MicrobiotaProcess Workshop

Ella Crotty

Contents

Links

- Workshop Link
- MicrobiotaProcess Documentation

Installs

MicrobiotaProcess is an R package for analysis, visualization and biomarker discovery of microbial datasets. It introduces MPSE class, this make it more interoperable with the existing computing ecosystem. Moreover, it introduces a tidy microbiome data structure paradigm and analysis grammar. It provides a wide variety of microbiome data analysis procedures under the unified and common framework (tidy-like framework).

Need to run `BiocManager::install("MicrobiotaProcess")` and `WGCNA`

```
suppressPackageStartupMessages({
  library(MicrobiotaProcess) # an R package for analysis, visualization and biomarker
  ↪  discovery of Microbiome.
  library(phyloseq) # Handling and analysis of high-throughput microbiome census data.
  library(tidyverse) # Easily Install and Load the 'Tidyverse'.
  library(VennDiagram)
  library(UpSetR)
  library(grid)
  library(ggtree)
  library(ggnewscale) # Multiple Fill and Colour Scales in 'ggplot2'.
  library(vegan) # Community Ecology Package.
  library(coin) # Conditional Inference Procedures in a Permutation Test Framework.
  library(reshape2) # Flexibly Reshape Data: A Reboot of the Reshape Package.
```

```
  library(WGCNA) # Needed for horrifying correlation plot
})
```

```
## Warning: package 'MicrobiotaProcess' was built under R version 4.3.3
## Warning: package 'ggplot2' was built under R version 4.3.1
## Warning: package 'tidyr' was built under R version 4.3.1
## Warning: package 'readr' was built under R version 4.3.1
## Warning: package 'dplyr' was built under R version 4.3.1
## Warning: package 'stringr' was built under R version 4.3.1
## Warning: package 'lubridate' was built under R version 4.3.1
## Warning: package 'ggtree' was built under R version 4.3.2
## Warning: package 'ggnewscale' was built under R version 4.3.1
## Warning: package 'vegan' was built under R version 4.3.3
## Warning: package 'lattice' was built under R version 4.3.1
## Warning: package 'coin' was built under R version 4.3.1
## Warning: package 'survival' was built under R version 4.3.1
## Warning: package 'WGCNA' was built under R version 4.3.1
## Warning: package 'fastcluster' was built under R version 4.3.1
```

This workshop uses the 43 pediatric IBD stool samples as example, obtained from the Integrative Human Microbiome Project Consortium (iHMP).

Data from `https://www.microbiomeanalyst.ca/MicrobiomeAnalyst/resources/data/ibd_data.zip`:

- `ibd_asv_table.txt` - feature table (row features X column samples)
- `ibd_meta.csv` - metadata file of samples
- `ibd_taxa.txt` - the taxonomic annotation of features

```
# Use import_dada2 of MicrobiotaProcess to import the datasets, and return a phyloseq
↪  object.

# MicrobiotaProcess also has other import functions with different input and output
↪  formats

otuda <- read.table("./ibd_data/IBD_data/ibd_asv_table.txt", header=T,
                    check.names=F,
                    comment.char="",
                    row.names=1, sep="\t")
# building the output format of removeBimeraDenovo of dada2
otuda <- data.frame(t(otuda),
                    check.names=F)
sampleda <- read.csv("./ibd_data/IBD_data/ibd_meta.csv",
                     row.names=1,
                     comment.char="")
taxda <- read.table("./ibd_data/IBD_data/ibd_taxa.txt",
                    header=T,
                    row.names=1,
                    check.names=F, comment.char="")
```

```r
# the feature names should be the same with rownames of taxda.
taxda <- taxda[match(colnames(otuda), rownames(taxda)),]
psraw <- import_dada2(seqtab=otuda,
                      taxatab=taxda,
                      sampleda=sampleda)
# view the reads depth of samples. In this example,
# we remove the sample contained less than 20914 otus.
# sort(rowSums(otu_table(psraw)))

# samples were removed if the reads number is too little.
psraw <- prune_samples(sample_sums(psraw)>=sort(rowSums(otu_table(psraw)))[3], psraw)
# then the samples were rarefied to 20914 reads.
set.seed(1024)
ps <- rarefy_even_depth(psraw)
```

```
## You set `rngseed` to FALSE. Make sure you've set & recorded
##  the random seed of your session for reproducibility.
## See `?set.seed`
```

```
## ...
```

```
## 4240TUs were removed because they are no longer
## present in any sample after random subsampling
```

```
## ...
```

```r
ps
```

```
## phyloseq-class experiment-level object
## otu_table()   OTU Table:         [ 1257 taxa and 43 samples ]
## sample_data() Sample Data:       [ 43 samples by 1 sample variables ]
## tax_table()   Taxonomy Table:    [ 1257 taxa by 7 taxonomic ranks ]
## refseq()      DNAStringSet:      [ 1257 reference sequences ]
```

Data Cleaning

Rarefaction is used to compensate for the effect of sample size on the number of units observed in a sample. `MicrobiotaProcess` can graph rarefaction curves. Rarefaction adjusts for different library sizes across samples to improve alpha diversity comparisons. It involves randomly discarding reads from larger samples until all of the samples are statistically comparable.

```r
# for reproducibly random number
set.seed(1024)
# Show rarefaction curves
# Had to replace "Group" with "Class" - the vignette says "Group" to divide into CD and
↪  Control, but "Group" doesn't exist and "Class" has CD and Control.

suppressWarnings(rareres <- get_rarecurve(obj=ps, chunks=400)) # This makes a bunch of
↪  warnings

p_rare <- ggrarecurve(obj=rareres,
                      indexNames=c("Observe","Chao1","ACE"),
                      ) +
          theme(legend.spacing.y=unit(0.01,"cm"),
                legend.text=element_text(size=4)) # Plot lines individually
```

```
## The color has been set automatically, you can reset it manually by adding scale_color_manual(values=
```

```
prare1 <- ggrarecurve(obj=rareres,
                      factorNames="Class",
                      indexNames=c("Observe", "Chao1", "ACE")
                      ) +
          scale_fill_manual(values=c("#00AED7", "#FD9347"))+
          scale_color_manual(values=c("#00AED7", "#FD9347"))+
          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"))
```

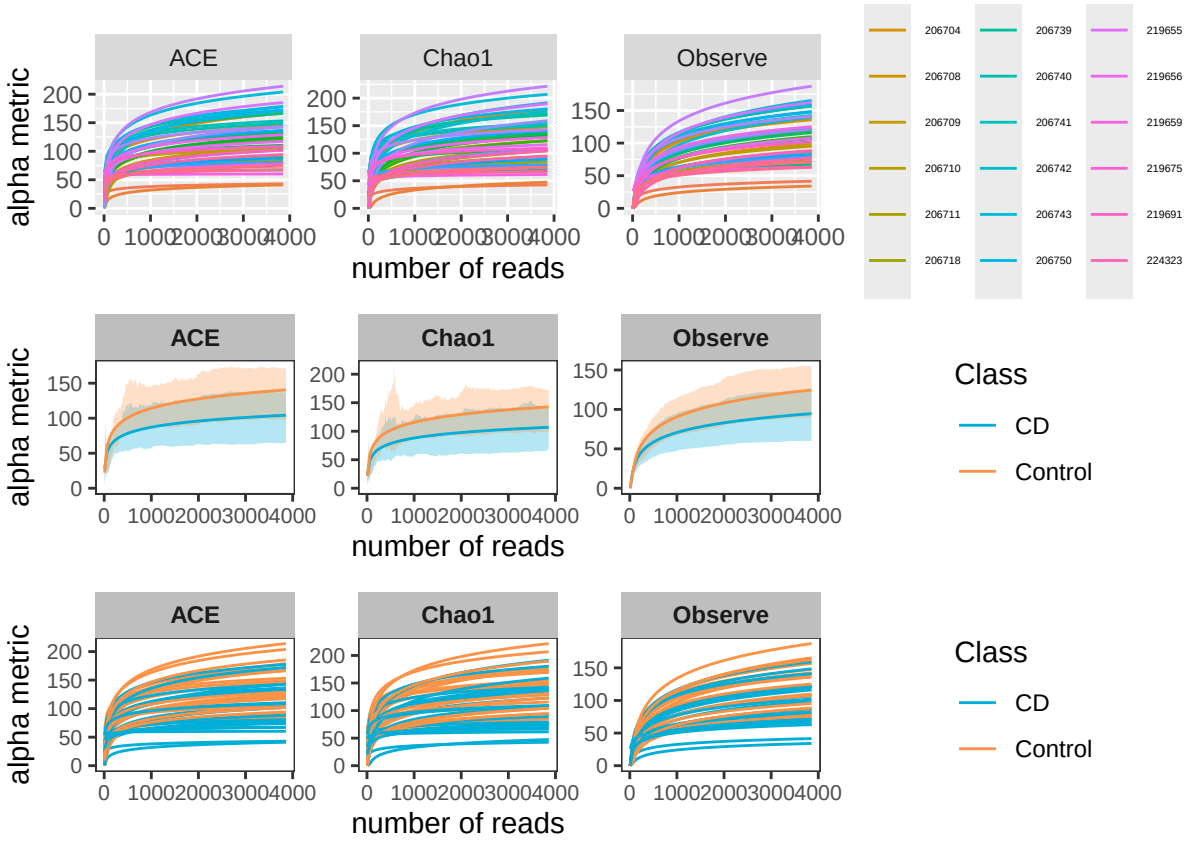
```
## The color has been set automatically, you can reset it manually by adding scale_color_manual(values=
```

```
prare2 <- ggrarecurve(obj=rareres,
                      factorNames="Class", # Plot lines by group
                      shadow=FALSE, # Plot every line instead of the average line
                      indexNames=c("Observe", "Chao1", "ACE")
                      ) +
          scale_color_manual(values=c("#00AED7", "#FD9347"))+
          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"))
```

```
## The color has been set automatically, you can reset it manually by adding scale_color_manual(values=
```

```
p_rare / prare1 / prare2
```

```
## Warning: The following aesthetics were dropped during statistical transformation: ymin
## and ymax.
## i This can happen when ggplot fails to infer the correct grouping structure in
##   the data.
## i Did you forget to specify a `group` aesthetic or to convert a numerical
##   variable into a factor?
## The following aesthetics were dropped during statistical transformation: ymin
## and ymax.
## i This can happen when ggplot fails to infer the correct grouping structure in
##   the data.
## i Did you forget to specify a `group` aesthetic or to convert a numerical
##   variable into a factor?
## The following aesthetics were dropped during statistical transformation: ymin
## and ymax.
## i This can happen when ggplot fails to infer the correct grouping structure in
##   the data.
## i Did you forget to specify a `group` aesthetic or to convert a numerical
##   variable into a factor?
```



These curves are near saturation - this means that there is enough data to detect species.

Alpha Diversity

```
alphaobj <- get_alphaindex(ps)
head(as.data.frame(alphaobj))
```

```
##        Observe     Chao1       ACE   Shannon   Simpson    Pielou   Class
## 206700      72  79.33333  80.61767 2.8500870 0.8910011 0.6664273      CD
## 206701      40  40.75000  40.90327 1.6786488 0.5756204 0.4550565      CD
## 206702      34  79.00000  44.47366 0.5485046 0.1628795 0.1555441      CD
## 206703     138 155.64706 158.44110 3.4903434 0.9081966 0.7083750 Control
## 206704     131 139.27273 137.89740 3.9340289 0.9641911 0.8069476 Control
## 206708      97 114.10000 110.69620 2.6931154 0.8369220 0.5886963      CD
```

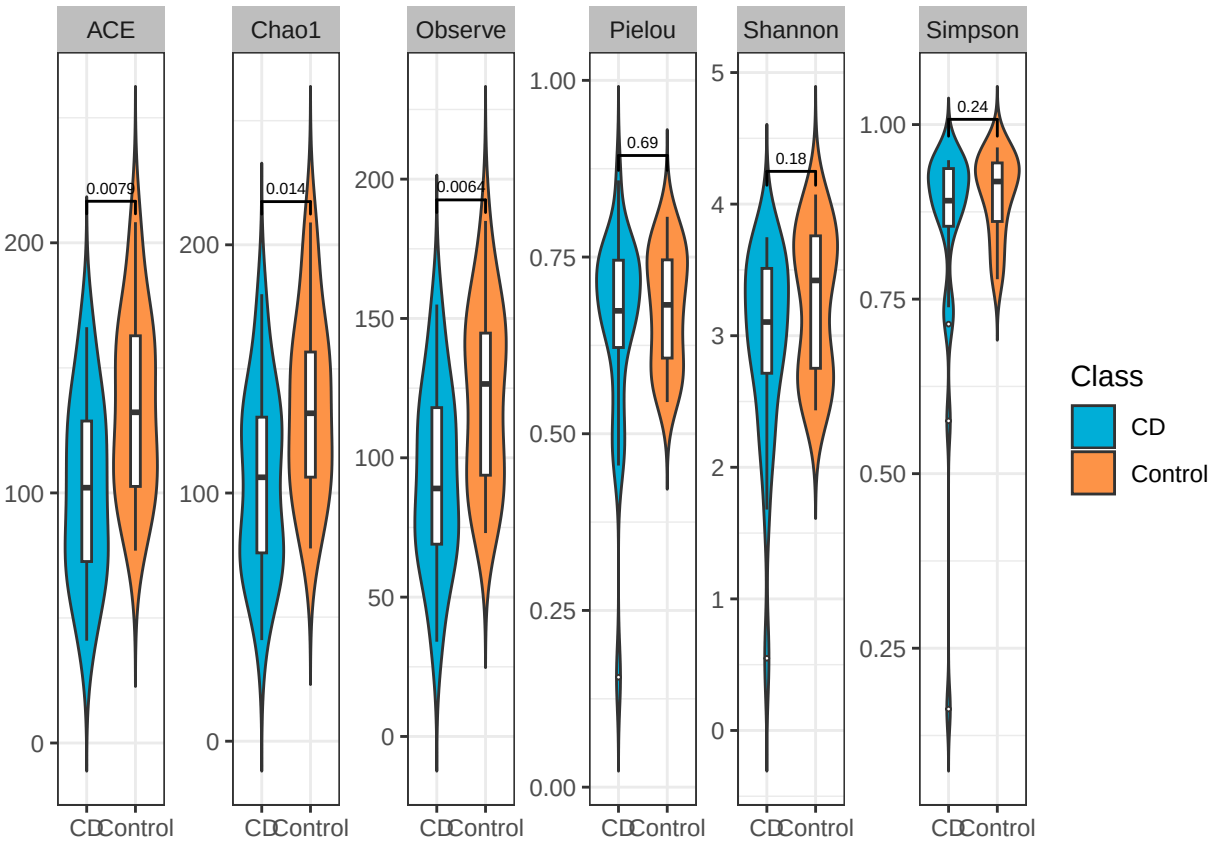
```
# This will make a dataframe of 6 common diversity measures
```

```
p_alpha <- ggbox(alphaobj, geom="violin", factorNames="Class") +
           scale_fill_manual(values=c("#00AED7", "#FD9347"))+
           theme(strip.background = element_rect(colour=NA, fill="grey"))
```

```
## The color has been set automatically, you can reset it manually by adding scale_fill_manual(values=y
```

```
p_alpha
```

```
## Warning in wilcox.test.default(c(72, 40, 34, 97, 88, 119, 117, 104, 114, :
## cannot compute exact p-value with ties
```



Taxonomic Plots

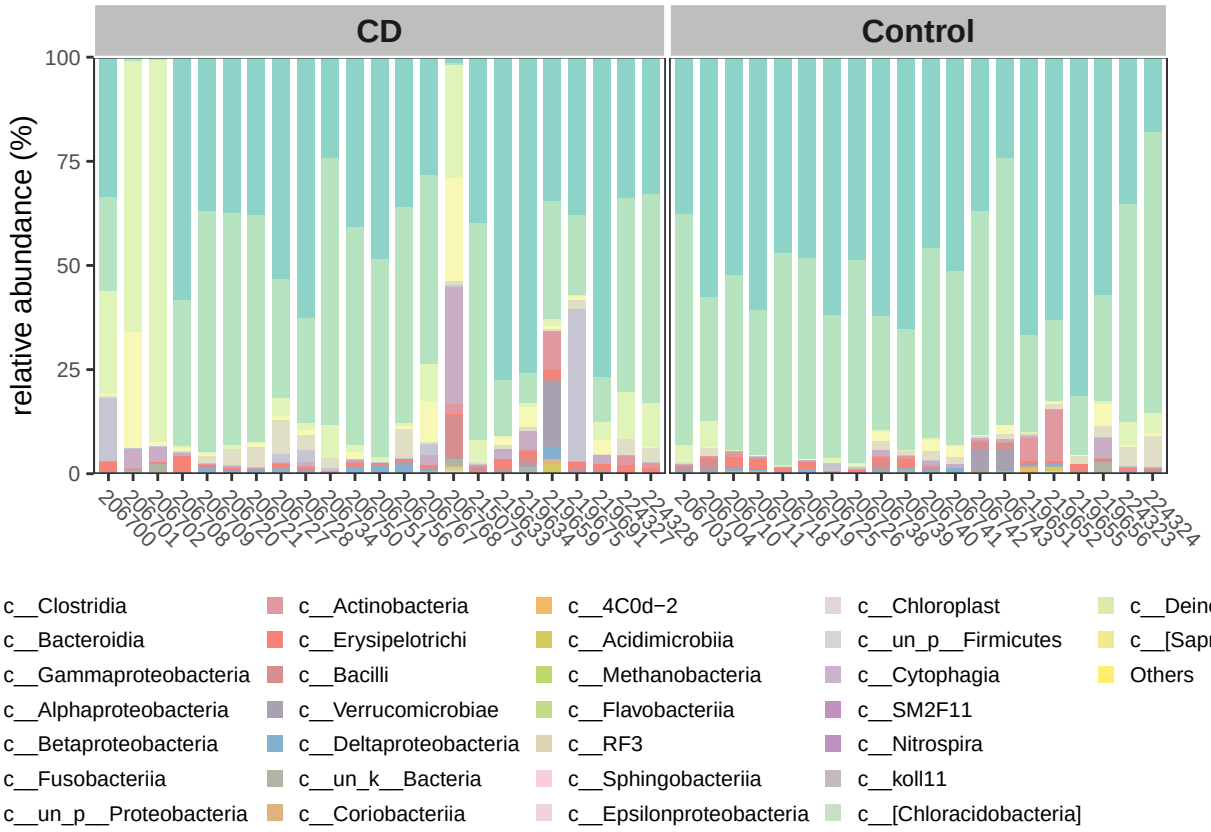
- ggbartax() can visualize community compositions
  - Use get_taxadf() first to visualize abundance of specific levels of class

```
classtaxa <- get_taxadf(obj=ps, taxlevel=3) # So that we can specify the taxonomic level

# The 30 most abundant taxonomy will be visualized by default (parameter `topn=30`).
pclass <- ggbartax(obj=classtaxa, facetNames="Class") +
          xlab(NULL) +
          ylab("relative abundance (%)") +

          ↪  scale_fill_manual(values=c(colorRampPalette(RColorBrewer::brewer.pal(12,"Set3"))(31)))
          ↪  +
          guides(fill= guide_legend(keywidth = 0.5, keyheight = 0.5))
```

```
## The color has been set automatically, you can reset it
##             manually by adding scale_fill_manual(values=yourcolors)
```

```
## Scale for fill is already present.
## Adding another scale for fill, which will replace the existing scale.
```

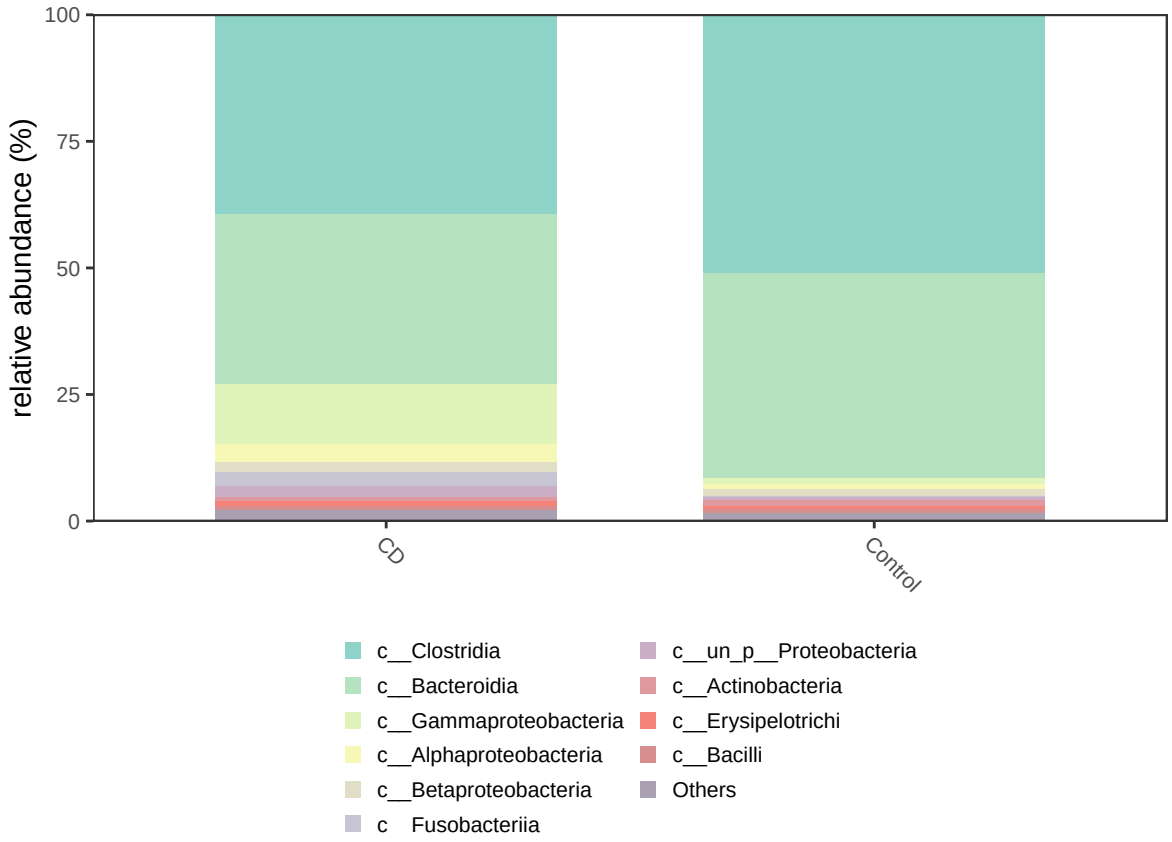
```
pclass
```




```
# Show the abundance in different groups.
fclass <- ggbartax(obj=classtaxa, facetNames="Class", plotgroup=TRUE, topn=10) + # Plot
↪  by group instead of by sample
          xlab(NULL) +
          ylab("relative abundance (%)") +

          ↪  scale_fill_manual(values=c(colorRampPalette(RColorBrewer::brewer.pal(12,"Set3"))(31)))
          ↪  +
          guides(fill= guide_legend(keywidth = 0.5, keyheight = 0.5, ncol=2))
```

```
## The color has been set automatically, you can reset it
##             manually by adding scale_fill_manual(values=yourcolors)
## Scale for fill is already present.
## Adding another scale for fill, which will replace the existing scale.
```

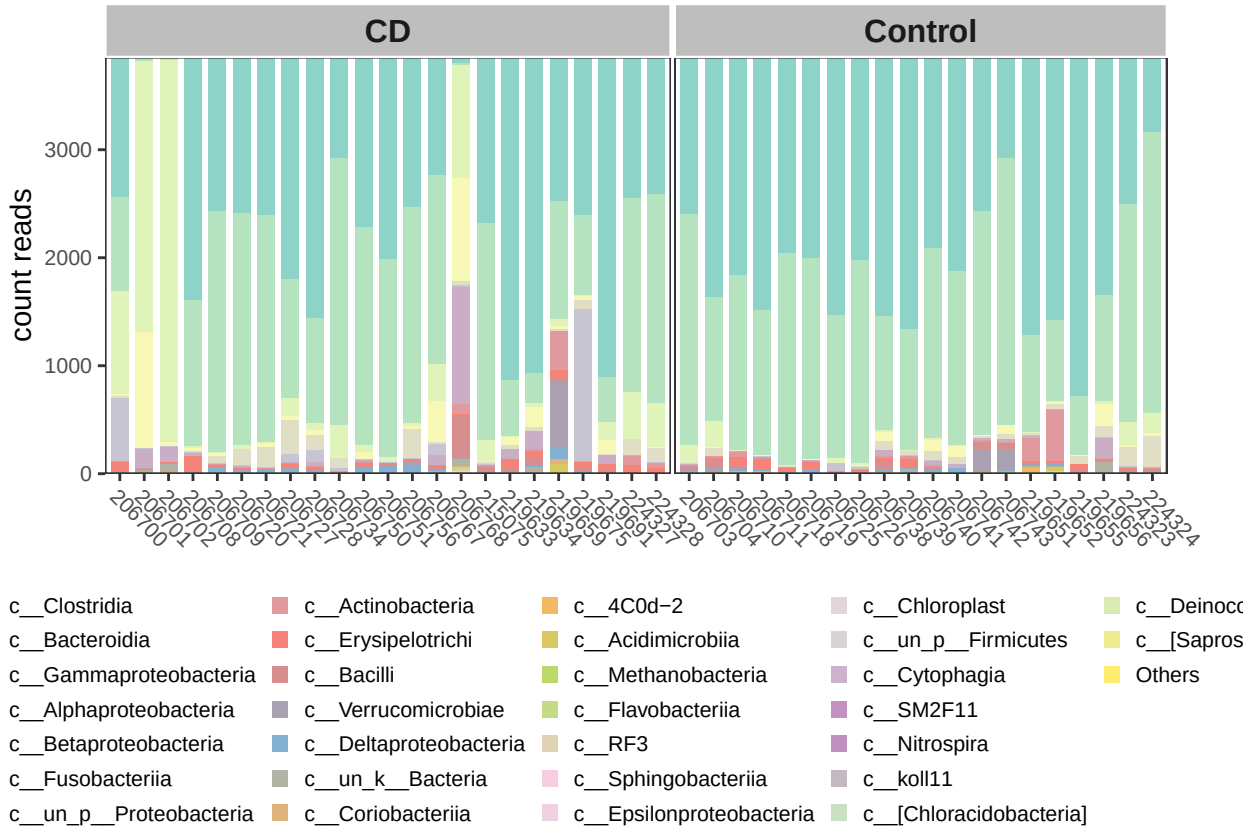
```
fclass
```




```
# View count instead of percent abundance
pclass2 <- ggbartax(obj=classtaxa, count=TRUE, facetNames="Class") +
          xlab(NULL) +
          ylab("count reads") +

          ↪  scale_fill_manual(values=c(colorRampPalette(RColorBrewer::brewer.pal(12,"Set3"))(31)))
          ↪  +
          guides(fill= guide_legend(keywidth = 0.5, keyheight = 0.5))
```

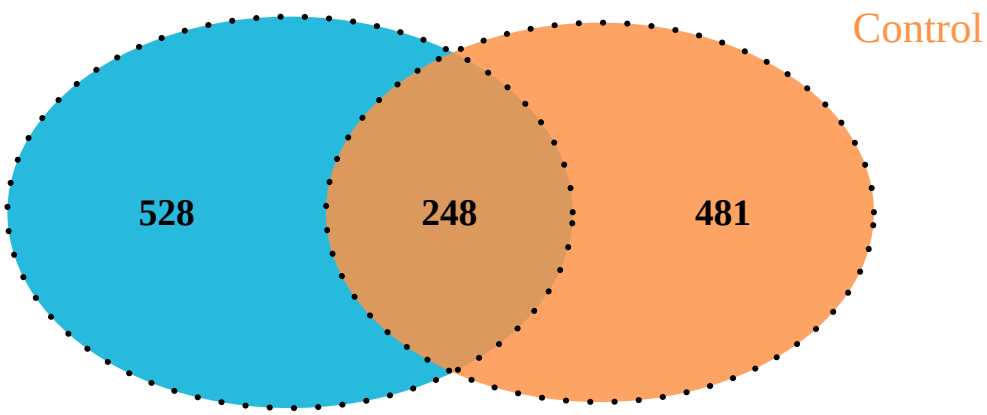
```
## The color has been set automatically, you can reset it
##             manually by adding scale_fill_manual(values=yourcolors)
## Scale for fill is already present.
## Adding another scale for fill, which will replace the existing scale.
```

```
pclass2
```

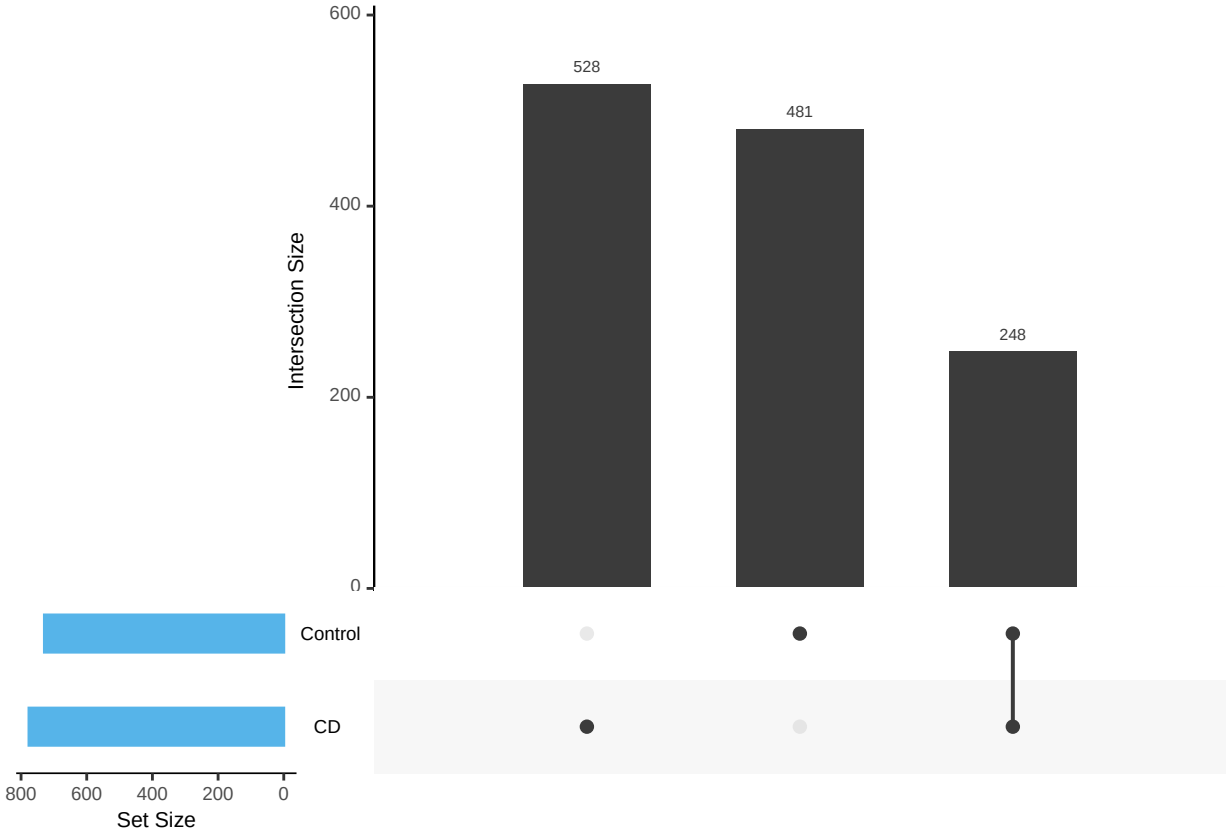


Venn plots can show the differences between groups. MicrobiotaProcess can use VennDiagram or UpSet.

```
vennlist <- get_vennlist(obj=ps, factorNames="Class")
upsetda <- get_upset(obj=ps, factorNames="Class")
vennp <- venn.diagram(vennlist,
                      height=5,
                      width=5,
                      filename=NULL,
                      fill=c("#00AED7", "#FD9347"),
                      cat.col=c("#00AED7", "#FD9347"),
                      alpha = 0.85,
                      fontfamily = "serif",
                      fontface = "bold",
                      cex = 1.2,
                      cat.cex = 1.3,
                      cat.default.pos = "outer",
                      cat.dist=0.1,
                      margin = 0.1,
                      lwd = 3,
                      lty ='dotted',
                      imagetype = "svg")
grid::grid.draw(vennp)
```




```
# Note: This has different numbers than the example on the website

upset(upsetda, sets=unique(as.vector(sample_data(ps)$Group)),
      sets.bar.color = "#56B4E9",
      order.by = "freq",
      empty.intersections = "on")
```



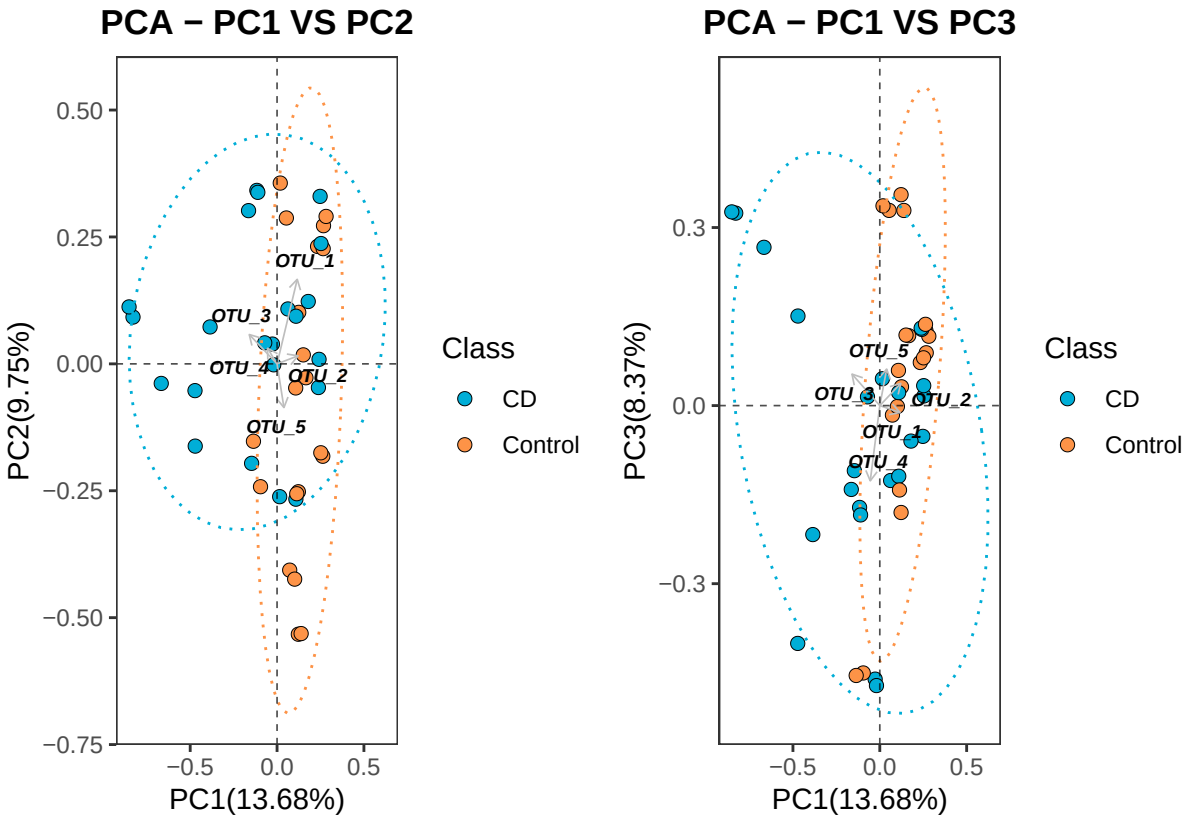
Beta Diversity

- `beta diversity` metrics can assess differences between communities
- Principal component analysis (PCA) = statistical procedure to compare groups
- Principle Coordinate Analysis (PCoA) = statistical procedure to compare groups of samples, which can be based on phylogenetic or count-based distance measures (ex. Bray-Curtis dissimilarity)

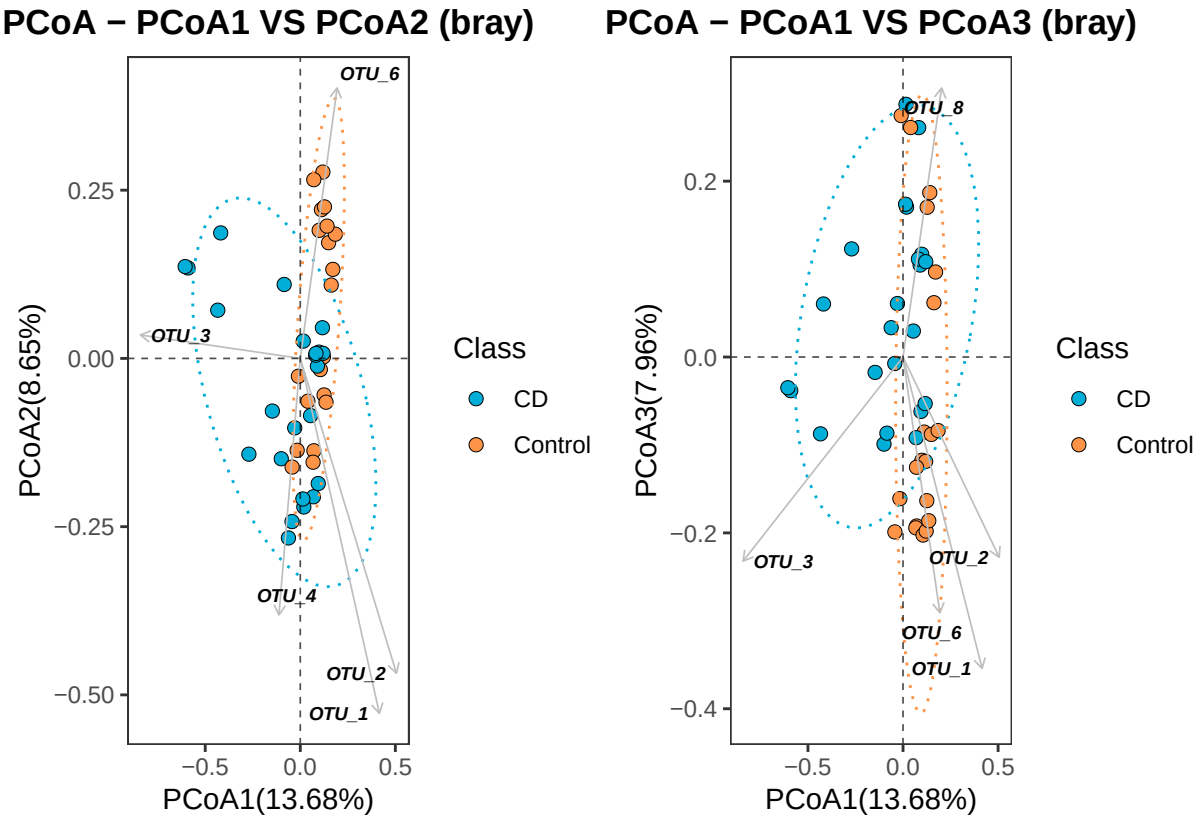
Notes on numbers of PCA

- PC1 is the component that represents the most variation, PC2 represents the second most
  - PC1 is strongly correlated with some group of variables that vary together
- Three samples = PC3 can exist, which is why these plots have PC3 sometimes
- The axes say how much variance is explained by each principal component

```
# If the input was normalized, the method parameter should be setted NULL.
# Get the PCA data
pcares <- get_pca(obj=ps, method="hellinger")

# Visulizing the result
pcaplot1 <- ggordpoint(obj=pcares,
                       biplot=TRUE,
                       speciesannot=TRUE,
                       factorNames=c("Class"),
                       ellipse=TRUE) +
            scale_color_manual(values=c("#00AED7", "#FD9347")) +
            scale_fill_manual(values=c("#00AED7", "#FD9347"))

# pc = c(1, 3) to show the first and third principal components.
pcaplot2 <- ggordpoint(obj=pcares,
                       pc=c(1, 3), # Change pc and biplot
                       biplot=TRUE,
                       speciesannot=TRUE,
                       factorNames=c("Class"),
                       ellipse=TRUE) +
            scale_color_manual(values=c("#00AED7", "#FD9347")) +
            scale_fill_manual(values=c("#00AED7", "#FD9347"))

pcaplot1 | pcaplot2
```




```
# distmethod options: "unifrac",  "wunifrac", "manhattan", "euclidean", "canberra",
↪  "bray", "kulczynski" ...(vegdist, dist)

# Make PCoA data
pcoares <- get_pcoa(obj=ps, distmethod="bray", method="hellinger")

# Visualizing the result
pcoaplot1 <- ggordpoint(obj=pcoares,
                        biplot=TRUE,
                        speciesannot=TRUE,
                        factorNames=c("Class"),
                        ellipse=TRUE) +
  scale_color_manual(values=c("#00AED7", "#FD9347")) +
  scale_fill_manual(values=c("#00AED7", "#FD9347"))

# first and third principal co-ordinates
pcoaplot2 <- ggordpoint(obj=pcoares, pc=c(1, 3), biplot=TRUE, speciesannot=TRUE,
                        factorNames=c("Class"), ellipse=TRUE) +
             scale_color_manual(values=c("#00AED7", "#FD9347")) +
             scale_fill_manual(values=c("#00AED7", "#FD9347"))

pcoaplot1 | pcoaplot2
```



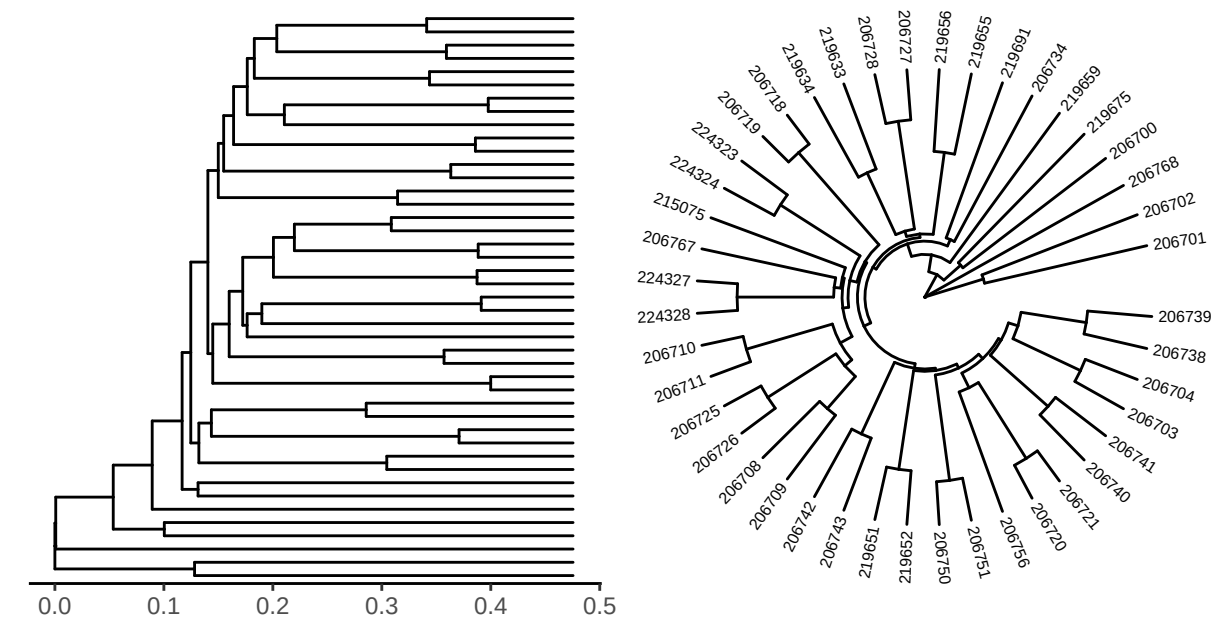

```
# Using the vegan package
distme <- get_dist(ps, distmethod ="bray", method="hellinger")
sampleda <- data.frame(sample_data(ps), check.names=FALSE)
sampleda <- sampleda[match(colnames(as.matrix(distme)),rownames(sampleda)),,drop=FALSE]
sampleda$Class <- factor(sampleda$Class)

set.seed(1024)
adores <- adonis2(distme ~ Class, # Switched adonis to adonis2 because vegan gave me that
↪  warning
                  data=sampleda,
                  permutation=9999)
data.frame(adores$aov.tab)
```

```
## data frame with 0 columns and 0 rows
```

```
# Well that didn't work. Unclear why.

# Intended output:
#>           Df  SumsOfSqs   MeanSqs  F.Model         R2 Pr..F.
#> Group      1  0.7645291 0.7645291 3.199178 0.07581139  1e-04
#> Residuals 39  9.3200922 0.2389767       NA 0.92418861     NA
#> Total     40 10.0846214        NA       NA 1.00000000     NA
```

```
# Hierarchical clustering can also visualize beta diversity using ggtree
# This gives a phylogenetic tree

# Make the hierarchical clustering object
hcsample <- get_clust(obj=ps,
```

```
                      distmethod="bray",
                      method="hellinger",
                      hclustmethod="average")

# Visualize with rectangular layout
cplot1 <- ggclust(obj=hcsample,
                  layout = "rectangular",
                  pointsize=1,
                  fontsize=0,
                  factorNames=c("Class")) +
              scale_color_manual(values=c("#00AED7",
                                          "#FD9347")) +
              theme_tree2(legend.position="right",
              plot.title = element_text(face="bold",
                                        lineheight=25,
                                        hjust=0.5))
# Visualize with circular layout
cplot2 <- ggclust(obj=hcsample,
                  layout = "circular",
                  pointsize=1,
                  fontsize=2,
                  factorNames=c("Class"),
                  factorLevels = list("CD", "Control")) + # Added to try to fix the
                   ↪  colors
          scale_color_manual(values=c("#00AED7",
                                      "#FD9347")) +
          theme(legend.position="right")

cplot1 | cplot2
```



```
# These are supposed to have CD and Control color dots at the end but they don't :(
```

Biomarker Detection

```
# Since the effect size was calculated by randomly re-sampling,
# the seed should be set for reproducibly results.
set.seed(1024)
deres <- diff_analysis(obj = ps, classgroup = "Class",
                       mlfun = "lda",
                       filtermod = "pvalue",
                       firstcomfun = "kruskal_test",
                       firstalpha = 0.05,
                       strictmod = TRUE,
                       secondcomfun = "wilcox_test",
                       subclmin = 3,
                       subclwilc = TRUE,
                       secondalpha = 0.01,
                       lda=3)
deres
```

```
## The original data: 634 features and 43 samples
## The sample data: 1 variables and 43 samples
## The taxda contained 1257 by 7 rank
## after first test (kruskal_test) number of feature (pvalue<=0.05):70
## after second test (wilcox_test and generalizedFC) number of significantly discriminative feature:25
## after lda, Number of discriminative features: 24 (certain taxonomy classification:15; uncertain taxo
```

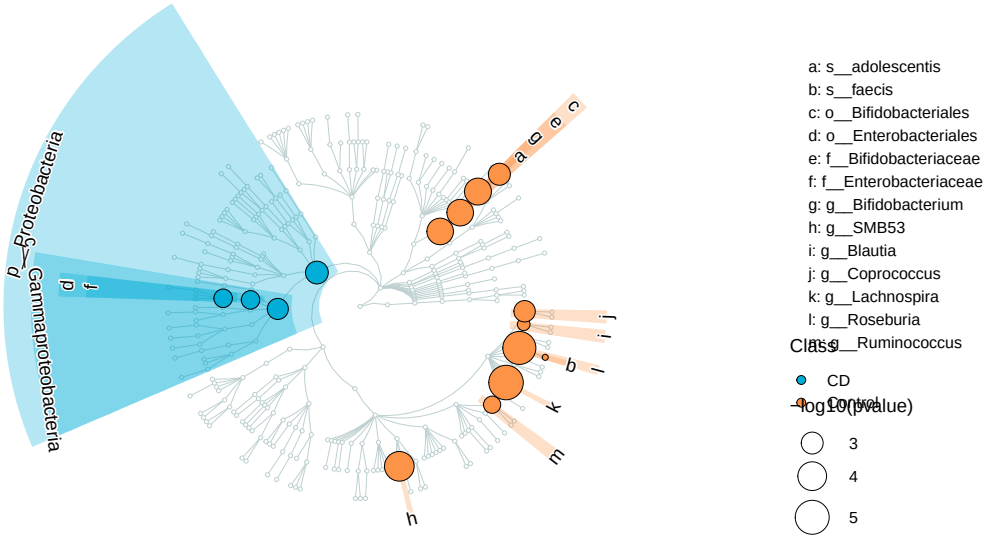
```
#> The original data: 689 features and 41 samples
#> The sample data: 1 variables and 41 samples
#> The taxda contained 1432 by 7 rank
#> after first test (kruskal_test) number of feature (pvalue<=0.05):71
#> after second test (wilcox_test) number of significantly discriminative feature:28
#> after lda, Number of discriminative features: 22 (certain taxonomy classification:15;
↪  uncertain taxonomy classication: 7)

# Visualize

diffclade_p <- ggdiffclade(
                    obj=deres,
                    alpha=0.3,
                    linewd=0.15,
                    skpointsize=0.6,
                    layout="radial",
                    taxlevel=3,
                    removeUnkown=TRUE,
                    reduce=TRUE # This argument is to remove the branch of unknown
                    ↪  taxonomy.
                ) +
                scale_fill_manual(
                    values=c("#00AED7", "#FD9347")
                ) +
                guides(color = guide_legend(
```

```
                                keywidth = 0.1,
                                keyheight = 0.6,
                                order = 3,
                                ncol=1)
            ) +
            theme(
                panel.background=element_rect(fill=NA),
                legend.position="right",
                plot.margin=margin(0,0,0,0),
                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")
            )
```

```
## The `removeUnkown` has been deprecated, Please use `removeUnknown` instead!
```

```
## Scale for fill is already present.
## Adding another scale for fill, which will replace the existing scale.
```

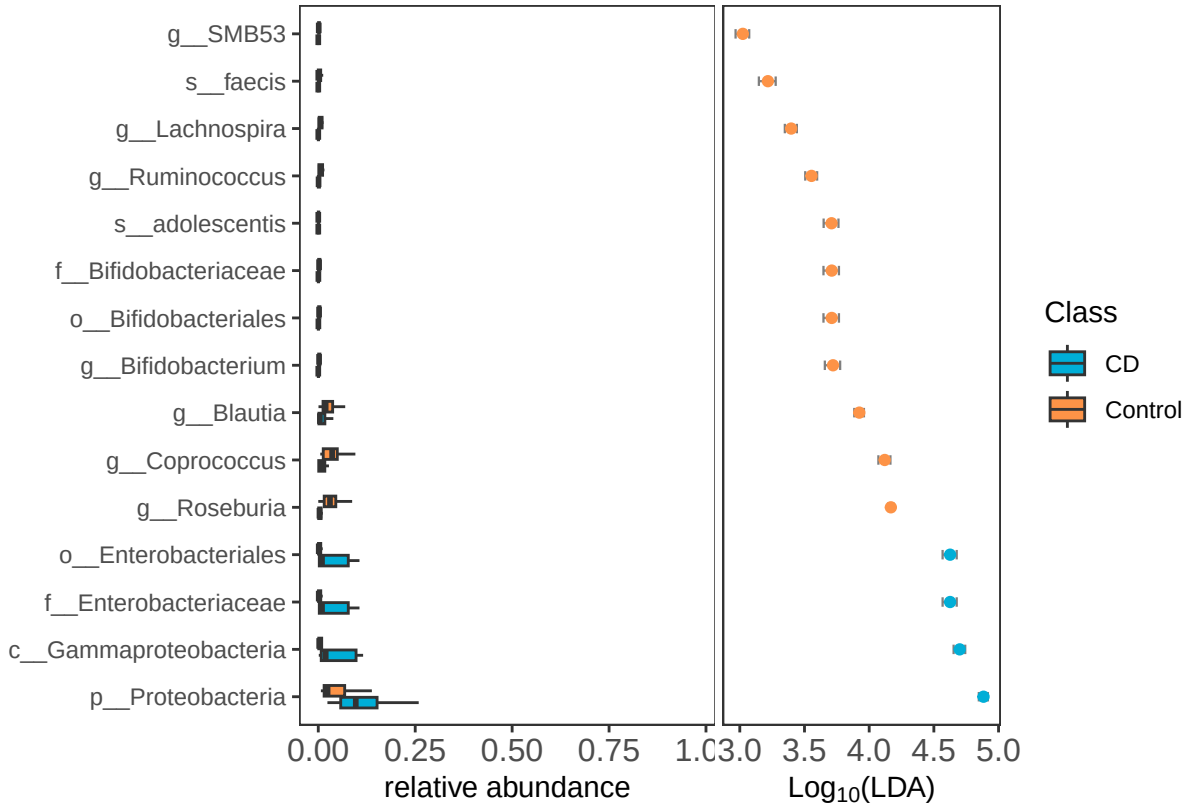
```
diffclade_p
```




```
# Looks pretty much like the example

# Visualize relative abundance and effect size

diffbox <- ggdiffbox(obj=deres, box_notch=FALSE,
            colorlist=c("#00AED7", "#FD9347"), l_xlabtext="relative abundance")
```

```
## The color has been set automatically, you can reset it manually by adding scale_color_manual(values=
## Scale for colour is already present.
## Adding another scale for colour, which will replace the existing scale.
```

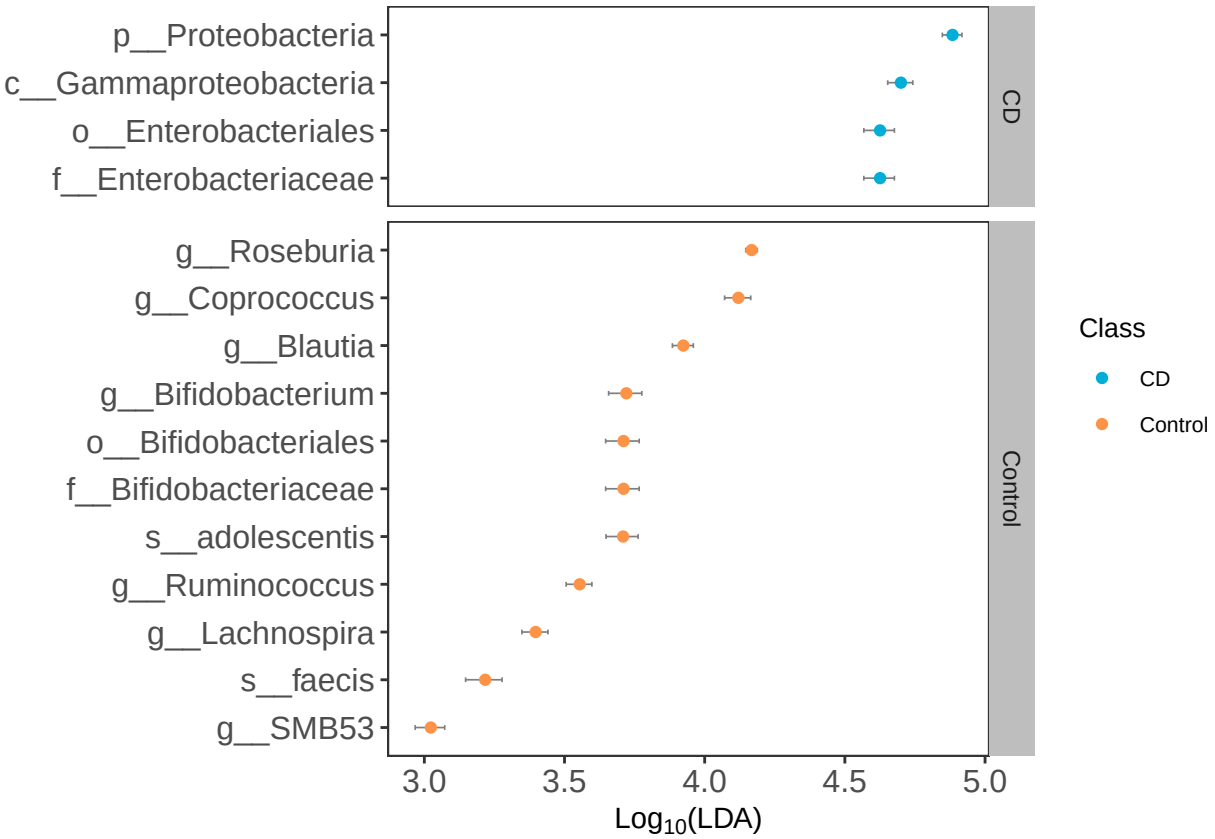
```
diffbox
```




```
# Now with taxonomic groups
ggdifftaxbar(obj=deres, xtextsize=1.5,
             output="IBD_biomarkder_barplot",
             coloslist=c("#00AED7", "#FD9347"))

# Now with confidence intervals
es_p <- ggeffectsize(obj=deres,
                     lineheight=0.1,
                     linewidth=0.3) +
        scale_color_manual(values=c("#00AED7",
                                    "#FD9347"))
```

```
## The color has been set automatically, you can reset it manually by adding scale_color_manual(values=
## Scale for colour is already present.
## Adding another scale for colour, which will replace the existing scale.
```

```
es_p
```




```
# All look pretty much like the examples
```

```
# Currently not working, Error in `calcCurveGrob()`:
  # ! end points must not be identical
genustab <- get_taxadf(ps, taxlevel=6)
genustab <- data.frame(t(otu_table(genustab)), check.names=FALSE)
genustab <- data.frame(apply(genustab, 2, function(x)x/sum(x)), check.names=FALSE)

cortest <- WGCNA::corAndPvalue(genustab, method="spearman", alternative="two.sided")

cortest$cor[upper.tri(cortest$cor, diag = TRUE)] <- NA
cortest$p[upper.tri(cortest$p, diag = TRUE)] <- NA
cortab1 <- na.omit(melt(t(cortest$cor))) %>%
  dplyr::rename(from=Var1, to=Var2, cor=value) # It was using a different rename that
↪  wanted oldname = newname instead of dplyr's newname = oldname
corptab1 <- na.omit(melt(t(cortest$p))) %>%
  dplyr::rename(pvalue=value)
cortab1$fdr <- p.adjust(corptab1$pvalue, method="fdr")

cortab1 <- cortab1 %>% mutate(correlation=case_when(cor>0 ~ "positive",cor < 0 ~
↪  "negative",TRUE ~ "No"))
cortab2 <- cortab1 %>% filter(fdr <= 0.05) %>% filter(cor <= -0.5 | cor >= 0.8)

p <- ggdiffclade(
         obj=deres,
         alpha=0.3,
         linewd=0.25,
```

```
        skpointsize=0.2,
        layout="inward_circular",
        taxlevel=7,
        cladetext=0,
        setColors=FALSE,
        xlim=16
    ) +
    scale_fill_manual(values=c("#00AED7", "#FD9347"),
                      guide=guide_legend(keywidth=0.5,
                                         keyheight=0.5,
                                         order=3,
                                         override.aes=list(alpha=1))
    ) +
    scale_size_continuous(range=c(1, 3),
                      guide=guide_legend(keywidth=0.5,keyheight=0.5,order=4,
                                         override.aes=list(shape=21))) +
    scale_colour_manual(values=rep("white", 100),guide="none")

p2 <- p +
     new_scale_color() +
     new_scale("size") +
     geom_tiplab(size=1, hjust=1) +
     geom_taxalink(
         data=cortab2,
         mapping=aes(taxa1=from,
                     taxa2=to,
                     colour=correlation,
                     size=abs(cor)),
         alpha=0.4,
         ncp=10,
         hratio=1,
         offset=1.2
     ) +
     scale_size_continuous(range = c(0.2, 1),
                           guide=guide_legend(keywidth=1, keyheight=0.5,
                                              order=1, override.aes=list(alpha=1))
     ) +
     scale_colour_manual(values=c("chocolate2", "#009E73"),
                         guide=guide_legend(keywidth=0.5,
                                            keyheight=0.5,
                                            order=2,
                                            override.aes=list(alpha=1, size=1)))
p2
```