MicrobiotaProcessWorkshop

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")`

```
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(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.
})
```

```
## 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 '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
```

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="")
# 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
```

```
## ...
```

```
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.

```
# 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"))
```

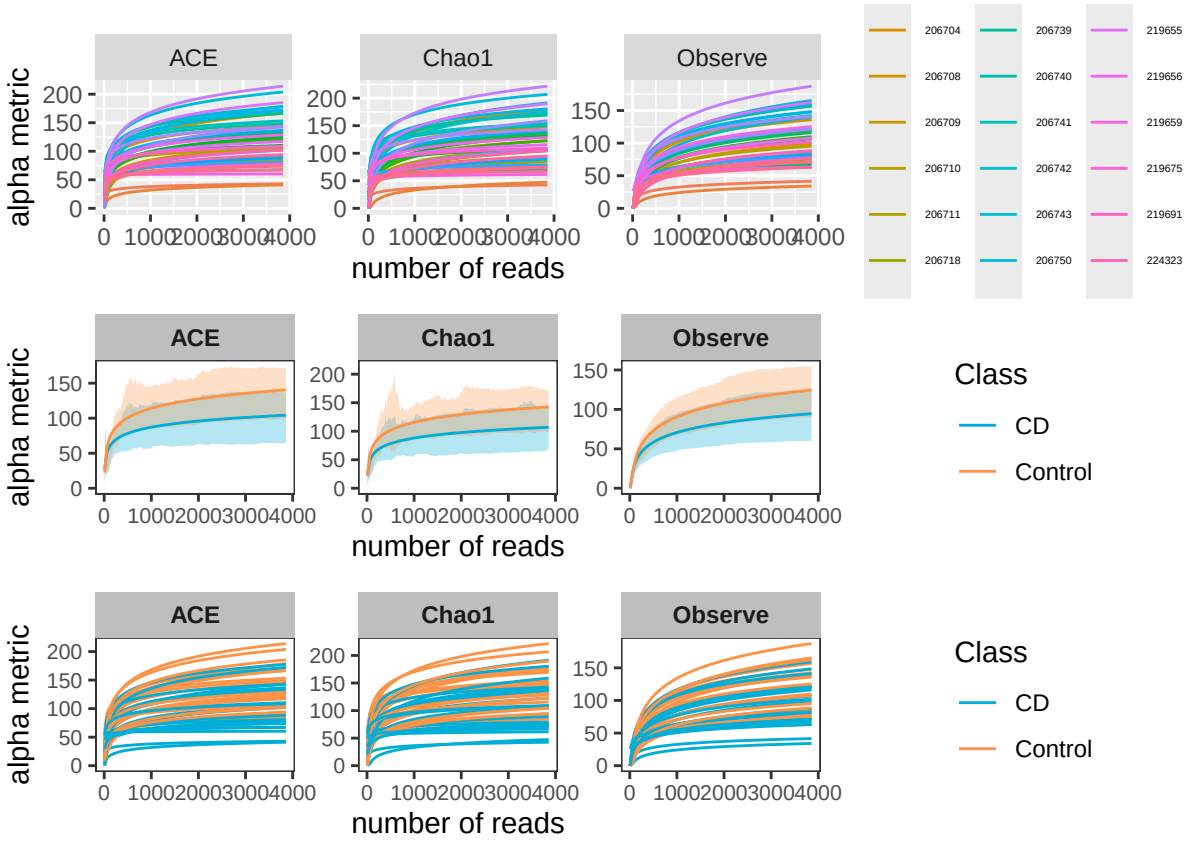
```
## The color has been set automatically, you can reset it manually by adding scale_color_manual(values=
```

```
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.

Diversity Measures

```
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
```

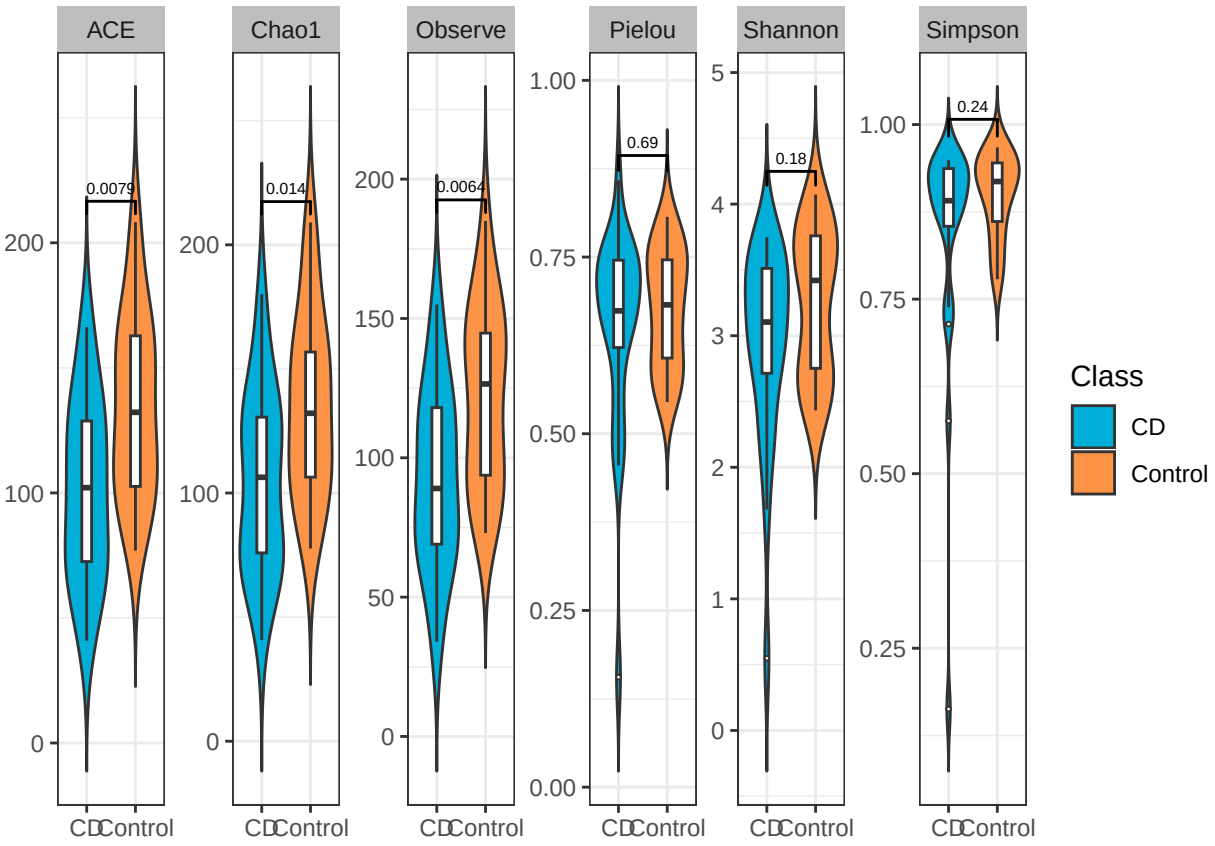
```
# This will make a dataframe of 6 common diversity measures
```

```
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

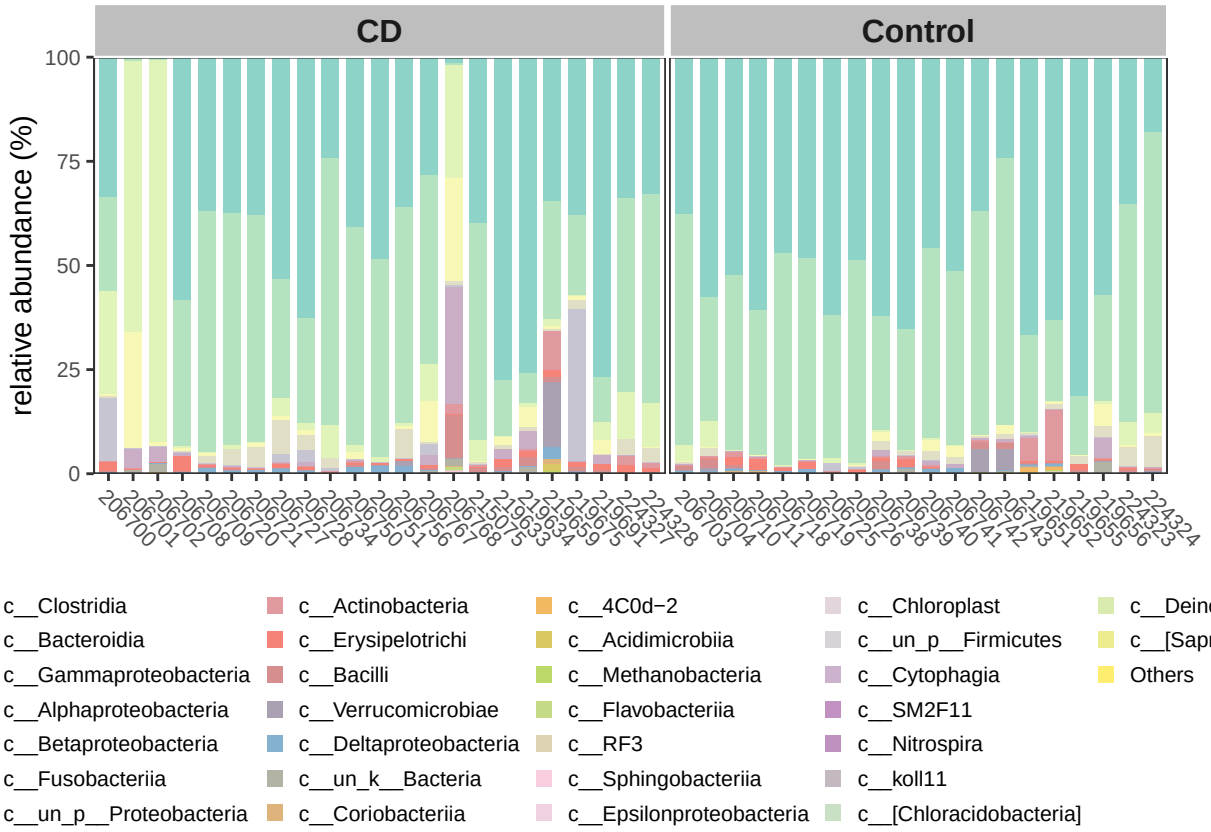
- ggbartax() can visualize community compositions
  - Use get_taxadf() first to visualize abundance of specific levels of class

```
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.
```

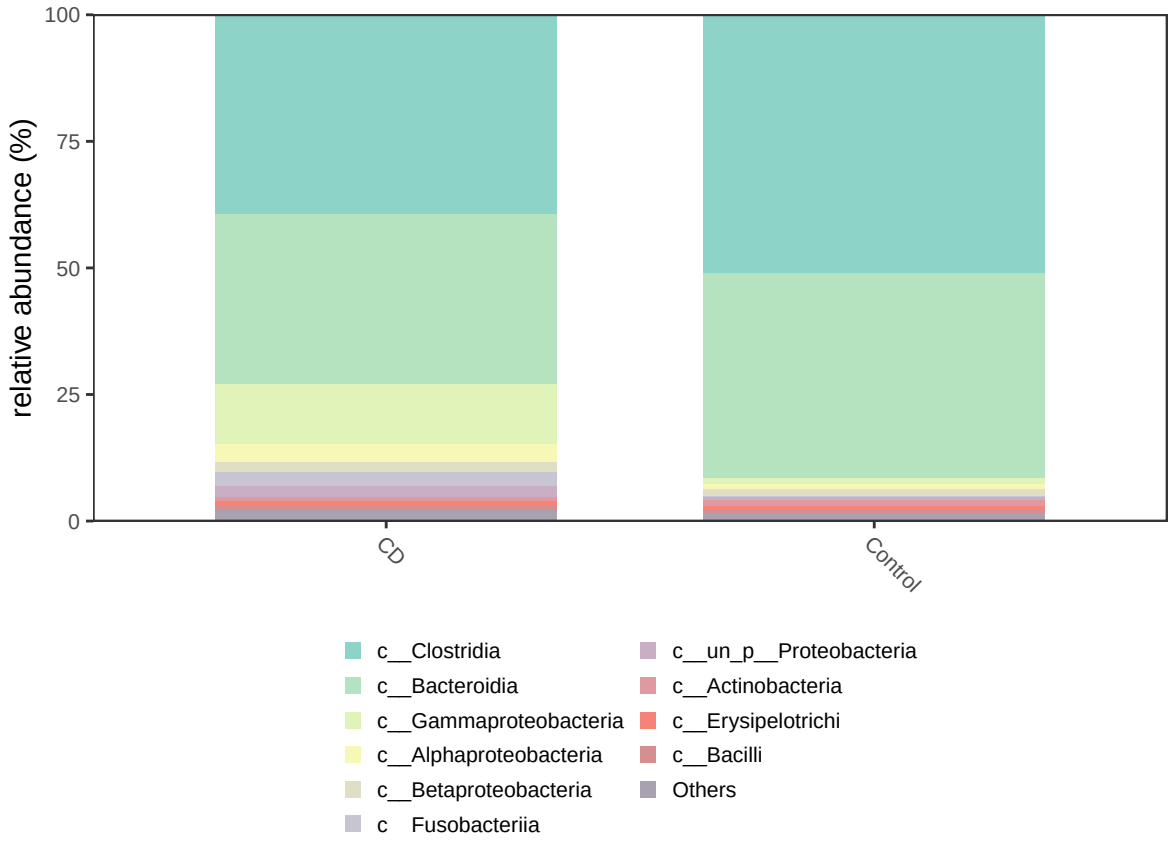
```
pclass
```




```
# 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.
```

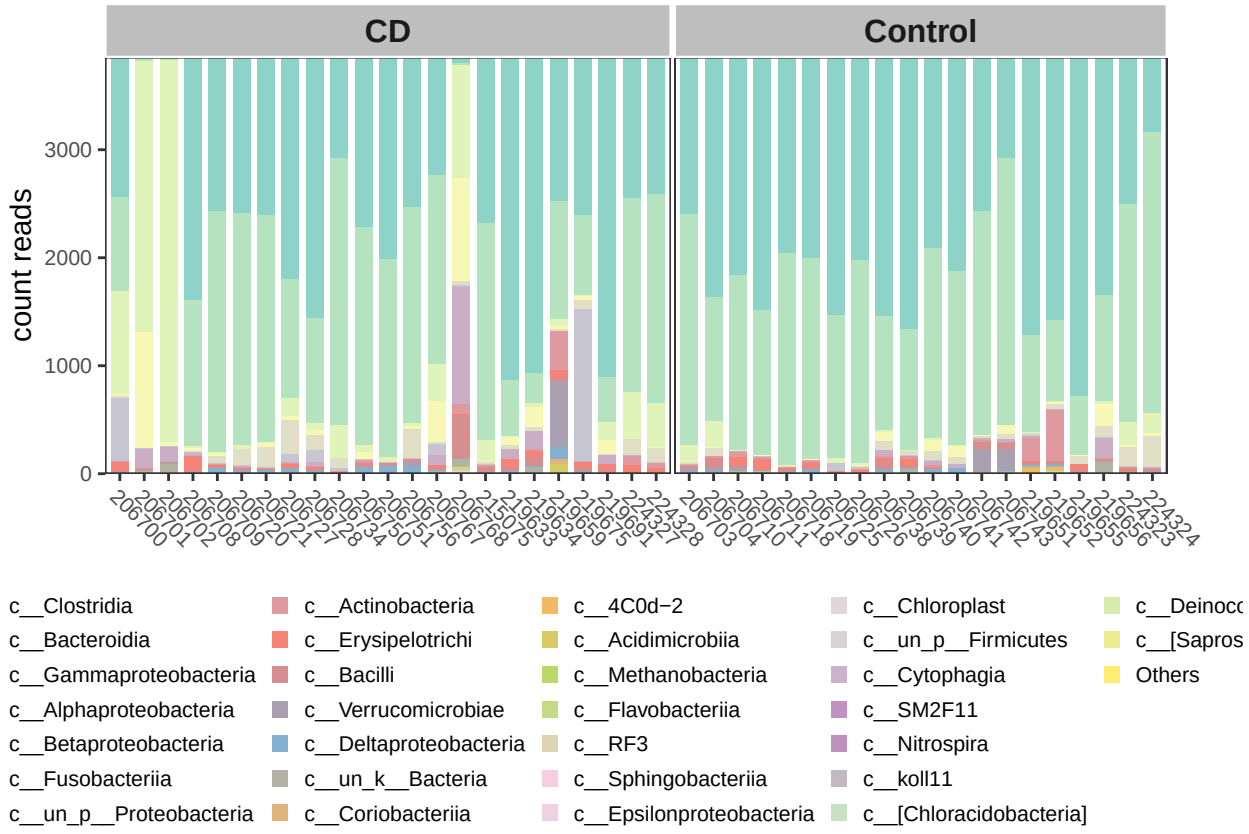
```
fclass
```




```
# 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))
```

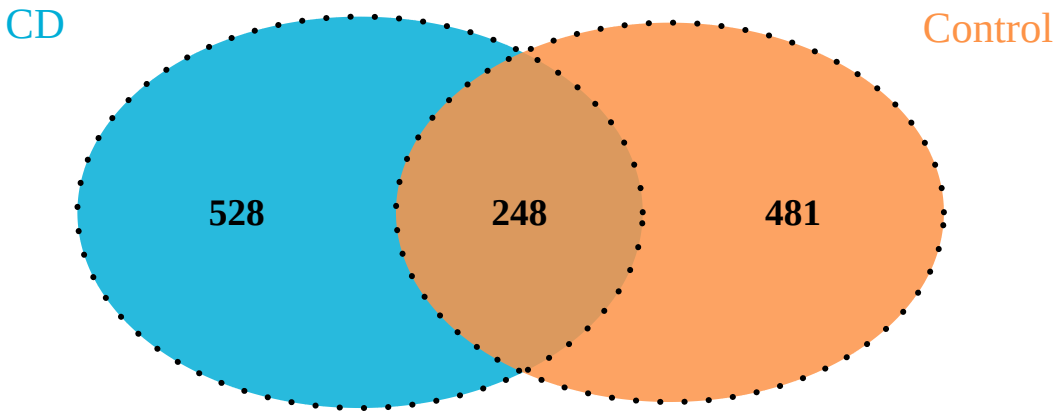
```
## The color has been set automatically, you can reset it
##             manually by adding scale_fill_manual(values=yourcolors)
## Scale for fill is already present.
## Adding another scale for fill, which will replace the existing scale.
```

```
pclass2
```

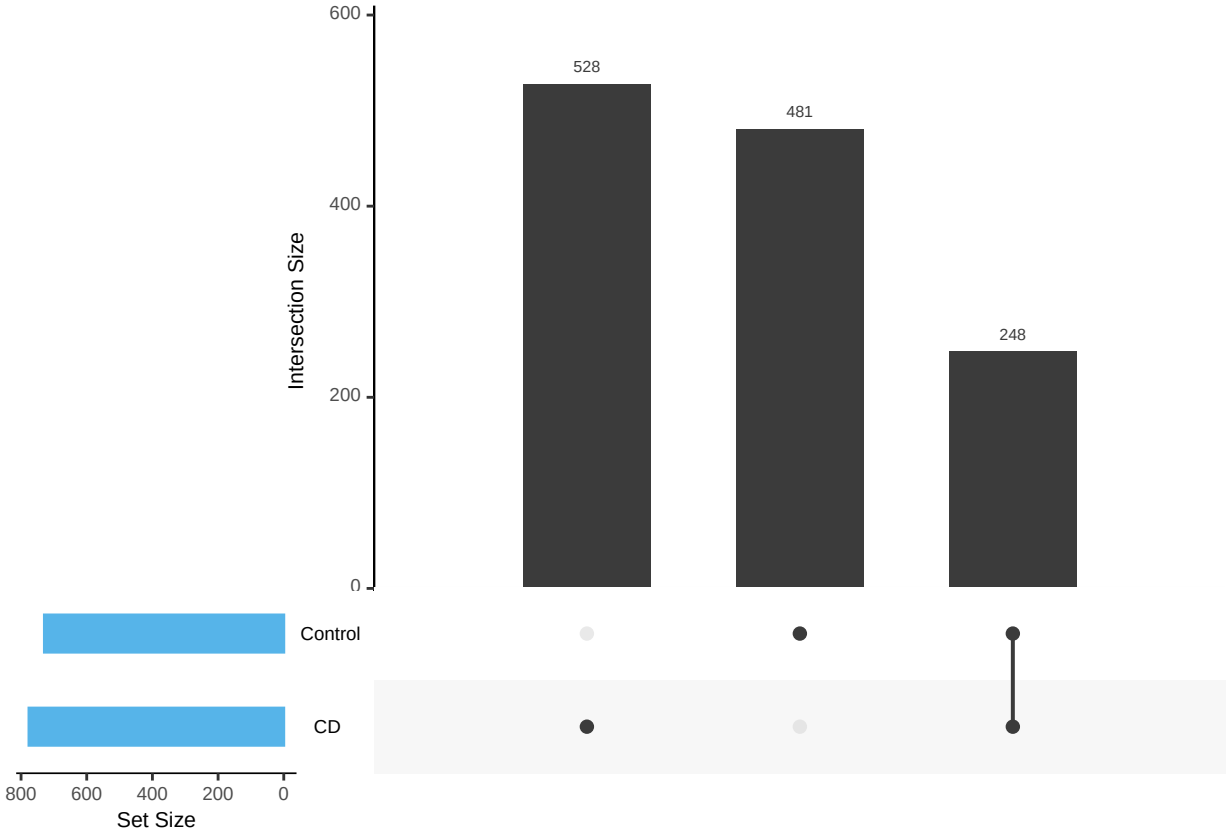


Venn plots can show the differences between groups. MicrobiotaProcess can use VennDiagram or UpSet.

```
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 analysis

- 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)

```
# If the input was normalized, the method parameter should be setted NULL.
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
```