_plot_diff_boxplot
# and mp_plot_diff_res

linda.p1 <- mpse5 %>%
      mp_plot_diff_res(
        .group = Sign_Group,
        point.size = padj,
        barplot.x = lfcSE
      ) +
      scale_fill_manual(
        aesthetics = "fill_new",
        values = cols
      ) +
      scale_fill_manual(
        values = cols
      )
linda.p1
```

```
linda.p2 <- mpse5 %>%
      mp_plot_diff_boxplot(
        .group = Sign_Group,
```

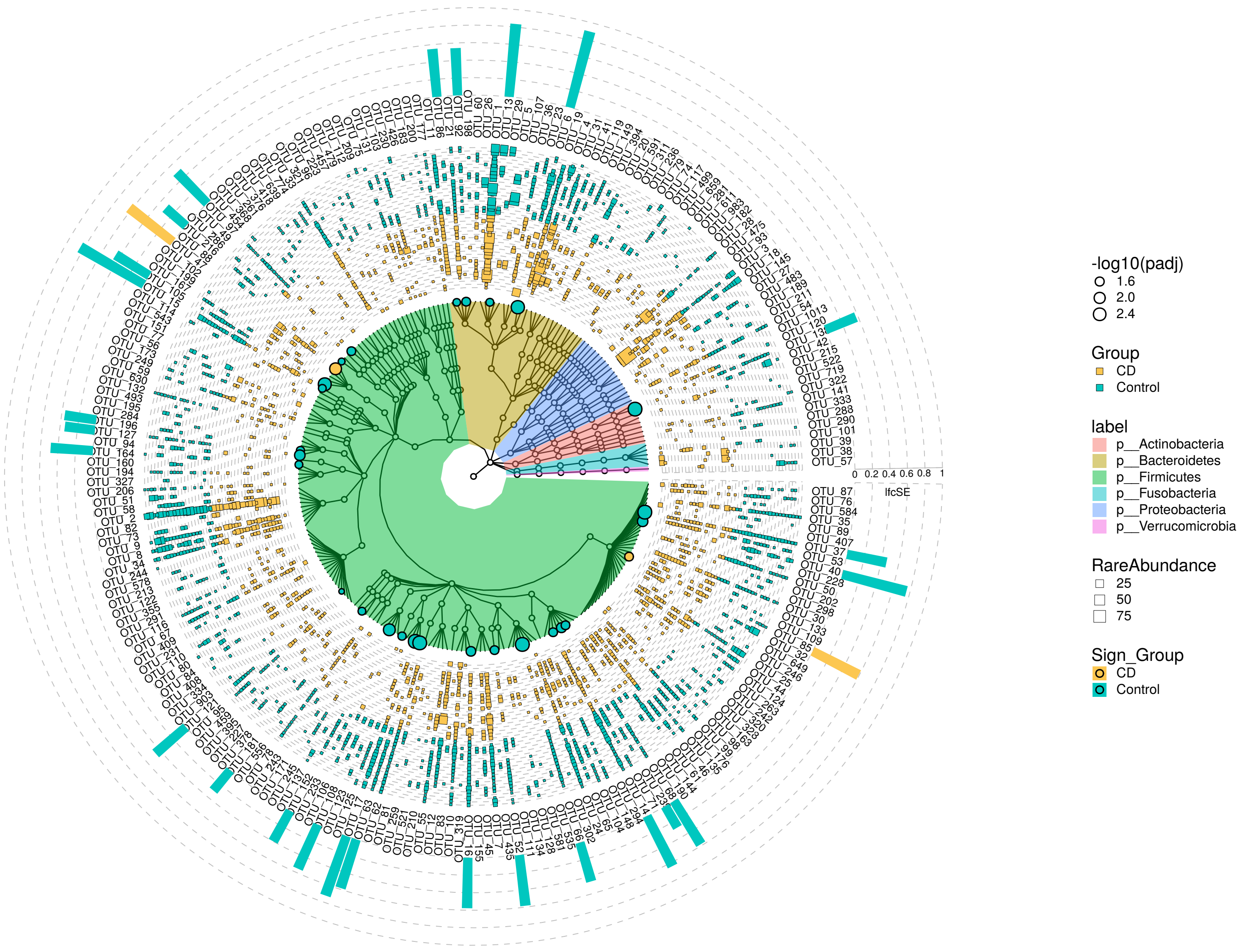


Fig. SA.23: **Integrating the different analysis results (only differential OTU) of LinDA and visualizing using mp__plot__diff__res**

```
            .size=-log10(padj),
            point.x = lfcSE
        ) %>%
        set_diff_boxplot_color(
            values = cols,
            guide = guide_legend(title=NULL)
        )
linda.p2
```

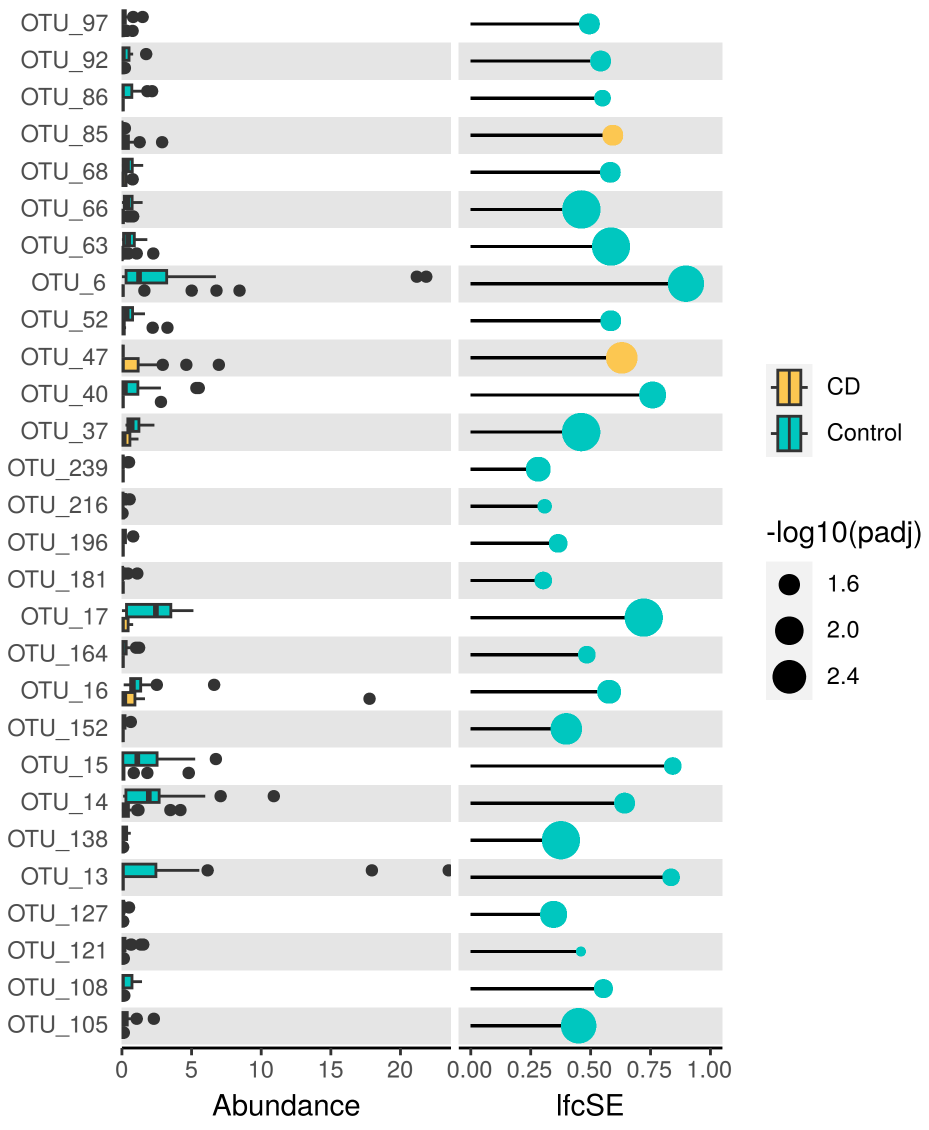


Fig. SA.24: **Integrating the different analysis results (only differential OTU) of LinDA and visualizing using mp_plot_diff_boxplot**

3 the analysis of the other published pediatric CD stool samples

In the previous session, we described how to use *MicrobiotaProcess* to analyze the 16s rDNA data. However, it also can be applied to metagenome or metatranscriptome species community data and functional data analysis. In this session, we used the example datasets about the other published pediatric CD stool microbial study (Douglas et al. 2018) to show how to use *MicrobiotaProcess* to do the related analysis. The datasets were obtained from the github[2]. 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 The parsing of the 16s data and construction of MPSE class

The session is similar with the session 2, some operations can refer to the previous session 2.

```
cols <- c('#fcc751ff', '#00c7bfff')
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
```

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

```
## # A MPSE-tibble (MPSE object) abstraction: 37,392 x 10
## # OTU=984 | Samples=38 | Assays=Abundance | Taxonomy=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__Firm~ c__C~ o__C~ f__Ru~ g__u~ s__un~
##  2 196271  S15            0 k__Bacteria p__Firm~ c__C~ o__C~ f__La~ g__u~ s__un~
##  3 196270  S15            2 k__Bacteria p__Firm~ c__C~ o__C~ f__un~ g__u~ s__un~
##  4 297149  S15            0 k__Bacteria p__Firm~ c__C~ o__C~ f__La~ g__u~ s__un~
##  5 3604981 S15            0 k__Bacteria p__Firm~ c__C~ o__C~ f__La~ g__B~ s__un~
##  6 240755  S15            0 k__Bacteria p__Prot~ c__G~ o__P~ f__Pa~ g__H~ s__in~
##  7 326482  S15            0 k__Bacteria p__Bact~ c__B~ o__B~ f__Pr~ g__P~ s__co~
##  8 4393540 S15            0 k__Bacteria p__Bact~ c__B~ o__B~ f__[B~ g__u~ s__un~
##  9 4339144 S15            0 k__Bacteria p__Bact~ c__B~ o__B~ f__[O~ g__B~ s__un~
## 10 4369050 S15            0 k__Bacteria p__Fuso~ c__F~ 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 | Taxonomy=Kingdom, Phylum, Class, Order, Family, Genus, Speies
##    OTU     Sample Abund~1 disease respo~2 sex     age Kingdom Phylum Class Order
##    <chr>   <chr>    <dbl> <chr>   <chr>   <chr> <dbl> <chr>   <chr>  <chr> <chr>
##  1 358030  S15          5 CN      CN      Male   15.4 k__Bac~ p__Fi~ c__C~ o__C~
##  2 196271  S15          0 CN      CN      Male   15.4 k__Bac~ p__Fi~ c__C~ o__C~
##  3 196270  S15          2 CN      CN      Male   15.4 k__Bac~ p__Fi~ c__C~ o__C~
##  4 297149  S15          0 CN      CN      Male   15.4 k__Bac~ p__Fi~ c__C~ o__C~
##  5 3604981 S15          0 CN      CN      Male   15.4 k__Bac~ p__Fi~ c__C~ o__C~
##  6 240755  S15          0 CN      CN      Male   15.4 k__Bac~ p__Pr~ c__G~ o__P~
##  7 326482  S15          0 CN      CN      Male   15.4 k__Bac~ p__Ba~ c__B~ o__B~
##  8 4393540 S15          0 CN      CN      Male   15.4 k__Bac~ p__Ba~ c__B~ o__B~
##  9 4339144 S15          0 CN      CN      Male   15.4 k__Bac~ p__Ba~ c__B~ o__B~
## 10 4369050 S15          0 CN      CN      Male   15.4 k__Bac~ p__Fu~ c__F~ o__F~
## # ... with 37,382 more rows, 3 more variables: Family <chr>, Genus <chr>,
## #   Speies <chr>, and abbreviated variable names 1: Abundance, 2: response
```

3.2 Functional characterization using the KEGG dataset

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 genes using the *mp_diff_analysis* of *MicrobiotaProcess* (refer to session 2.6). Other operations are similar with the analysis of 16s rDNA data (refer to session 2).

```
KO.da <- read.table("./data/CD_RF_microbiome/biscuit_mgs_KOs.tsv",
           header=TRUE, sep = "\t", row.names=1, check.names=F)
# 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 Differential analysis of KEGG genes abundance

The metric of the KEGG genes was 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(!is.na(Sign_disease)) %>% # Extracting the differential features
    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")
p
```

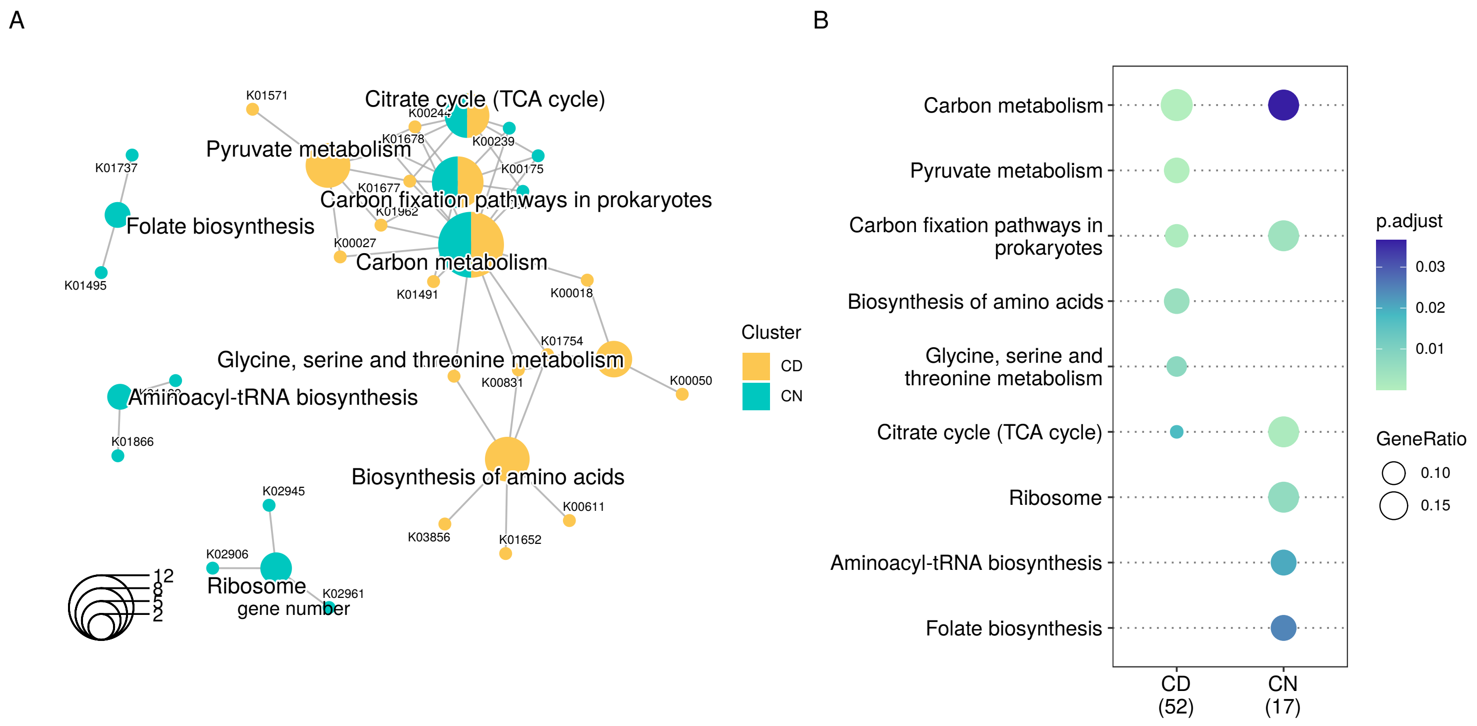


Fig. SA.25: **The result of KEGG pathway enrichment analysis**

The KEGG enrichment results showed that the KEGG pathways of the CD stool group were significantly enriched in the Biosynthesis of amino acids and Glycine, serine, and threonine metabolism, and Pyruvate metabolism (Fig. SA.25). This result was not revealed in the original paper (Douglas et al. 2018), but it was consistent with recent some other related studies, which found that Crohn's Disease microbiomes had an increased potential to synthesize amino acids and Pyruvate metabolism (Heinken, Hertel, and Thiele 2021; Bjerrum et al. 2017; Polunin et al. 2013). In addition, we used some other differential abundance methods to identify the differential KEGG genes, but the two pathways were not found simultaneously in the enrichment results of CD based on the differential genes identified by other methods (refer to the second session of supplemental file B). We think this is because the *mp_diff_analysis* of *MicrobiotaProcess* achieves a better false positive rate (refer to the third session of supplemental file B)

3.3 The species characterization of the metagenomics data

The taxa abundance data from the metagenomics study also can be analyzed by *MicrobiotaProcess*, Here we used the example data from the 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 further analyzed by *MicrobiotaProcess*, which can refer to session3.2 and session4.

```
# 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 | Taxonomy=Kingdom, Phylum, Class, Order, Family, Genus
##    OTU   Sample Abund~1 unkno~2 disease respo~3 sex     age Kingdom Phylum Class
##    <chr> <chr>    <dbl> <chr>   <chr>   <chr>   <chr> <dbl> <chr>   <chr>  <chr>
##  1 s__u~ S12       0    S12     CN      CN      Fema~   8.6 k__Arc~ p__Eu~ c__M~
##  2 s__B~ S12       0    S12     CN      CN      Fema~   8.6 k__Bac~ p__Ac~ c__A~
##  3 s__B~ S12       0    S12     CN      CN      Fema~   8.6 k__Bac~ p__Ac~ c__A~
##  4 s__B~ S12       0    S12     CN      CN      Fema~   8.6 k__Bac~ p__Ac~ c__A~
##  5 s__C~ S12       0    S12     CN      CN      Fema~   8.6 k__Bac~ p__Ac~ c__A~
##  6 s__C~ S12       0    S12     CN      CN      Fema~   8.6 k__Bac~ p__Ac~ c__A~
##  7 s__u~ S12       0    S12     CN      CN      Fema~   8.6 k__Bac~ p__Ac~ c__A~
##  8 s__u~ S12       0    S12     CN      CN      Fema~   8.6 k__Bac~ p__Ac~ c__A~
##  9 s__B~ S12       6.34 S12     CN      CN      Fema~   8.6 k__Bac~ p__Ba~ c__B~
## 10 s__B~ S12       0    S12     CN      CN      Fema~   8.6 k__Bac~ p__Ba~ c__B~
## # ... with 4,360 more rows, 3 more variables: Order <chr>, Family <chr>,
## #   Genus <chr>, and abbreviated variable names 1: Abundance, 2: unknown1,
## #   3: response
```

3.3.1 Alpha diversity analysis in MGS (metagenomics sequencing) level

The metric of metagenomics data usually is relative abundance. But 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 means 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
```

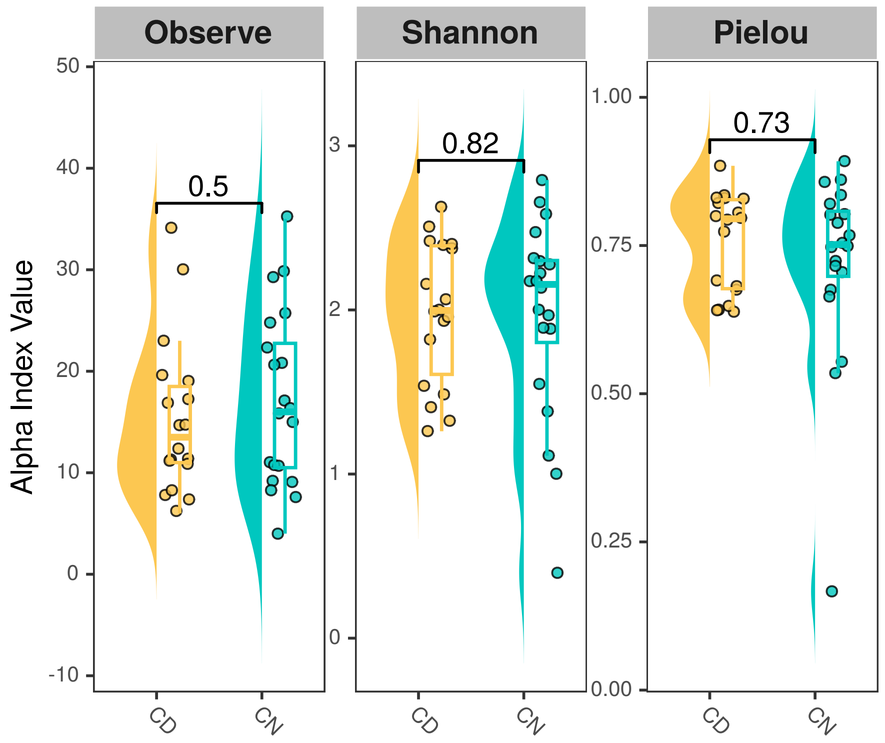


Fig. SA.26: **The alpha diversity boxplot based on MGS data**

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.SA.27.A) and boxplot (Fig.SA.27.B), then the distance was used to perform the PCoA analysis (Fig.SA.27.C).

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) %>% mp_fortify()
```

```
## # A tibble: 4 x 6
##   factors     Df SumOfSqs     R2     F `Pr(>F)`
##   <chr>    <dbl>    <dbl>  <dbl> <dbl>    <dbl>
## 1 disease      1    0.406 0.0370  1.38    0.156
## 2 response     1    0.308 0.0280  1.05    0.395
## 3 Residual    35   10.3   0.935  NA      NA
## 4 Total       37   11.0   1      NA      NA
```

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.SA.28).

```
mpseMGS %<>%
    mp_diff_analysis(
       .abundance = Abundance,
       force = TRUE,
```

```r
       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(
         data = td_filter(nodeClass == 'Phylum'),
         mapping = aes(
           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(Sign_disease)),
         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 extracted the abundance of the different species, then using ggplot2 (Wickham 2011) to visualize them (Fig.SA.29).

```
deT <- mpseMGS %>% mp_extract_tree() %>% dplyr::filter(!is.na(Sign_disease) & isTip, keep.td=F) %>% dplyr::pull
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 other 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 | Taxonomy=NULL
##    OTU     Sample Abund~1 Region Trans~2 Habitat Decid~3 Everg~4 Grass~5 Mixed~6
##    <chr>   <chr>    <int> <chr>  <chr>   <chr>     <dbl>   <dbl>   <dbl>   <dbl>
##  1 Cx.sal  DU1.1       19 Durham DU1     Field      125.   2321.  28734.   0.333
##  2 Ae.albo DU1.1        0 Durham DU1     Field      125.   2321.  28734.   0.333
##  3 Ae.cin  DU1.1        1 Durham DU1     Field      125.   2321.  28734.   0.333
##  4 Ae.vex  DU1.1       16 Durham DU1     Field      125.   2321.  28734.   0.333
##  5 Ps.fer  DU1.1        1 Durham DU1     Field      125.   2321.  28734.   0.333
##  6 Cx.err  DU1.1      372 Durham DU1     Field      125.   2321.  28734.   0.333
##  7 Ps.col  DU1.1      104 Durham DU1     Field      125.   2321.  28734.   0.333
##  8 Ae.tris DU1.1        0 Durham DU1     Field      125.   2321.  28734.   0.333
##  9 Cx.pip~ DU1.1        2 Durham DU1     Field      125.   2321.  28734.   0.333
## 10 Ae.can  DU1.1        0 Durham DU1     Field      125.   2321.  28734.   0.333
## # ... with 1,025 more rows, 6 more variables: ShrubScrub <dbl>,
## #   BarrenLand <dbl>, Building <dbl>, Pavement <dbl>, CultivatedCrops <dbl>,
## #   TrapNights <int>, and abbreviated variable names 1: Abundance, 2: Transect,
## #   3: DeciduousForest, 4: EvergreenForest, 5: Grassland, 6: MixedForest
```

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 | Taxonomy=NULL
##    OTU     Sample Abund~1 Region Trans~2 Habitat Decid~3 Everg~4 Grass~5 Mixed~6
##    <chr>   <chr>    <int> <chr>  <chr>   <fct>     <dbl>   <dbl>   <dbl>   <dbl>
##  1 Cx.sal  DU1.1       19 Durham DU1     Field      125.   2321.  28734.   0.333
```

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

```
##  2 Ae.albo DU1.1        0 Durham DU1     Field      125.   2321.  28734.   0.333
##  3 Ae.cin  DU1.1        1 Durham DU1     Field      125.   2321.  28734.   0.333
##  4 Ae.vex  DU1.1       16 Durham DU1     Field      125.   2321.  28734.   0.333
##  5 Ps.fer  DU1.1        1 Durham DU1     Field      125.   2321.  28734.   0.333
##  6 Cx.err  DU1.1      372 Durham DU1     Field      125.   2321.  28734.   0.333
##  7 Ps.col  DU1.1      104 Durham DU1     Field      125.   2321.  28734.   0.333
##  8 Ae.tris DU1.1        0 Durham DU1     Field      125.   2321.  28734.   0.333
##  9 Cx.pip~ DU1.1        2 Durham DU1     Field      125.   2321.  28734.   0.333
## 10 Ae.can  DU1.1        0 Durham DU1     Field      125.   2321.  28734.   0.333
## # ... with 1,025 more rows, 6 more variables: ShrubScrub <dbl>,
## #   BarrenLand <dbl>, Building <dbl>, Pavement <dbl>, CultivatedCrops <dbl>,
## #   TrapNights <int>, and abbreviated variable names 1: Abundance, 2: Transect,
## #   3: DeciduousForest, 4: EvergreenForest, 5: Grassland, 6: MixedForest
```

```
# 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.SA.30).

```
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.SA.31).

```
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 | Taxonomy=NULL
##    OTU     Sample Abund~1 NormA~2 Region Trans~3 Habitat Decid~4 Everg~5 Grass~6
##    <chr>   <chr>    <int>   <dbl> <chr>  <chr>   <fct>     <dbl>   <dbl>   <dbl>
##  1 Cx.sal  DU1.1       19   0.436 Durham DU1     Field      125.   2321.  28734.
##  2 Ae.albo DU1.1        0   0     Durham DU1     Field      125.   2321.  28734.
##  3 Ae.cin  DU1.1        1   0.1   Durham DU1     Field      125.   2321.  28734.
##  4 Ae.vex  DU1.1       16   0.4   Durham DU1     Field      125.   2321.  28734.
##  5 Ps.fer  DU1.1        1   0.1   Durham DU1     Field      125.   2321.  28734.
##  6 Cx.err  DU1.1      372   1.93  Durham DU1     Field      125.   2321.  28734.
##  7 Ps.col  DU1.1      104   1.02  Durham DU1     Field      125.   2321.  28734.
##  8 Ae.tris DU1.1        0   0     Durham DU1     Field      125.   2321.  28734.
##  9 Cx.pip~ DU1.1        2   0.141 Durham DU1     Field      125.   2321.  28734.
## 10 Ae.can  DU1.1        0   0     Durham DU1     Field      125.   2321.  28734.
## # ... with 1,025 more rows, 16 more variables: 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>, and
## #   abbreviated variable names 1: Abundance, 2: NormAbun, 3: Transect,
## #   4: DeciduousForest, 5: EvergreenForest, 6: Grassland
```

The raw result of pCCA was added the *internal_attr*, which can be extracted 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) %>% mp_fortify()
```

```
## # A tibble: 5 x 7
##   label             CCA1   CCA2    CCA3     r  pvals type
##   <chr>            <dbl>  <dbl>   <dbl> <dbl>  <dbl> <chr>
## 1 DeciduousForest  0.265  0.557 -0.0120 0.380 0.002  vectors
## 2 EvergreenForest  0.682 -0.258 -0.153  0.556 0.0001 vectors
## 3 Grassland       -0.830 -0.181  0.0139 0.722 0.0001 vectors
## 4 MixedForest      0.339 -0.114  0.256  0.194 0.0929 vectors
## 5 ShrubScrub      -0.377  0.117 -0.322  0.259 0.0537 vectors
```

Then we used *mp_plot_ord* to visualize the result of pCCA (Fig.SA.31).

```
# 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),
     )
ff <- aplot::plot_list(f, p, tag_levels="A")
ff
```

4.4 The distribution of Mosquito species in the study.

We used *mp_cal_abundance* and *mp_plot_abundance* to calculate and visualize the abundance of the Mosquito species in the study (Fig.SA.32).

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

Then We can use *mp_diff_analysis* to identify the significant differential species 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 & !is.na(Sign_Habitat)) %>%
    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 16]>      4.96    4.92     4.87 Field        0.00705
##  2 Ae.albo <tibble [18 x 16]>      4.83    4.79     4.75 Forest       0.000229
##  3 Ae.cin  <tibble [18 x 16]>      4.36    4.31     4.25 Forest       0.0159
##  4 Ps.fer  <tibble [18 x 16]>      4.94    4.90     4.87 Forest       0.00122
##  5 Ps.col  <tibble [18 x 16]>      5.26    5.24     5.22 Field        0.000327
##  6 Ae.tris <tibble [18 x 16]>      4.49    4.46     4.43 Forest       0.000530
##  7 Ae.can  <tibble [18 x 16]>      4.28    4.24     4.19 Forest       0.0119
##  8 Ae.hen  <tibble [18 x 16]>      4.28    4.23     4.18 Forest       0.000483
##  9 Ae.atl  <tibble [18 x 16]>      4.59    4.56     4.52 Forest       0.00311
## 10 Ae.dup  <tibble [18 x 16]>      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 METHODS

5.1 The MPSE class

To better store the input data (abundance data, sequence data, phylogenetic tree data) and the result of downstream analysis, *MPSE* class was implemented in the *MicrobiotaProcess* package. This class inherits the *SummarizedExperiment* (Morgan et al. 2021) class. In which, the assays slot was designed to store the rectangular abundance matrices of features (microbiota profiling or function profiling) for microbiome experiment results. The *colData* slot was designed to store the meta-data of features and results about the features generated in the downstream analysis. Compared to the *SummarizedExperiment* (Morgan et al. 2021) class, *MPSE* introduces the following additional slots, 1) the *otutree* slot is a *treedata* object and was designed to store the phylogenetic tree and the associated data, including the results of the features in the downstream analysis and the evolutionary statistics inferred by the software building the tree; 2) the *taxatree* slot is also a *treedata* (Yu 2021; Wang et al. 2020) object and was designed to store hierarchical taxonomy relationships and the associated data, such as the relative abundances and the results from different analysis; 3) the *refseq* slot is a *XStringSet* (Pagès et al. 2021) object and was designed to store the reference sequences, with names corresponding to the rows of the *assays* slot. In addition, *an internal attribute internal_attr* (a *list* object) was introduced to store the raw results of pca (Principal Components Analysis), pcoa (Principal Coordinate Analysis), and hierarchical cluster analysis.

5.2 Overview of the design of the MicrobiotaProcess package

The overall design of the *MicrobiotaProcess* package was illustrated in **Figure 2**. It presents multiple parser functions to read the outputs of upstream analysis tools, such as qiime or qiime2 (Bolyen et al. 2019) , dada2 (Callahan et al. 2016), and MetaPhlAn (Truong et al. 2015). After parsing, the abundance of microbiota (or other features), the metadata of the sample (optional), and the phylogenetic tree information (optional) are extracted from the outputs and stored as an *MPSE* object. Other objects designed to store microbiome data, such as phyloseq (McMurdie 2013), TreeSummarizedExperiment (Huang et al. 2021), and SummarizedExperiment (Morgan et al. 2021) can be converted to an *MPSE* object. This enables *MPSE* to serve as a standardized entry point for downstream analysis while being compatible with existing analysis software and pipelines. *MicrobiotaProcess* provides a wide variety of microbiome analysis procedures to work with *MPSE* objects. These procedures were designed to follow the tidy data principles and thus are human-friendly, consistent, and composable to solve complicated problems. The results of the analysis can be returned in three modes via the action argument. The (intermediate) result can be stored in the *MPSE* object if the action is 'add'. If the action is 'only', it returns a tidy data frame with non-redundant sample or OTU (features) information with the result. While the action is 'get' returns the analysis result only. *MicrobiotaProcess* also extends the *dplyr* package to offer dplyr-verbs for data operation.

5.3 Parser functions and the MPSE constructor

The *MicrobiotaProcess* package provides *mp_import_qiime2* to load the output of *qiime2* (Bolyen et al. 2019), which is a common tool for the analysis of amplicon data. The feature abundance table of the output (i.e., a qza format file) of qiime2 is required while the taxonomy information, phylogenetic tree, and representative sequences are optional. The *mp_import_dada2* function was designed to parse the output of dada2 (Callahan et al. 2016). The output of *removeBimeraDenovo* of *dada2* is required, while the taxonomy information, representation phylogenetic tree, and representative sequences are optional. The *mp_import_metaphlan* function was designed to parse the output of *MetaPhlAn* (Truong et al. 2015), which is a common tool for profiling the composition of microbial communities. The microbiota abundance output of *MetaPhlAn* (Truong et al. 2015) is required, while the phylogenetic tree and metadata are also optional. In addition, An *MPSE* object can be constructed from scratch by calling the *MPSE* function with the following key parameters: 1) *assays* (required): A list or *SimpleList* of matrix-like objects or a matrix-like object (rows represent the features and columns represent the samples) providing abundance data for all samples. 2) *colData* (optional): A *DataFrame* object storing the characteristics of the samples. 3) *otutree* (optional): A *treedata* object storing a phylogenetic tree with/without associated data. Any tree file formats as well as commonly used software outputs that can be parsed by treeio15 are supported. 4) *taxatree* (optional): A *treedata* object storing the taxonomy information (or other hierarchical data). *MicrobiotaProcess* provides the *convert_to_treedata* function to convert taxonomy data (a data.frame object) to a treedata object. 5) *refseq* (optional): A *XStingSet* object storing the representative sequences. Both nucleic acid sequences and amino acid sequences are supported via the *readDNAStringSet* or *readAAStringSet* functions provided by the *Biostrings* packages.

5.4 Process MPSE object using dplyr-verbs

To facilitate data manipulation and exploration of the microbiome data, *MicrobiotaProcess* defined a tidy-like formatted output for the *MPSE* object and extended a subset of the dplyr-verbs to support the *MPSE* object. The extended dplyr-verbs include: 1) *filter* function: subset *MPSE* class, retaining the data that satisfy the provided conditions. 2) *select* function: select the data according to the provided variables. 3) *group_by* function: return a grouped tbl_df-like data frame according to groups defined by the provided variables, then some data operations can be done on the groups. 4) *mutate* function: create new columns according to the provided variables and conditions. 5) *left_join* function: add columns from 'y' (a data.frame object) to 'x' (an *MPSE* object), by matching all rows of 'x' based on the keys. The keys only should be one or all of the Sample or OTU. 6) *rename* function: rename the column names of an *MPSE* object, except the OTU, Sample, and Abundance column names which cannot be renamed.

5.5 Data preprocessing

The microbiome features (OTU or ASV) with very low abundance and rare occurrence (exits in few samples) are difficult to distinguish from the sequencing error or other experimental technical errors and are usually uninformative. It is better to improve the statistical power of multiple testing in the downstream analysis by filtering these features. *MicrobiotaProcess* presents *mp_filter_taxa* to filter the features based on their abundance (default Abundance count) and sample prevalence. By default, this function will screen out the features with zeros counts in 0.05% sample prevalence and users can reset the criteria via the parameters of *min.abun* (minimum abundance in a sample) and *min.prop* (the minimum sample prevalence). To make the data more meaningful for the downstream analysis after filtering, *MicrobiotaProcess* provides the *mp_decostand* and *mp_rrarefy* functions for the standardization of community data by inheriting the *decostand* and *rrarefy* functions from vegan (Oksanen et al. 2020). The *mp_rrarefy* function allows users to estimate expected diversity (e.g., taxonomic richness) for a reduced sampling size, while the *mp_decostand* function provides several standardization methods for community data, such as total (divide by total abundance of each sample (relative abundance)), max (divide by the max abundance of the feature in all samples), frequency (divide by total abundance of each sample and multiply the number of non-zero features), hellinger (square root for the result of total) (Legendre and Gallagher 2001), log (logarithmic transformation log_b(x > 0) + 1) (Anderson, Ellingsen, and McArdle 2006). More importantly, we developed the mp_balance_clade method to convert the abundance of species to the balances of internal nodes with the geometric mean, mean or median abundance of the offspring tips in the same clade of the phylogenetic tree. This will convert the compositional microbiota data to an unconstrained coordinate system effectively and may improve the identification of differential clades by accumulating the small consilient differences at a higher resolution on the phylogenetic tree (Morton et al. 2017; Egozcue et al. 2003). These functions are developed to follow the tidiness concept and the results will be added to the assays slot of the MPSE object automatically to enhance reproducibility and reuse in the follow-up analysis.

5.6 Alpha diversity

Alpha diversity measures the species richness and evenness within a community. *MicrobiotaProcess* provides the *mp_cal_alpha* function to calculate the community diversity. There are six commonly used methods to calculate alpha diversity, including Observe (calculates the total species per sample), Chao1, ACE (estimate species richness by considering the low abundance of species). Pielou (measures the species' evenness), Shannon and Simpson (take both the richness and evenness of species into account). By default, the *mp_cal_alpha* will rarefy the abundance before calculating the alpha diversity. Users can specify the force argument to TRUE to calculate the diversity directly without performing rarefaction. This will be useful for taxonomic profiling data since they are usually in relative abundance and cannot be rarefied. By default, the results are added to the *colData* slot that stored the metadata information of samples and returns an updated *MPSE* object. *MicrobiotaProcess* also provides the *mp_cal_pd_*metric function to calculate several phylogenetic community structure metrics, such as PD (Faith's Phylogenetic Diversity), NRI (Nearest Relative Index), NTI (Nearest Taxon Index), IAC (Relative deviation from the null expectation of phylogenetically balanced abundances), PAE (Phylogenetic evenness of the abundance distribution scaled by branch lengths), HAED (Entropic measure of the diversity of evolutionary distinctiveness among individuals), EAED (Equitability of HAED). These metrics provide the measures of the phylogenetic diversity incorporating the species abundance of community, which can help users to understand the impact of phylogenetic history on the corresponding microbiota ecological interactions (Webb 2000; Cadotte et al. 2010). PAE, HAED, EAED, and IAC can be used to evaluate the structure of a phylogeny of assemblage communities by incorporating the species abundance of the community, NRI and NTI can be used to examine whether an observed assembly of communities is a phylogenetically biased subset of the species that could coexist in that assemblage (Webb 2000). The *mp_plot_alpha* function is implemented to visualize alpha diversity and it allows comparing different communities that were specified via the *.group* parameter.

5.7 Taxonomy composition

To compare the difference in OTUs (features) composition between different communities, *MicrobiotaProcess* provides the *mp_cal_upset* and *mp_cal_venn* to calculate the conjunct OTUs (features) or specific OTUs (features) of different groups (specified by the *.group* parameter). The result can be visualized by the *mp_plot_upset* and *mp_plot_venn* functions respectively. The microbiome OTUs (features) are often annotated to different taxonomy levels in upstream analyses, and to survey the species profile of different samples, it is often necessary to calculate the abundances of different taxonomy levels. *MicrobiotaProcess* implements the *mp_cal_abundance* function to calculate the abundances of all taxonomy levels. Similar to the calculation of alpha diversity, the *mp_cal_abundance* will rarefy the raw abundance and then calculate the relative abundance by default. Users can specify the *force* argument to *TRUE* to disable rarefaction. And the relative argument controls whether to calculate the relative abundance (total is 100% for the same taxonomy level). The results will be added to the associated data of the *taxatree* (*treedata* object) slot by default. The abundance of a selected taxonomy level can be extracted by the *mp_extract_abundance* function with the *taxa.class* parameter specified. The results of the *mp_cal_abundance* can be visualized by the *mp_plot_abundance* function.

5.8 Beta diversity

The beta diversity has been applied in a broad sense to measure variation or changes in community composition. It can assess how microbiota composition changes across spatial and temporal scales. Some distance indexes, such as the Bray-Curtis index, Jaccard index, and UniFrac (weighted or unweighted) index, are useful and popular to measure the degree of community differentiation. These distances can be further subjected to ordination which aims to capture essential information in a lower-dimensional representation and is commonly used to visualize sample dissimilarities. *MicrobiotaProcess* implements the *mp_cal_dist* function to compute the common distances (dissimilarity) and provides the mp_plot_dist function to visualize the result. It also provides several commonly-used ordination methods, such as Principal Components Analysis (PCA: *mp_cal_pca*), Principal Coordinate Analysis (PCoA: *mp_cal_pcoa*), Nonmetric Multidimensional Scaling (NMDS: *mp_cal_nmds*), Detrended Correspondence Analysis (DCA: *mp_cal_dca*), Redundancy Analysis (RDA: *mp_cal_rda*), and (Constrained) Correspondence Analysis (CCA: *mp_cal_cca*). To fit environmental vectors or factors onto an ordination, this package provides the *mp_envfit* function to perform this analysis. All the ordination results can be visualized by the *mp_plot_ord* function. In addition, it also wraps several statistical analyses for the distance matrices, such as permutational multivariate analysis of variance (*mp_adonis*), analysis of similarities (*mp_anosim*), and multi-response permutation procedure (*mp_mrpp*), and mantel tests for dissimilarity matrices (*mp_mantel*). All these functions are developed based on a tidy-like framework. These functions can be assembled into linear workflows with the pipe operator (%>% or |>).

5.9 Differential analysis and biomarker discovery

MicrobiotaProcess implements the *mp_diff_analysis* function for identifying differentially abundant genera as biomarkers based on the tidy-like framework. Similar to LEfSe(Segata et al. 2011), there are three steps to perform this analysis. First, all features are tested to determine whether values (e.g., abundance) in different groups of samples are differentially distributed via the Kruskal-Wallis rank-sum test (default, other option is oneway.test, glm or glm.nb). Then, the resulting features infringing the null hypothesis (using the FDR to filter by default which is different with LEfSe) are further tested by the second round of the test using Wilcoxon rank-sum test (default, another option is t.test, glm or glm.nb) to keep the features that in all pairwise comparisons between the sub-groups are significantly consistent with the group level trend. Finally, the linear discriminant analysis (LDA) or random forest model was built to rank all the features based on the relative difference among different groups. Compare to LEfSe, *mp_diff_analysis* is more flexible. Not only the test method but also the test value (using generalized fold change (Wirbel et al. 2019) by default, another option is comparing the median or mean value of different groups by using *compare_median* or *compare_mean*) can be set to return which group has more abundant the significant features by users. The result is integrated into the *taxatree* component (default) or *rowData* component depending on whether the taxonomy is provided or not. The result can be extracted via the *mp_extract_tree* or *mp_extract_feature* respectively. Then it can be processed and displayed via treeio (Wang et al. 2020), tidytree (Yu 2021), ggtree (Yu et al. 2017), ggtreeExtra (Xu et al. 2021), and ggplot2 (Figure 4A and Figure 7). To decrease the coding burden, we also developed *mp_plot_diff_boxplot*, *mp_plot_diff_manhattan*, *mp_plot_diff_res*, and *mp_plot_diff_cladogram* to visualize the result of differential analysis (Figure 3F). The evaluation of simulation dataset and real datasets between the *mp_diff_analysis* and other tools are available in the supplemental B.

5.10 Accessors to fetch internal data

The *MPSE* object is composed of several objects to store different data including primary data and analysis results. To extract the components of the data, *MicrobiotaProcess* provides several accessors starting with *mp_extract_*, including: 1) *mp_extract_sample* function: to return the sample characteristics in a tidy data table, similar to the *colData* function for the *SummarizedExperiment* class. 2) *mp_extract_assays* function: to extract the assays from an *MPSE* object, similar to the *assay* function for the *SummarizedExperiment* class. 3) *mp_extract_feature* function: to extract the features characteristics (optional with taxonomy information by setting *addtaxa* argument to *TRUE*) and return tidy data, similar to the *rowData* function for the *SummarizedExperiment* class. 4) *mp_extract_tree* function: to extract the *taxatree* (by default) or *otutree* (by specifying the type argument to *otutree*) from an *MPSE* object. 5) *mp_extract_taxonomy* function: to extract the taxonomy information from an *MPSE* object. 6) *mp_extract_refseq* function: to extract the representative sequences from an *MPSE* object. 7) *mp_extract_dist* function: to extract distances in a matrix (as a dist object, by default) or a tidy data frame with comparison among the groups (by specifying the *.group* argument). 8) *mp_extract_rarecurve* function: to extract rarefaction in a *rarecurve* object, which can be visualized by *ggrarecurve* function. 9) *mp_extract_internal_attr* function: to extract the raw result of *mp_cal_pca*, *mp_cal_pcoa*, *mp_cal_rda*, *mp_cal_cca*, *mp_cal_clust*, *mp_envfit*, *mp_adonis*, *mp_anosim*, *mp_mrpp* and *mp_mantel*.

6 Session information

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

```
## - Session info ------------------------------------------------------------------------------
##  setting  value
##  version  R version 4.2.0 (2022-04-22)
##  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     2023-01-30
##  pandoc   2.9.2 @ /usr/bin/ (via rmarkdown)
##
## - Packages -------------------------------------------------------------------------------------------------
##  package                  * version    date (UTC) lib source
##  abind                      1.4-5      2016-07-21 [1] CRAN (R 4.2.0)
##  ade4                       1.7-19     2022-04-19 [1] CRAN (R 4.2.0)
##  AnnotationDbi              1.58.0     2022-04-26 [1] Bioconductor
##  AnnotationHub              3.4.0      2022-04-26 [1] Bioconductor
##  ape                        5.6-3      2022-10-30 [1] Github (emmanuelparadis/ape@090e82c)
##  aplot                    * 0.1.8      2022-11-17 [1] local
##  assertthat                 0.2.1      2019-03-21 [1] CRAN (R 4.2.0)
##  attempt                    0.3.1      2020-05-03 [1] CRAN (R 4.2.0)
##  backports                  1.4.1      2021-12-13 [1] CRAN (R 4.2.0)
##  beachmat                   2.12.0     2022-04-26 [1] Bioconductor
##  beeswarm                   0.4.0      2021-06-01 [1] CRAN (R 4.2.0)
##  Biobase                  * 2.56.0     2022-04-26 [1] Bioconductor
##  BiocFileCache              2.4.0      2022-04-26 [1] Bioconductor
##  BiocGenerics             * 0.42.0     2022-04-26 [1] Bioconductor
##  BiocManager                1.30.18    2022-05-18 [1] CRAN (R 4.2.0)
##  BiocNeighbors              1.14.0     2022-04-26 [1] Bioconductor
##  BiocParallel               1.30.3     2022-06-05 [1] Bioconductor
##  BiocSingular               1.12.0     2022-04-26 [1] Bioconductor
##  BiocVersion                3.15.2     2022-03-29 [1] Bioconductor
##  biomformat                 1.24.0     2022-04-26 [1] Bioconductor
##  Biostrings               * 2.64.1     2022-08-18 [1] Bioconductor
##  bit                        4.0.4      2020-08-04 [1] CRAN (R 4.2.0)
##  bit64                      4.0.5      2020-08-30 [1] CRAN (R 4.2.0)
##  bitops                     1.0-7      2021-04-24 [1] CRAN (R 4.2.0)
##  blob                       1.2.3      2022-04-10 [1] CRAN (R 4.2.0)
##  bookdown                   0.29       2022-09-12 [1] CRAN (R 4.2.0)
##  boot                       1.3-28     2021-05-03 [1] CRAN (R 4.2.0)
##  broom                      1.0.0      2022-07-01 [1] CRAN (R 4.2.0)
##  bslib                      0.4.0      2022-07-16 [1] CRAN (R 4.2.0)
##  cachem                     1.0.6      2021-08-19 [1] CRAN (R 4.2.0)
##  car                        3.1-0      2022-06-15 [1] CRAN (R 4.2.0)
##  carData                    3.0-5      2022-01-06 [1] CRAN (R 4.2.0)
##  class                      7.3-20     2022-01-16 [1] CRAN (R 4.2.0)
##  classInt                   0.4-7      2022-06-10 [1] CRAN (R 4.2.0)
##  cli                        3.4.1      2022-09-23 [1] CRAN (R 4.2.0)
##  clue                       0.3-61     2022-05-30 [1] CRAN (R 4.2.0)
##  cluster                    2.1.3      2022-03-28 [1] CRAN (R 4.2.0)
##  clusterProfiler          * 4.5.2      2022-09-06 [1] Bioconductor
##  codetools                  0.2-18     2020-11-04 [1] CRAN (R 4.2.0)
##  coin                     * 1.4-2      2021-10-08 [1] CRAN (R 4.2.0)
##  colorspace                 2.0-3      2022-02-21 [1] CRAN (R 4.2.0)
##  config                     0.3.1      2020-12-17 [1] CRAN (R 4.2.0)
##  corrr                      0.4.3      2020-11-24 [1] CRAN (R 4.2.0)
##  crayon                     1.5.1      2022-03-26 [1] CRAN (R 4.2.0)
##  curatedMetagenomicData   * 3.4.2      2022-05-19 [1] Bioconductor
##  curl                       4.3.2      2021-06-23 [1] CRAN (R 4.2.0)
##  cvTools                    0.3.2      2012-05-14 [1] CRAN (R 4.2.0)
##  data.table                 1.14.2     2021-09-27 [1] CRAN (R 4.2.0)
##  DBI                        1.1.3      2022-06-18 [1] CRAN (R 4.2.0)
##  dbplyr                     2.2.1      2022-06-27 [1] CRAN (R 4.2.0)
##  DECIPHER                   2.24.0     2022-04-26 [1] Bioconductor
##  decontam                   1.16.0     2022-04-26 [1] Bioconductor
##  DelayedArray               0.22.0     2022-04-26 [1] Bioconductor
##  DelayedMatrixStats         1.18.0     2022-04-26 [1] Bioconductor
##  DEoptimR                   1.0-11     2022-04-03 [1] CRAN (R 4.2.0)
##  desc                       1.4.2      2022-09-08 [1] CRAN (R 4.2.0)
##  deSolve                    1.33       2022-07-16 [1] CRAN (R 4.2.0)
##  digest                     0.6.30     2022-10-18 [1] CRAN (R 4.2.0)
##  diptest                    0.76-0     2021-05-04 [1] CRAN (R 4.2.0)
##  DirichletMultinomial       1.38.0     2022-04-26 [1] Bioconductor
##  DO.db                      2.9        2022-09-06 [1] Bioconductor
##  DOSE                       3.23.2.001 2022-09-06 [1] Bioconductor
##  downloader                 0.4        2015-07-09 [1] CRAN (R 4.2.0)
##  dplyr                    * 1.0.10     2022-09-01 [1] CRAN (R 4.2.0)
##  DT                         0.25       2022-09-12 [1] CRAN (R 4.2.0)
##  dtplyr                     1.2.1      2022-01-19 [1] CRAN (R 4.2.0)
##  e1071                      1.7-11     2022-06-07 [1] CRAN (R 4.2.0)
##  edgeR                    * 3.38.4     2022-08-07 [1] Bioconductor
##  ellipsis                   0.3.2      2021-04-29 [1] CRAN (R 4.2.0)
##  enrichplot               * 1.17.0.995 2022-09-06 [1] Bioconductor
##  evaluate                   0.16       2022-08-09 [1] CRAN (R 4.2.0)
##  ExperimentHub              2.4.0      2022-04-26 [1] Bioconductor
##  fansi                      1.0.3      2022-03-24 [1] CRAN (R 4.2.0)
##  farver                     2.1.1      2022-07-06 [1] CRAN (R 4.2.0)
##  fastmap                    1.1.0      2021-01-25 [1] CRAN (R 4.2.0)
##  fastmatch                  1.1-3      2021-07-23 [1] CRAN (R 4.2.0)
##  fBasics                    4021.92    2022-08-08 [1] CRAN (R 4.2.0)
##  fda                        6.0.5      2022-07-04 [1] CRAN (R 4.2.0)
##  fds                        1.8        2018-10-31 [1] CRAN (R 4.2.0)
##  fgsea                      1.22.0     2022-04-26 [1] Bioconductor
##  filelock                   1.0.2      2018-10-05 [1] CRAN (R 4.2.0)
##  flexmix                    2.3-18     2022-06-07 [1] CRAN (R 4.2.0)
##  forcats                  * 0.5.1      2021-01-27 [1] CRAN (R 4.2.0)
##  foreach                    1.5.2      2022-02-02 [1] CRAN (R 4.2.0)
##  fpc                        2.2-9      2020-12-06 [1] CRAN (R 4.2.0)
##  fs                         1.5.2      2021-12-08 [1] CRAN (R 4.2.0)
##  generics                   0.1.3      2022-07-05 [1] CRAN (R 4.2.0)
##  GenomeInfoDb             * 1.32.4     2022-09-06 [1] Bioconductor
##  GenomeInfoDbData           1.2.8      2022-04-28 [1] Bioconductor
##  GenomicRanges            * 1.48.0     2022-04-26 [1] Bioconductor
##  ggalluvial                 0.12.3     2020-12-05 [1] CRAN (R 4.2.0)
##  GGally                     2.1.2      2021-06-21 [1] CRAN (R 4.2.0)
##  ggbeeswarm                 0.6.0      2017-08-07 [1] CRAN (R 4.2.0)
##  ggforce                    0.3.3      2021-03-05 [1] CRAN (R 4.2.0)
##  ggfortify                  0.4.14     2022-01-03 [1] CRAN (R 4.2.0)
##  ggfun                      0.0.6      2022-08-30 [1] local
##  ggh4x                      0.2.2      2022-08-14 [1] CRAN (R 4.2.0)
##  gghalves                   0.1.3      2022-05-30 [1] CRAN (R 4.2.0)
##  ggnewscale               * 0.4.7      2022-03-25 [1] CRAN (R 4.2.0)
##  ggplot2                  * 3.4.0      2022-11-04 [1] CRAN (R 4.2.0)
##  ggplotify                  0.1.0      2021-09-02 [1] CRAN (R 4.2.0)
##  ggpp                     * 0.4.5      2022-09-30 [1] CRAN (R 4.2.0)
##  ggraph                     2.0.6      2022-08-08 [1] CRAN (R 4.2.0)
##  ggrepel                  * 0.9.1      2021-01-15 [1] CRAN (R 4.2.0)
##  ggsci                    * 2.9        2018-05-14 [1] CRAN (R 4.2.0)
##  ggside                     0.2.1      2022-07-20 [1] CRAN (R 4.2.0)
##  ggsignif                   0.6.3      2021-09-09 [1] CRAN (R 4.2.0)
##  ggstar                   * 1.0.3      2021-12-03 [1] CRAN (R 4.2.0)
##  ggtree                   * 3.7.1      2022-11-10 [1] Bioconductor
##  ggtreeExtra              * 1.9.1.990  2022-11-24 [1] Bioconductor
##  ggupset                    0.3.0      2020-05-05 [1] CRAN (R 4.2.0)
##  ggVennDiagram          * 1.2.0    2021-10-22 [1] CRAN (R 4.2.0)
##  glue                     1.6.2    2022-02-24 [1] CRAN (R 4.2.0)
##  GO.db                    3.15.0   2022-09-06 [1] Bioconductor
##  golem                    0.3.5    2022-10-18 [1] CRAN (R 4.2.0)
##  GOSemSim                 2.22.0   2022-04-26 [1] Bioconductor
##  graphlayouts             0.8.0    2022-01-03 [1] CRAN (R 4.2.0)
##  gridExtra                2.3      2017-09-09 [1] CRAN (R 4.2.0)
##  gridGraphics             0.5-1    2020-12-13 [1] CRAN (R 4.2.0)
##  gson                     0.0.8    2022-08-20 [1] CRAN (R 4.2.0)
##  gtable                   0.3.1    2022-09-01 [1] CRAN (R 4.2.0)
##  GUniFrac               * 1.7      2022-10-23 [1] CRAN (R 4.2.0)
##  hdrcde                   3.4      2021-01-18 [1] CRAN (R 4.2.0)
##  hms                      1.1.1    2021-09-26 [1] CRAN (R 4.2.0)
##  htmltools                0.5.3    2022-07-18 [1] CRAN (R 4.2.0)
##  htmlwidgets              1.5.4    2021-09-08 [1] CRAN (R 4.2.0)
##  httpuv                   1.6.6    2022-09-08 [1] CRAN (R 4.2.0)
##  httr                     1.4.4    2022-08-17 [1] CRAN (R 4.2.0)
##  igraph                   1.3.4    2022-07-19 [1] CRAN (R 4.2.0)
##  interactiveDisplayBase   1.34.0   2022-04-26 [1] Bioconductor
##  IRanges                * 2.30.1   2022-08-18 [1] Bioconductor
##  irlba                    2.3.5    2021-12-06 [1] CRAN (R 4.2.0)
##  iterators                1.0.14   2022-02-05 [1] CRAN (R 4.2.0)
##  jquerylib                0.1.4    2021-04-26 [1] CRAN (R 4.2.0)
##  jsonlite                 1.8.0    2022-02-22 [1] CRAN (R 4.2.0)
##  kableExtra             * 1.3.4    2021-02-20 [1] CRAN (R 4.2.0)
##  KEGGREST                 1.36.3   2022-07-12 [1] Bioconductor
##  kernlab                  0.9-31   2022-06-09 [1] CRAN (R 4.2.0)
##  KernSmooth               2.23-20  2021-05-03 [1] CRAN (R 4.2.0)
##  knitr                    1.39     2022-04-26 [1] CRAN (R 4.2.0)
##  ks                       1.13.5   2022-04-14 [1] CRAN (R 4.2.0)
##  labeling                 0.4.2    2020-10-20 [1] CRAN (R 4.2.0)
##  laeken                   0.5.2    2021-10-06 [1] CRAN (R 4.2.0)
##  later                    1.3.0    2021-08-18 [1] CRAN (R 4.2.0)
##  lattice                  0.20-45  2021-09-22 [1] CRAN (R 4.2.0)
##  lazyeval                 0.2.2    2019-03-15 [1] CRAN (R 4.2.0)
##  libcoin                  1.0-9    2021-09-27 [1] CRAN (R 4.2.0)
##  lifecycle                1.0.3    2022-10-07 [1] CRAN (R 4.2.0)
##  limma                  * 3.52.3   2022-09-11 [1] Bioconductor
##  lme4                     1.1-30   2022-07-08 [1] CRAN (R 4.2.0)
##  lmerTest                 3.1-3    2020-10-23 [1] CRAN (R 4.2.0)
##  lmtest                   0.9-40   2022-03-21 [1] CRAN (R 4.2.0)
##  locfit                   1.5-9.6  2022-07-11 [1] CRAN (R 4.2.0)
##  magrittr                 2.0.3    2022-03-30 [1] CRAN (R 4.2.0)
##  MASS                     7.3-57   2022-04-22 [1] CRAN (R 4.2.0)
##  Matrix                   1.4-1    2022-03-23 [1] CRAN (R 4.2.0)
##  MatrixGenerics         * 1.8.1    2022-06-26 [1] Bioconductor
##  matrixStats            * 0.62.0   2022-04-19 [1] CRAN (R 4.2.0)
##  mclust                   5.4.10   2022-05-20 [1] CRAN (R 4.2.0)
##  memoise                  2.0.1    2021-11-26 [1] CRAN (R 4.2.0)
##  mgcv                     1.8-40   2022-03-29 [1] CRAN (R 4.2.0)
##  mia                      1.4.0    2022-04-26 [1] Bioconductor
##  MicrobiomeProfiler     * 1.1.0    2022-11-08 [1] Bioconductor
##  MicrobiomeStat         * 1.1      2022-01-24 [1] CRAN (R 4.2.0)
##  MicrobiotaProcess      * 1.11.4   2023-01-30 [1] Bioconductor
##  mime                     0.12     2021-09-28 [1] CRAN (R 4.2.0)
##  minqa                    1.2.4    2014-10-09 [1] CRAN (R 4.2.0)
##  modeest                  2.4.0    2019-11-18 [1] CRAN (R 4.2.0)
##  modeltools               0.2-23   2020-03-05 [1] CRAN (R 4.2.0)
##  multcomp                 1.4-19   2022-04-26 [1] CRAN (R 4.2.0)
##  MultiAssayExperiment     1.22.0   2022-04-26 [1] Bioconductor
##  multtest               2.52.0     2022-04-26 [1] Bioconductor
##  munsell                0.5.0      2018-06-12 [1] CRAN (R 4.2.0)
##  mvtnorm                1.1-3      2021-10-08 [1] CRAN (R 4.2.0)
##  NADA                   1.6-1.1    2020-03-22 [1] CRAN (R 4.2.0)
##  nlme                   3.1-158    2022-06-15 [1] CRAN (R 4.2.0)
##  nloptr                 2.0.3      2022-05-26 [1] CRAN (R 4.2.0)
##  nnet                   7.3-17     2022-01-16 [1] CRAN (R 4.2.0)
##  numDeriv               2016.8-1.1 2019-06-06 [1] CRAN (R 4.2.0)
##  patchwork            * 1.1.2      2022-08-19 [1] CRAN (R 4.2.0)
##  pcaPP                  2.0-2      2022-07-08 [1] CRAN (R 4.2.0)
##  permute                0.9-7      2022-01-27 [1] CRAN (R 4.2.0)
##  perry                  0.3.1      2021-11-03 [1] CRAN (R 4.2.0)
##  phyloseq               1.40.0     2022-04-26 [1] Bioconductor
##  pillar                 1.8.1      2022-08-19 [1] CRAN (R 4.2.0)
##  pkgconfig              2.0.3      2019-09-22 [1] CRAN (R 4.2.0)
##  pkgload                1.3.0      2022-06-27 [1] CRAN (R 4.2.0)
##  pls                    2.8-1      2022-07-16 [1] CRAN (R 4.2.0)
##  plyr                   1.8.7      2022-03-24 [1] CRAN (R 4.2.0)
##  png                    0.1-7      2013-12-03 [1] CRAN (R 4.2.0)
##  polyclip               1.10-0     2019-03-14 [1] CRAN (R 4.2.0)
##  prabclus               2.3-2      2020-01-08 [1] CRAN (R 4.2.0)
##  pracma                 2.3.8      2022-03-04 [1] CRAN (R 4.2.0)
##  preprocessCore         1.58.0     2022-04-26 [1] Bioconductor
##  pROC                   1.18.0     2021-09-03 [1] CRAN (R 4.2.0)
##  promises               1.2.0.1    2021-02-11 [1] CRAN (R 4.2.0)
##  proxy                  0.4-27     2022-06-09 [1] CRAN (R 4.2.0)
##  purrr                  0.3.5      2022-10-06 [1] CRAN (R 4.2.0)
##  qvalue                 2.28.0     2022-04-26 [1] Bioconductor
##  R6                     2.5.1      2021-08-19 [1] CRAN (R 4.2.0)
##  ragg                   1.2.2      2022-02-21 [1] CRAN (R 4.2.0)
##  rainbow                3.6        2019-01-29 [1] CRAN (R 4.2.0)
##  randomForest         * 4.7-1.1    2022-05-23 [1] CRAN (R 4.2.0)
##  ranger                 0.14.1     2022-06-18 [1] CRAN (R 4.2.0)
##  rappdirs               0.3.3      2021-01-31 [1] CRAN (R 4.2.0)
##  RColorBrewer           1.1-3      2022-04-03 [1] CRAN (R 4.2.0)
##  Rcpp                   1.0.9      2022-07-08 [1] CRAN (R 4.2.0)
##  RCurl                  1.98-1.8   2022-07-30 [1] CRAN (R 4.2.0)
##  readr                  2.1.2      2022-01-30 [1] CRAN (R 4.2.0)
##  reshape                0.8.9      2022-04-12 [1] CRAN (R 4.2.0)
##  reshape2               1.4.4      2020-04-09 [1] CRAN (R 4.2.0)
##  rhdf5                  2.40.0     2022-04-26 [1] Bioconductor
##  rhdf5filters           1.8.0      2022-04-26 [1] Bioconductor
##  Rhdf5lib               1.18.2     2022-05-15 [1] Bioconductor
##  rlang                  1.0.6      2022-09-24 [1] CRAN (R 4.2.0)
##  rmarkdown              2.15       2022-08-16 [1] CRAN (R 4.2.0)
##  rmutil                 1.1.9      2022-03-01 [1] CRAN (R 4.2.0)
##  robCompositions        2.3.1      2021-09-20 [1] CRAN (R 4.2.0)
##  robustbase             0.95-0     2022-04-02 [1] CRAN (R 4.2.0)
##  robustHD               0.7.3      2022-08-12 [1] CRAN (R 4.2.0)
##  roxygen2               7.2.1      2022-07-18 [1] CRAN (R 4.2.0)
##  rpart                  4.1.16     2022-01-24 [1] CRAN (R 4.2.0)
##  rprojroot              2.0.3      2022-04-02 [1] CRAN (R 4.2.0)
##  rrcov                  1.7-1      2022-08-12 [1] CRAN (R 4.2.0)
##  RSQLite                2.2.17     2022-09-10 [1] CRAN (R 4.2.0)
##  rstudioapi             0.13       2020-11-12 [1] CRAN (R 4.2.0)
##  rsvd                   1.0.5      2021-04-16 [1] CRAN (R 4.2.0)
##  RVenn                  1.1.0      2019-07-18 [1] CRAN (R 4.2.0)
##  rvest                  1.0.3      2022-08-19 [1] CRAN (R 4.2.0)
##  s2                     1.1.0      2022-07-18 [1] CRAN (R 4.2.0)
##  S4Vectors            * 0.34.0     2022-04-26 [1] Bioconductor
##  sandwich                    3.0-2      2022-06-15 [1] CRAN (R 4.2.0)
##  sass                        0.4.2      2022-07-16 [1] CRAN (R 4.2.0)
##  ScaledMatrix                1.4.0      2022-04-26 [1] Bioconductor
##  scales                      1.2.1      2022-08-20 [1] CRAN (R 4.2.0)
##  scater                      1.24.0     2022-04-26 [1] Bioconductor
##  scatterpie                  0.1.7      2022-09-02 [1] local
##  scuttle                     1.6.3      2022-08-23 [1] Bioconductor
##  sessioninfo                 1.2.2      2021-12-06 [1] CRAN (R 4.2.0)
##  sf                          1.0-8      2022-07-14 [1] CRAN (R 4.2.0)
##  shadowtext                * 0.1.2      2022-04-22 [1] CRAN (R 4.2.0)
##  shiny                       1.7.2      2022-07-19 [1] CRAN (R 4.2.0)
##  shinycustomloader           0.9.0      2018-03-27 [1] CRAN (R 4.2.0)
##  shinyWidgets                0.7.4      2022-10-05 [1] CRAN (R 4.2.0)
##  SingleCellExperiment      * 1.18.0     2022-04-26 [1] Bioconductor
##  sp                          1.5-0      2022-06-05 [1] CRAN (R 4.2.0)
##  sparseMatrixStats           1.8.0      2022-04-26 [1] Bioconductor
##  spatial                     7.3-15     2022-01-16 [1] CRAN (R 4.2.0)
##  stable                      1.1.6      2022-03-02 [1] CRAN (R 4.2.0)
##  stabledist                  0.7-1      2016-09-12 [1] CRAN (R 4.2.0)
##  statip                      0.2.3      2019-11-17 [1] CRAN (R 4.2.0)
##  statmod                     1.4.37     2022-08-12 [1] CRAN (R 4.2.0)
##  stringi                     1.7.8      2022-07-11 [1] CRAN (R 4.2.0)
##  stringr                     1.4.1      2022-08-20 [1] CRAN (R 4.2.0)
##  SummarizedExperiment      * 1.26.1     2022-04-29 [1] Bioconductor
##  survival                  * 3.3-1      2022-03-03 [1] CRAN (R 4.2.0)
##  svglite                     2.1.0      2022-02-03 [1] CRAN (R 4.2.0)
##  systemfonts                 1.0.4      2022-02-11 [1] CRAN (R 4.2.0)
##  textshaping                 0.3.6      2021-10-13 [1] CRAN (R 4.2.0)
##  TH.data                     1.1-1      2022-04-26 [1] CRAN (R 4.2.0)
##  tibble                      3.1.8      2022-07-22 [1] CRAN (R 4.2.0)
##  tidybulk                  * 1.8.2      2022-09-29 [1] Bioconductor
##  tidygraph                   1.2.2      2022-08-22 [1] CRAN (R 4.2.0)
##  tidyr                       1.2.1      2022-09-08 [1] CRAN (R 4.2.0)
##  tidyselect                  1.2.0      2022-10-10 [1] CRAN (R 4.2.0)
##  tidytree                    0.4.2      2022-12-18 [1] CRAN (R 4.2.0)
##  timeDate                    4021.104   2022-07-19 [1] CRAN (R 4.2.0)
##  timeSeries                  4021.104   2022-07-17 [1] CRAN (R 4.2.0)
##  treeio                      1.21.3     2022-10-30 [1] Bioconductor
##  TreeSummarizedExperiment  * 2.4.0      2022-04-26 [1] Bioconductor
##  truncnorm                   1.0-8      2018-02-27 [1] CRAN (R 4.2.0)
##  tweenr                      1.0.2      2021-03-23 [1] CRAN (R 4.2.0)
##  tzdb                        0.3.0      2022-03-28 [1] CRAN (R 4.2.0)
##  units                       0.8-0      2022-02-05 [1] CRAN (R 4.2.0)
##  usethis                     2.1.6      2022-05-25 [1] CRAN (R 4.2.0)
##  utf8                        1.2.2      2021-07-24 [1] CRAN (R 4.2.0)
##  vcd                         1.4-10     2022-06-09 [1] CRAN (R 4.2.0)
##  vctrs                       0.5.0      2022-10-22 [1] CRAN (R 4.2.0)
##  vegan                       2.6-2      2022-04-17 [1] CRAN (R 4.2.0)
##  VIM                         6.2.2      2022-08-25 [1] CRAN (R 4.2.0)
##  vipor                       0.4.5      2017-03-22 [1] CRAN (R 4.2.0)
##  viridis                     0.6.2      2021-10-13 [1] CRAN (R 4.2.0)
##  viridisLite                 0.4.1      2022-08-22 [1] CRAN (R 4.2.0)
##  webshot                     0.5.4      2022-09-26 [1] CRAN (R 4.2.0)
##  withr                       2.5.0      2022-03-03 [1] CRAN (R 4.2.0)
##  wk                          0.6.0      2022-01-03 [1] CRAN (R 4.2.0)
##  xfun                        0.32       2022-08-10 [1] CRAN (R 4.2.0)
##  xml2                        1.3.3      2021-11-30 [1] CRAN (R 4.2.0)
##  xtable                      1.8-4      2019-04-21 [1] CRAN (R 4.2.0)
##  XVector                   * 0.36.0     2022-04-26 [1] Bioconductor
##  yaml                        2.3.5      2022-02-21 [1] CRAN (R 4.2.0)
##  yulab.utils              0.0.5     2022-06-30 [1] CRAN (R 4.2.0)
##  zCompositions            1.4.0-1   2022-03-26 [1] CRAN (R 4.2.0)
##  zlibbioc                 1.42.0    2022-04-26 [1] Bioconductor
##  zoo                      1.8-10    2022-04-15 [1] CRAN (R 4.2.0)
##
##  [1] /mnt/d/UbuntuApps/R/4.2.0/lib/R/library
##
## ------------------------------------------------------------------------------------------------------
```

References

Anderson, Marti J, Kari E Ellingsen, and Brian H McArdle. 2006. "Multivariate Dispersion as a Measure of Beta Diversity." *Ecology Letters* 9 (6): 683—693. https://doi.org/10.1111/j.1461-0248.2006.00926.x.

Bjerrum, Jacob Tveiten, Casper Steenholdt, Mark Ainsworth, Ole Haagen Nielsen, Michelle AC Reed, Karen Atkins, Ulrich Leonhard GÃŒnther, Fuhua Hao, and Yulan Wang. 2017. "Metabonomics Uncovers a Reversible Proatherogenic Lipid Profile During Infliximab Therapy of Inflammatory Bowel Disease." *BMC Medicine* 15 (1): 184. https://doi.org/10.1186/s12916-017-0949-7.

Bolyen, Evan, Jai Ram Rideout, Matthew R. Dillon, Nicholas A. Bokulich, Christian C. Abnet, Gabriel A. Al-Ghalith, Harriet Alexander, et al. 2019. "Reproducible, Interactive, Scalable and Extensible Microbiome Data Science Using Qiime 2." *Nature Biotechnology* 37 (8): 852–57. https://doi.org/10.1038/s41587-019-0209-9.

Cadotte, Marc W., T. Jonathan Davies, James Regetz, Steven W. Kembel, Elsa Cleland, and Todd H. Oakley. 2010. "Phylogenetic Diversity Metrics for Ecological Communities: Integrating Species Richness, Abundance and Evolutionary History." *Ecology Letters* 13 (1): 96–105. https://doi.org/https://doi.org/10.1111/j.1461-0248.2009.01405.x.

Callahan, Benjamin J., Paul J. McMurdie, Michael J. Rosen, Andrew W. Han, Amy Jo A. Johnson, and Susan P. Holmes. 2016. "DADA2: High-Resolution Sample Inference from Illumina Amplicon Data." *Nature Methods* 13 (7): 581–83. https://doi.org/10.1038/nmeth.3869.

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 Crohn's Disease." *Microbiome* 6 (1): 13. https://doi.org/10.1186/s40168-018-0398-3.

Egozcue, J. J., V. Pawlowsky-Glahn, G. Mateu-Figueras, and C. BarcelÃ³-Vidal. 2003. "Isometric Logratio Transformations for Compositional Data Analysis." *Mathematical Geology* 35 (3): 279–300. https://doi.org/10.1023/A:1023818214614.

Heinken, Almut, Johannes Hertel, and Ines Thiele. 2021. "Metabolic Modelling Reveals Broad Changes in Gut Microbial Metabolism in Inflammatory Bowel Disease Patients with Dysbiosis." *Npj Systems Biology and Applications* 7 (1): 19. https://doi.org/10.1038/s41540-021-00178-6.

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.

Legendre, Pierre, and Eugene D. Gallagher. 2001. "Ecologically Meaningful Transformations for Ordination of Species Data." *Oecologia* 129 (2): 271–80. https://doi.org/10.1007/s004420100716.

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.

Morton, James T., Jon Sanders, Robert A. Quinn, Daniel McDonald, Antonio Gonzalez, Yoshiki Vázquez-Baeza, Jose A. Navas-Molina, et al. 2017. "Balance Trees Reveal Microbial Niche Differentiation." *mSystems* 2 (1): e00162–16. https://doi.org/10.1128/mSystems.00162-16.

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.

Pagès, H., P. Aboyoun, R. Gentleman, and S. DebRoy. 2021. *Biostrings: Efficient Manipulation of Biological Strings*. https://bioconductor.org/packages/Biostrings.

Polunin, German, Igor Sedakov, Oleksandr Borota, Olga Shatova, and Oleksandr Jr Borota. 2013. "The Influence of Sodium Lactate on Carbohydrates Metabolism in Patients with Inflammatory Bowel Diseases and Colon Cancer." *Annals of Oncology* 24 (June): iv116. https://doi.org/10.1093/annonc/mdt203.283.

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, Jacques Izard, Levi Waldron, Dirk Gevers, Larisa Miropolsky, Wendy S. Garrett, and Curtis Huttenhower. 2011. "Metagenomic Biomarker Discovery and Explanation." *Genome Biology* 12 (6): R60. https://doi.org/10.1186/gb-2011-12-6-r60.

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.

Truong, Duy Tin, Eric A. Franzosa, Timothy L. Tickle, Matthias Scholz, George Weingart, Edoardo Pasolli, Adrian Tett, Curtis Huttenhower, and Nicola Segata. 2015. "MetaPhlAn2 for Enhanced Metagenomic Taxonomic Profiling." *Nature Methods* 12 (10): 902–3. https://doi.org/10.1038/nmeth.3589.

Wang, Li-Gen, Tommy Tsan-Yuk Lam, Shuangbin Xu, Zehan Dai, Lang Zhou, Tingze Feng, Pingfan Guo, et al. 2020. "Treeio: An R Package for Phylogenetic Tree Input and Output with Richly Annotated and Associated Data." *Molecular Biology and Evolution* 37 (2): 599–603. https://doi.org/10.1093/molbev/msz240.

Webb, Campbell O. 2000. "Exploring the Phylogenetic Structure of Ecological Communities: An Example for Rain Forest Trees." *The American Naturalist* 156 (2): 145–55. https://doi.org/10.1086/303378.

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

Wirbel, Jakob, Paul Theodor Pyl, Ece Kartal, Konrad Zych, Alireza Kashani, Alessio Milanese, Jonas S. Fleck, et al. 2019. "Meta-Analysis of Fecal Metagenomes Reveals Global Microbial Signatures That Are Specific for Colorectal Cancer." *Nature Medicine* 25 (4): 679–89. https://doi.org/10.1038/s41591-019-0406-6.

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.

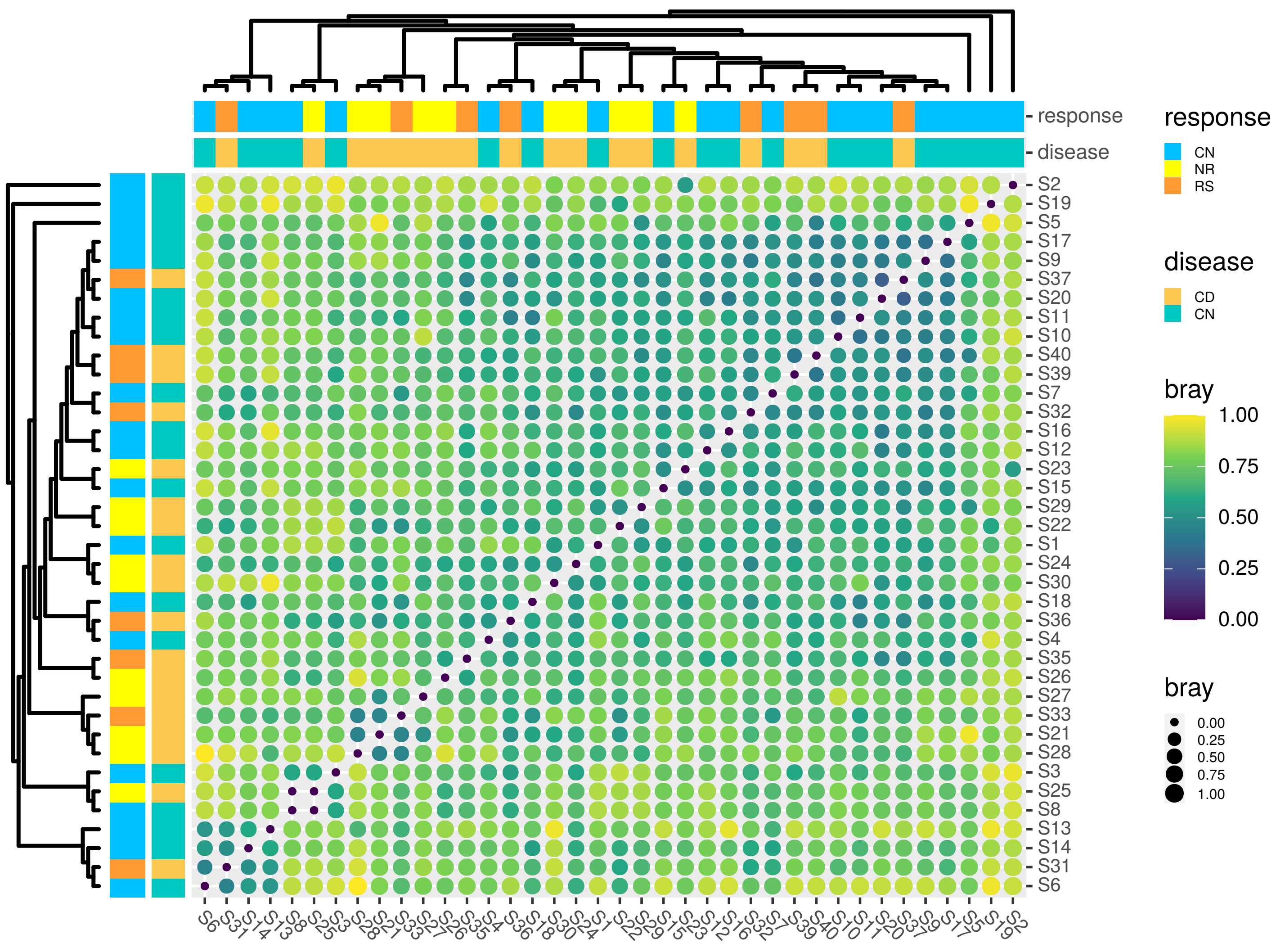
Yu, Guangchuang. 2021. *Tidytree: A Tidy Tool for Phylogenetic Tree Data Manipulation*. https://CRAN.R-project.org/package=tidytree.

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.

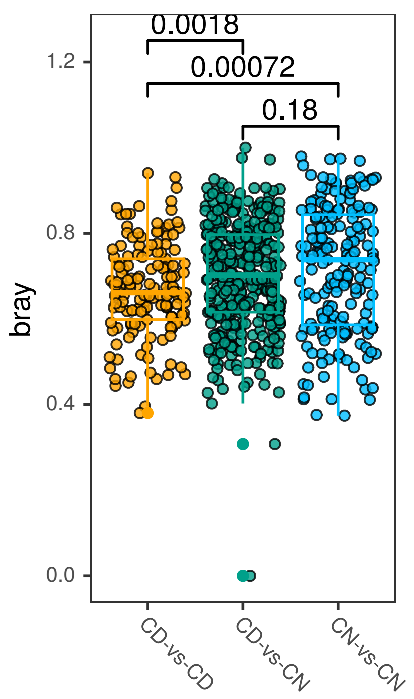
Zeller, Georg, Julien Tap, Anita Y Voigt, Shinichi Sunagawa, Jens Roat Kultima, Paul I Costea, Aurélien Amiot, et al. 2014. "Potential of Fecal Microbiota for Early-Stage Detection of Colorectal Cancer." *Molecular Systems Biology* 10 (11): 766.

https://doi.org/https://doi.org/10.15252/msb.20145645.

A



B



C

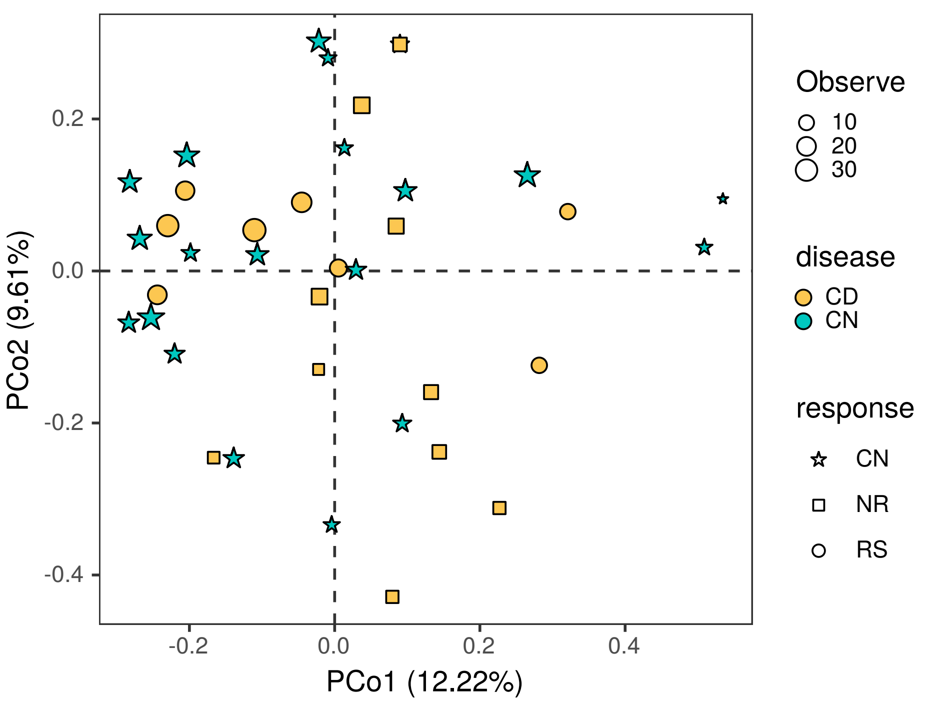


Fig. SA.27: **The distance heatmap and boxplot and the PCoA plot based on the MGS data**

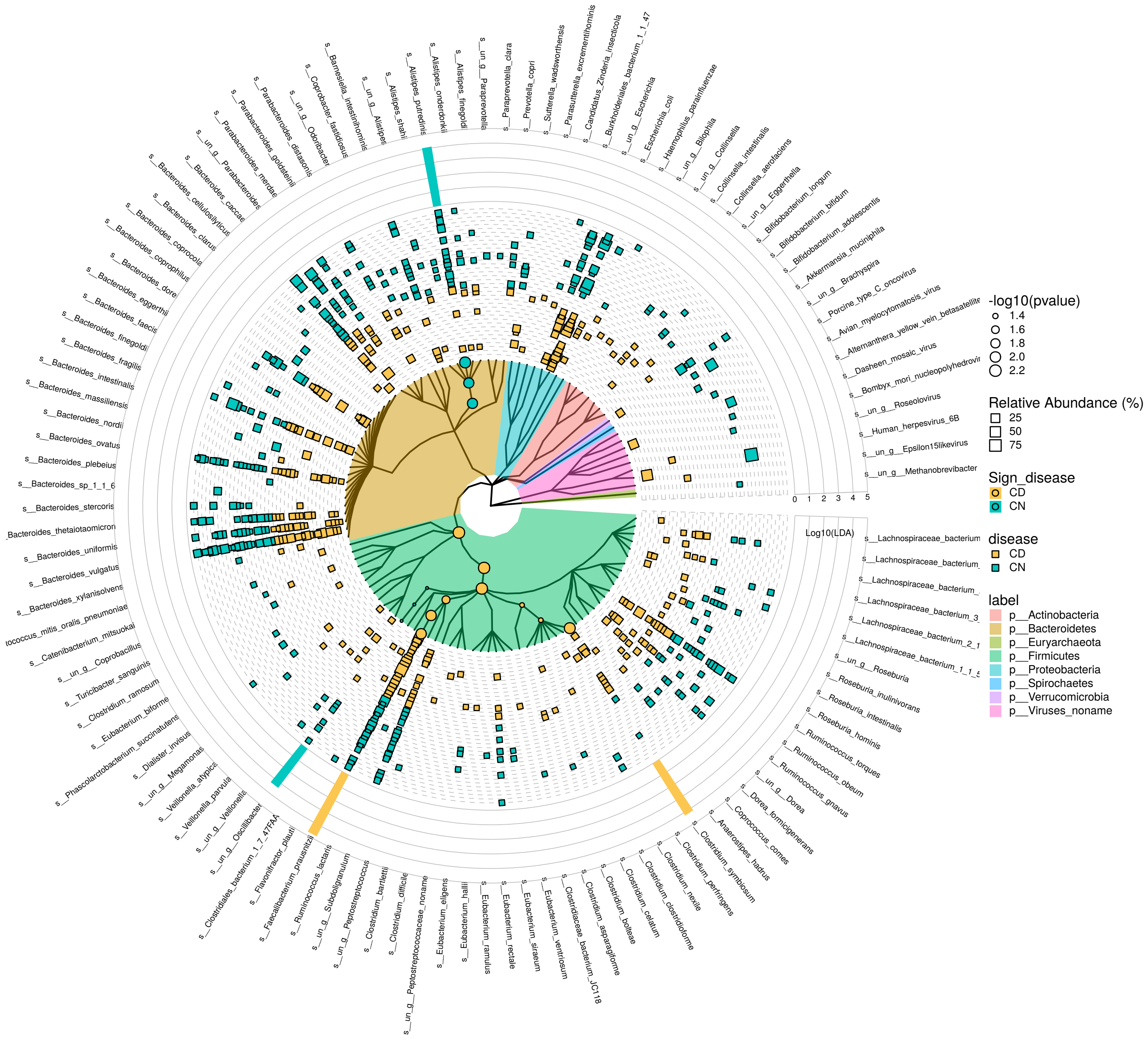


Fig. SA.28: **The result of differential analysis based on the MGS data**

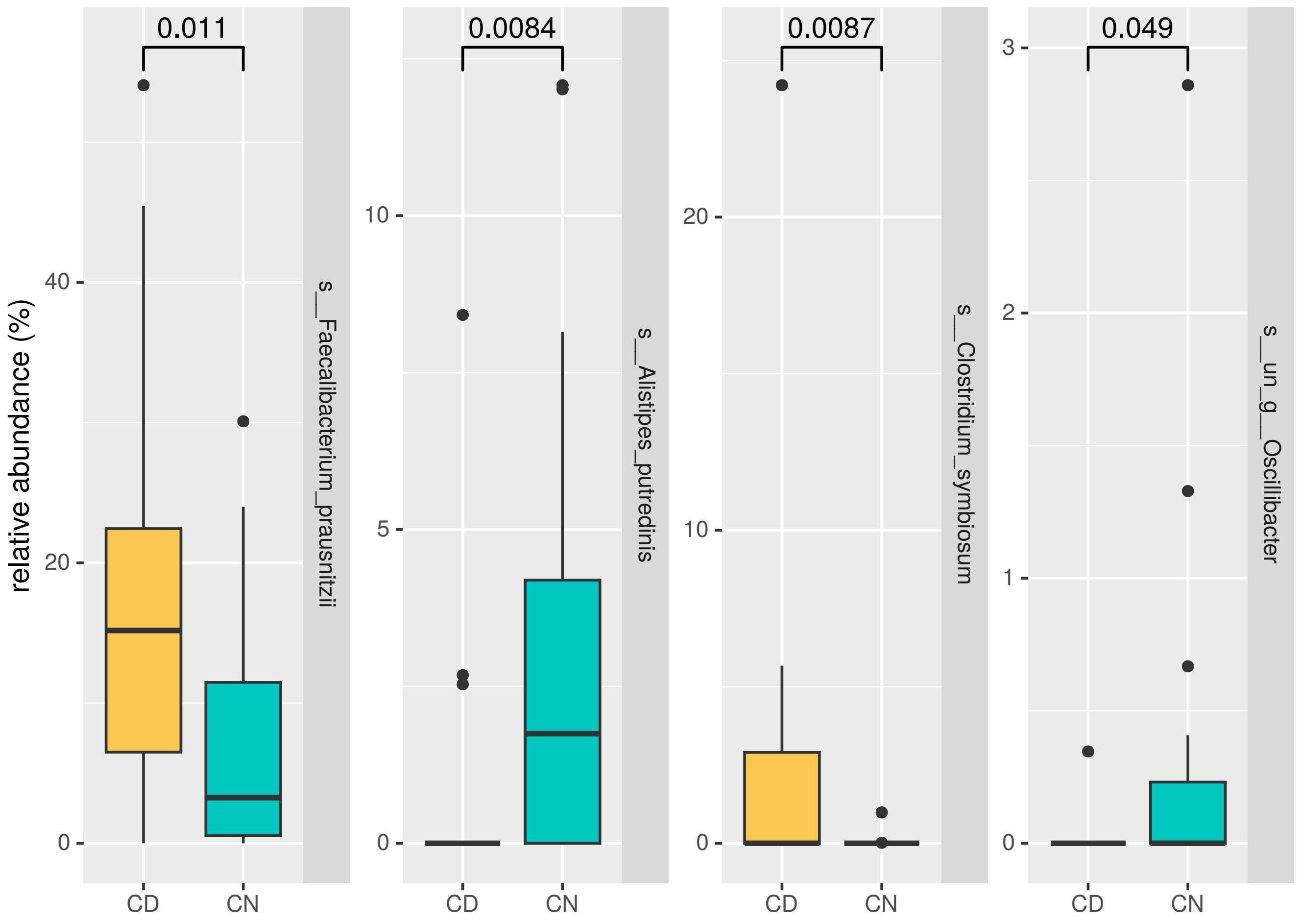


Fig. SA.29: **The abundance boxplot of the differential species between the CD and control group**

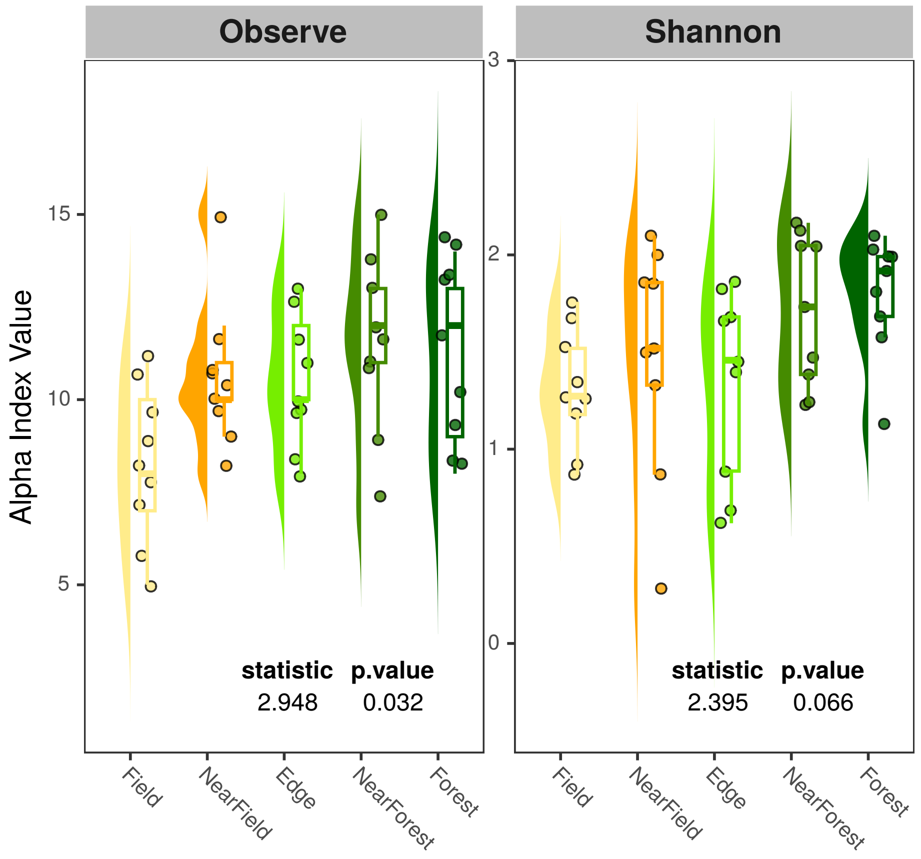


Fig. SA.30: **The raincloud plot of the alpha diversity of the Mosquito ecology community.** The result of the alpha diversity analysis about the Mosquito ecology study showed that the Mosquito species richness gradually increases from field to forest (field --> near field --> edge --> near field --> forest).

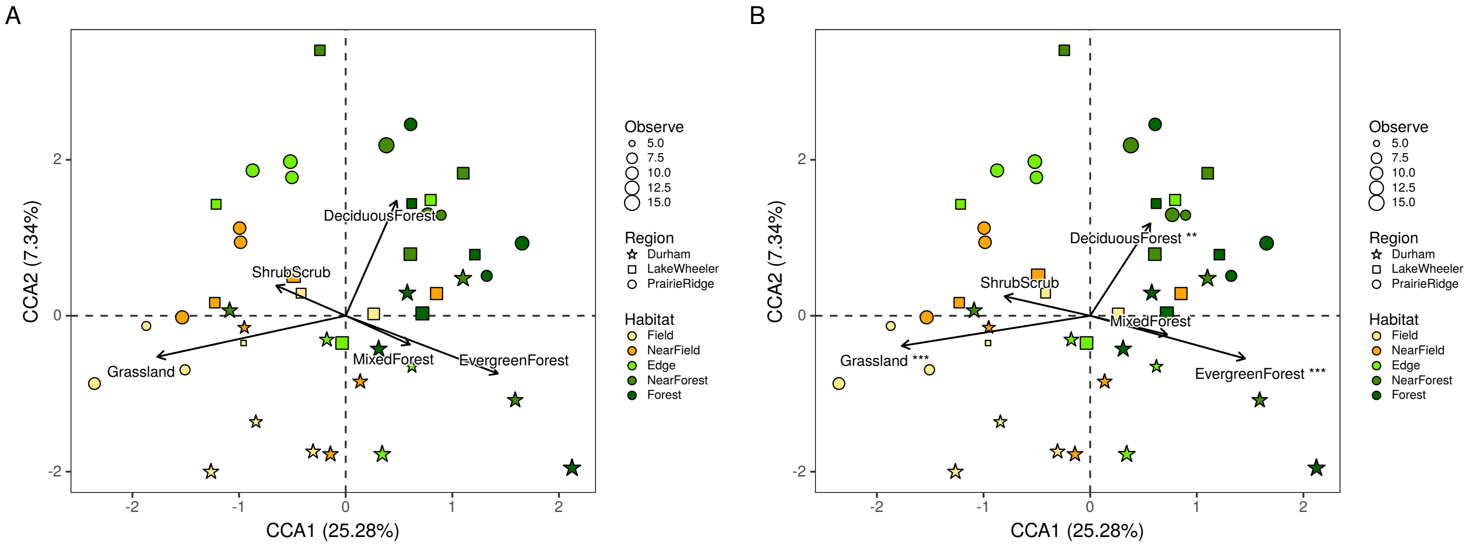


Fig. SA.31: **The CCA plot of the Mosquito ecology study (A) without the result of *mp_envfit* (B) with the result of *mp_envfit*.** Each point represents one sample, the size of the points represents the observe species of the corresponding sample, the color of the points represents the habitat of the 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).

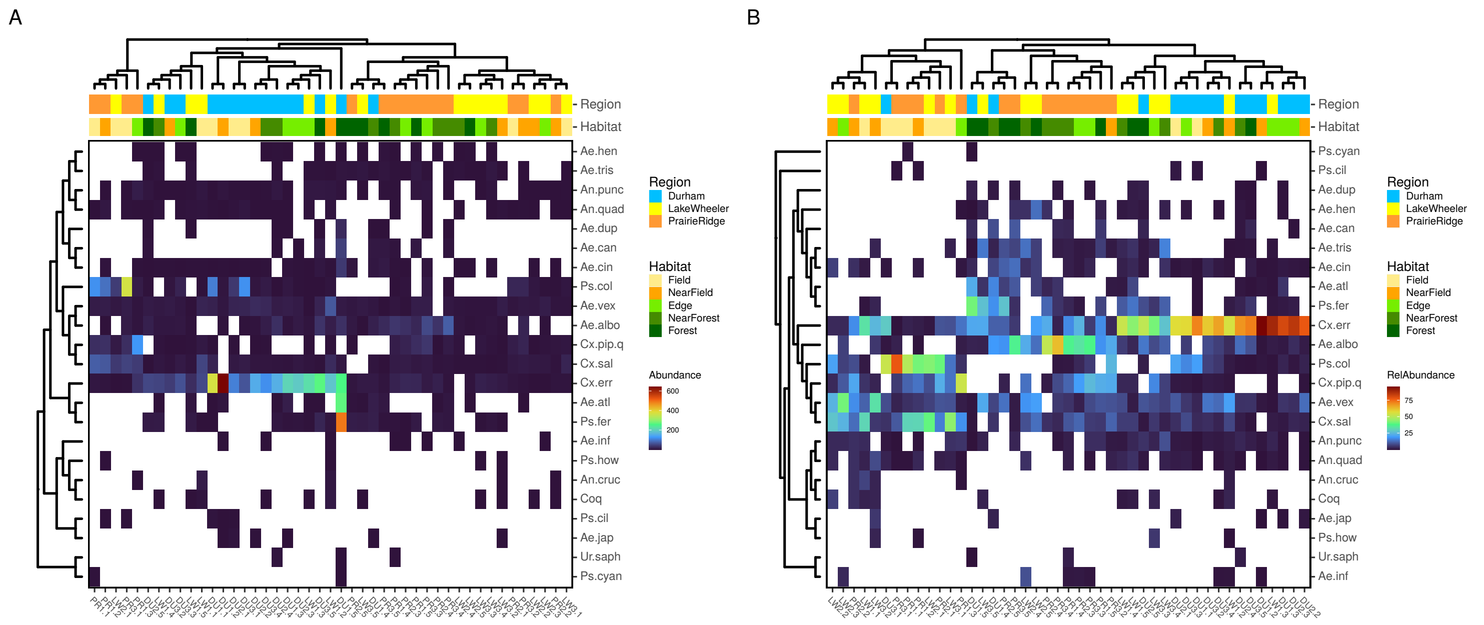


Fig. SA.32: **The heatmap of the abundance (A) and relative abundance (B) of the Mosquito species.**

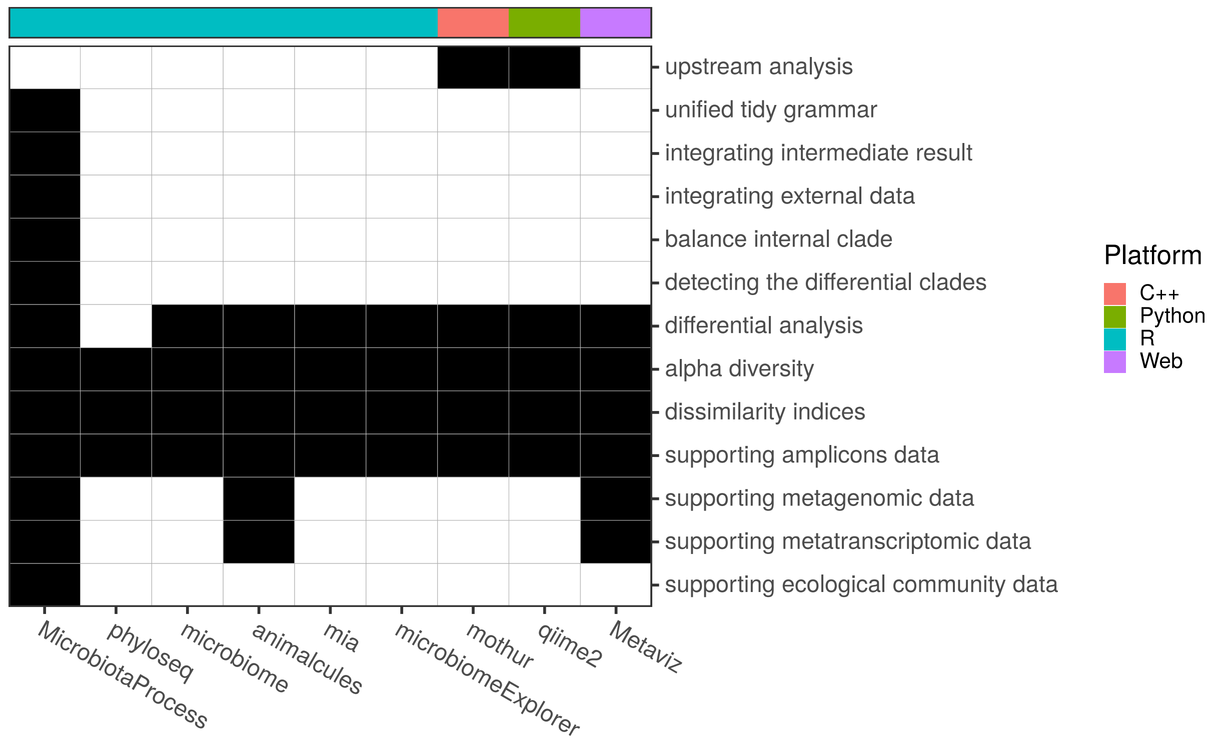


Fig. SA.33: **The comparison of features among the common tools developed for microbiome study**