The Silent Scourge: Investigating Soil-Borne Fungal Diseases in Wheat Crops

2024-01-24

The Silent Scourge: Investigating Soil-Borne Fungal Diseases in Wheat Crops

The fungi-wheat relationship significantly affects agricultural ecosystems and productivity. Wheat root rot disease, caused by soil-borne fungi, leads to major crop losses. This study explores the link between soil fungal diversity and wheat roots across nine farms to assess the prevalence of these pathogens.

Qiime2 to Phyloseq

Converting outcomes from Qiime2 data structure to the phyloseq object structure in R using Qiim2R package. Also, the phyloseq package provides functions for organising and manipulating data within the phyloseq object, allowing for diverse analyses such as diversity assessments, differential abundance testing, and taxonomic profile visualisation.

```r
# Download qiime2R from Github
# if (!requireNamespace("devtools", quietly = TRUE))
#   install.packages("devtools")
# devtools::install_github("jbisanz/qiime2R")
library("qiime2R")

# Download phyloseq from CRAN
# if (!require("BiocManager", quietly = TRUE))
#   install.packages("BiocManager")
# BiocManager::install("phyloseq")
library("phyloseq")
# install.packages("tidyverse")
library("tidyverse")
# install.packages("ggpubr")
library("ggpubr")
```

Install and load the required packages for the conversion

```r
# Convert qiime2 to phyloseq format
physeq.a <- qza_to_phyloseq(
  features = "ITS/table-its-fungi-with-phyla-no-mitochondria-no-chloroplast.qza", # table.qza
  tree = "ITS/phylogeny-align-to-tree-mafft-fasttree/rooted_tree.qza",
```

```
  taxonomy = "ITS/taxonomy-fungi.qza",
  metadata = "ITS/meta-table.txt"
)
physeq.a ## confirm the object
```

The conversion of Qiime2 to Phyloseq datastructure

```
## phyloseq-class experiment-level object
## otu_table()   OTU Table:         [ 3904 taxa and 37 samples ]
## sample_data() Sample Data:       [ 37 samples by 7 sample variables ]
## tax_table()   Taxonomy Table:    [ 3904 taxa by 7 taxonomic ranks ]
## phy_tree()    Phylogenetic Tree: [ 3904 tips and 3878 internal nodes ]
```

```
## Merge the replicate samples for each Group
physeq.a.group = merge_samples(physeq.a, "Group") # Sum between replicate samples

# (3B) repair factors in the sample metadata
sample_data(physeq.a.group)$Group <- levels(sample_data(physeq.a)$Group)[get_variable(physeq.a.group, "(
sample_data(physeq.a.group)$Soil.Type <- levels(sample_data(physeq.a)$Soil.Type)[get_variable(physeq.a.g
sample_data(physeq.a.group)$Soil <- levels(sample_data(physeq.a)$Soil)[get_variable(physeq.a.group, "So:
```

Assessing ASV distribution (merged and filtered)   We evaluated the number of Amplicon Sequence
Variants (ASVs) and visualised the raw read counts per sample within each group. This provided insights into
the distribution of ASVs and the sequencing depth of each sample, allowing for a comprehensive assessment
of data quality and quantity.

```
# install.packages("ggplot2")
library("ggplot2")
# install.packages("RColorBrewer")
library("RColorBrewer")
```

```
ASV <- sample_sums(physeq.a)
ASV <- as.data.frame(ASV)
ASV$Group <- physeq.a@sam_data$Group

#### pdf
# pdf(file = "Suppl_ASV_ITS.pdf", width = 8,height = 5)
ggplot(ASV, aes(x = Group, y = ASV,  colour = Group))+
  scale_color_brewer(palette = "Paired") +
  geom_point(alpha = 1, position = "jitter", size = 4) +
  geom_boxplot(alpha = 0, colour = "black", size = 0.8)+
  theme_classic() +
  theme(text = element_text(size=15, colour = "black"),
        axis.ticks = element_line(colour = "black", size = 1.25),
        axis.line = element_line(colour = 'black', size = 1.25),
        axis.text.x = element_text(angle=45, hjust=1,
                                   colour = "black", size = 13),
        axis.text.y = element_text(angle=0, hjust=0.5,
                                   colour = "black",size = 13),
        axis.title.y = element_text(color="black", size=15,face="bold"),
        legend.position = "none")
```
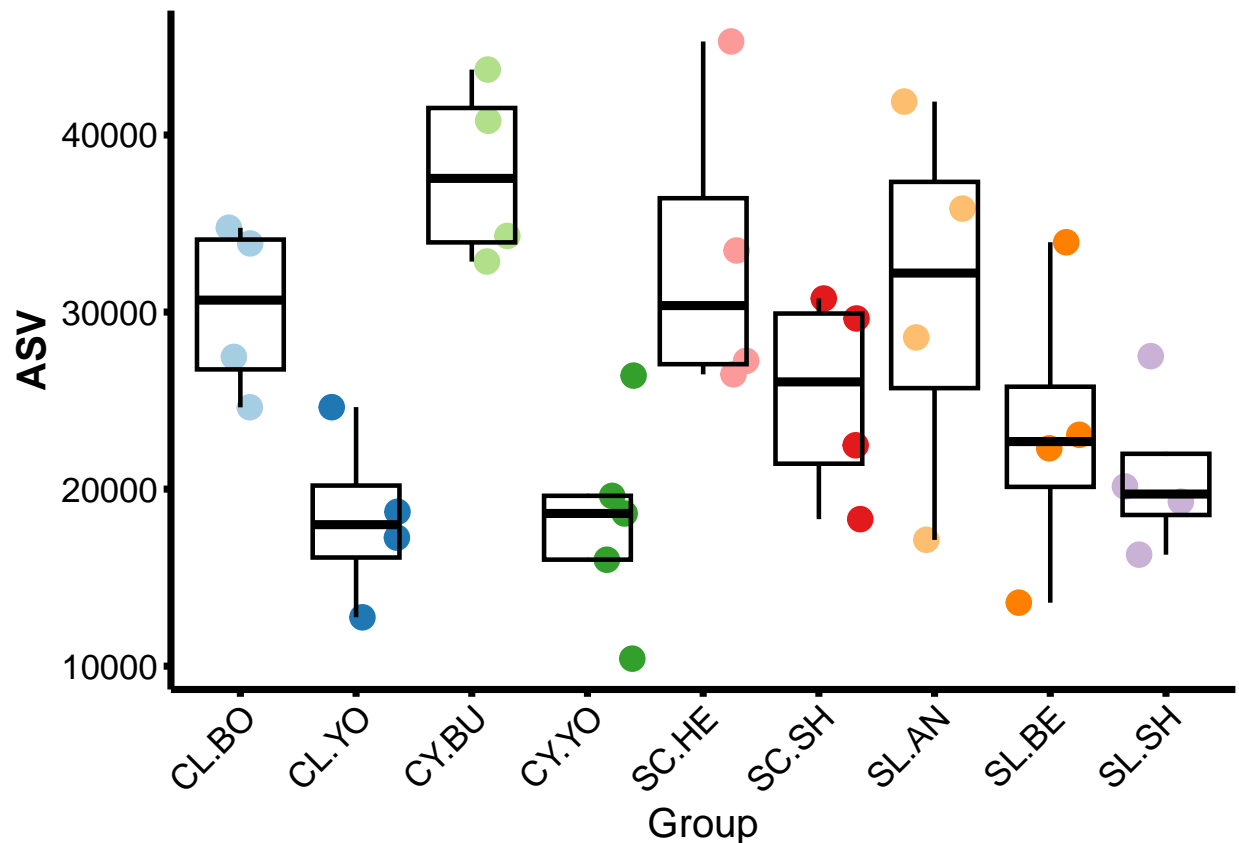
```
# Close the PDF device and save the plot to a file
# dev.off()

rm(ASV)
```

Beta diversity   Beta diversity is a metric used in microbial community studies to assess species composition dissimilarity between samples, providing insights into microbial community dynamics and their ecological significance. By quantifying variation in community structure and employing visualisation techniques, beta diversity analysis allows comparisons across habitats or treatments, revealing key drivers of community variation and functional implications. It aids our understanding of microbial assemblages and their responses to environmental changes in ecology, environmental science, and human health research.

```
# method options: NMDS / PCoA
NMDS <- ordinate(physeq = physeq.a, method = "NMDS", distance = "bray")
```

```
## Square root transformation
## Wisconsin double standardization
## Run 0 stress 0.196196
## Run 1 stress 0.196196
## ... Procrustes: rmse 2.841158e-06  max resid 8.065493e-06
## ... Similar to previous best
## Run 2 stress 0.2096687
## Run 3 stress 0.2110517
## Run 4 stress 0.196196
```

```
## ... Procrustes: rmse 4.970957e-06  max resid 1.160098e-05
## ... Similar to previous best
## Run 5 stress 0.196196
## ... New best solution
## ... Procrustes: rmse 4.679965e-06  max resid 2.024994e-05
## ... Similar to previous best
## Run 6 stress 0.2496971
## Run 7 stress 0.2180555
## Run 8 stress 0.1929835
## ... New best solution
## ... Procrustes: rmse 0.1085377  max resid 0.2861609
## Run 9 stress 0.2110518
## Run 10 stress 0.2457738
## Run 11 stress 0.196196
## Run 12 stress 0.1929835
## ... New best solution
## ... Procrustes: rmse 3.488217e-06  max resid 1.075266e-05
## ... Similar to previous best
## Run 13 stress 0.2490951
## Run 14 stress 0.1929835
## ... Procrustes: rmse 2.175218e-06  max resid 5.524086e-06
## ... Similar to previous best
## Run 15 stress 0.2502029
## Run 16 stress 0.1929835
## ... Procrustes: rmse 4.285922e-06  max resid 1.317781e-05
## ... Similar to previous best
## Run 17 stress 0.196196
## Run 18 stress 0.2225928
## Run 19 stress 0.1929835
## ... Procrustes: rmse 2.286218e-06  max resid 4.91364e-06
## ... Similar to previous best
## Run 20 stress 0.196196
## *** Best solution repeated 4 times
```

```r
# pdf
# pdf(file = "Fig9A_bata_ITS.pdf", width = 8,height = 5)
# Plot ordination
plot_ordination(physeq = physeq.a,
                 ordination = NMDS,
                 color = "Group",
                 shape = "Group"
) +
  theme_classic() +
  scale_color_brewer(palette = "Paired") +
  scale_fill_brewer(palette = "Paired") +
  geom_point(aes(color = Group), alpha = 1, size = 3) +
  theme(
    text = element_text(size = 18, colour = "black"),
    axis.ticks = element_line(colour = "black", size = 1.1),
    axis.line = element_line(colour = 'black', size = 1.1),
    axis.text.x = element_text(colour = "black", angle = 0, hjust = 0.5,
                               size = 13, face = "bold"),
    axis.text.y = element_text(colour = "black", angle = 0, hjust = 0.5,
                               size = 13, face = "bold"),
```

```
    axis.title.y = element_text(color = "black", size = 20, face = "bold"),
    axis.title.x = element_text(color = "black", size = 20, face = "bold")
) + stat_ellipse(geom = "polygon", type="norm", alpha=0.25, aes(fill = Group)) +
  scale_shape_manual(values = c(15, 17, 3, 4, 16, 18, 21, 22, 23)) # Set custom shapes
```
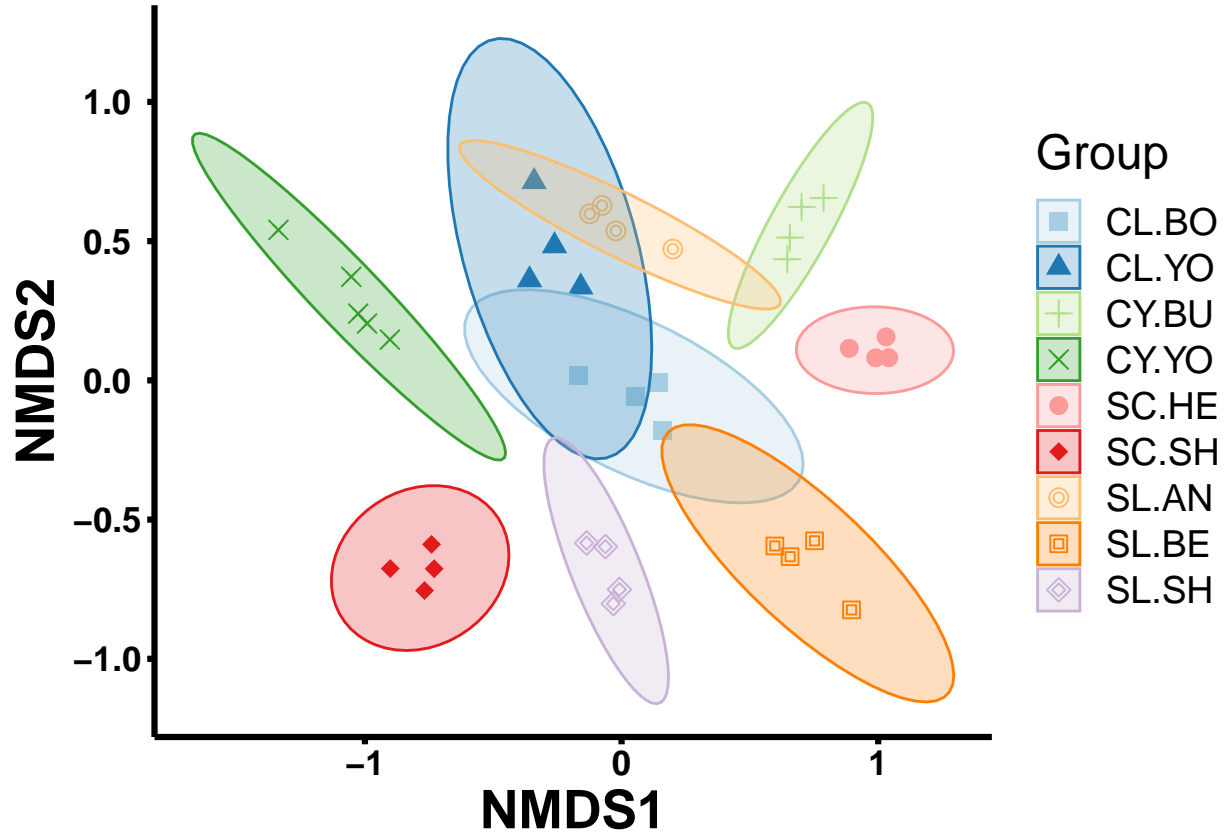


```
# Close the PDF device and save the plot to a file
# dev.off()

# Clean up by removing objects that are no longer needed
rm(NMDS)
```

Alpha diversity   Alpha diversity, a key ecological concept, denotes species richness within a specific habitat or community. In microbial ecology, it signifies the variety and distribution of microorganisms in a sample or microbiome. It's measured by species richness and evenness, reflecting the number and distribution of species respectively. Alpha diversity is vital for assessing the stability and functional potential of microbial communities and can be influenced by environmental conditions, host factors, and disturbances. Comparing alpha diversity across samples or groups allows researchers to understand the effects of factors like disease or habitat changes on microbial community structure.

Here we use Shannon for alpha diversity.

```
# available measurements [c("Observed", "Chao1", "ACE", "Shannon", "Simpson", "InvSimpson", "Fisher")]
alpha.object <- cbind(
  x = sample_data(physeq.a),
  y = estimate_richness(physeq.a, measures = 'Shannon')
```

```
)

#### pdf
# pdf(file = "Fig9B_Alpha_ITS.pdf", width = 8,height = 5)
# Create a plot for alpha diversity
ggplot(data = alpha.object, aes(x = x.Group, y = Shannon, color = x.Group)) +
  scale_color_brewer(palette = "Paired") +
  theme_classic() +
  labs(
    x = element_blank(),                    # No x-axis label
    y = "Alpha Diversity (Shannon)"         # y-axis label
  ) +
  geom_point(alpha = 1, position = "jitter", size = 4) +
  geom_boxplot(alpha = 0, colour = "black", size = 0.8)+
  theme(
    text = element_text(size = 18, colour = "black"),
    axis.ticks = element_line(colour = "black", size = 1.1),
    axis.line = element_line(colour = 'black', size = 1.1),
    axis.text.x = element_text(colour = "black", angle = 45,
                               hjust = 1, size = 13, face = "bold"),
    axis.text.y = element_text(angle = 0, hjust = 0,
                               colour = "black", size = 13, face = "bold"),
    axis.title.y = element_text(color = "black", size = 15, face = "bold"),
    legend.position = "none"                 # Hide legend
  )
```
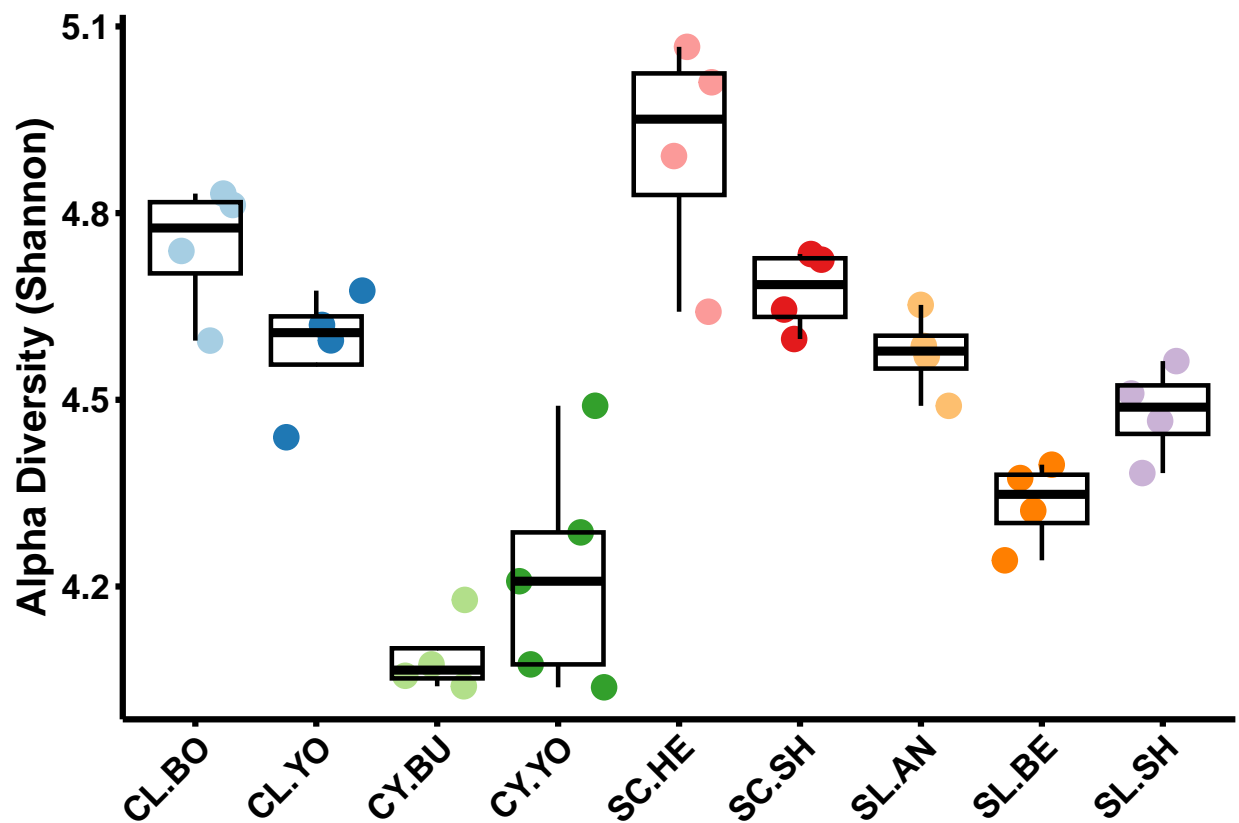
```
# Close the PDF device and save the plot to a file
# dev.off()


# Clean up by removing the alpha.object
rm(alpha.object)
```

The proportions of different microbial taxa   Bar plots illustrating relative abundance in microbiome enable to visually represent the proportions of different microbial taxa within a given sample. Each bar corresponds to a specific taxonomic group, with its height indicating the relative abundance of that group in the microbiome.

```
# install.packages("plyr")
library("plyr")


# https://github.com/joey711/phyloseq/issues/901


# Filter taxa with mean abundance > 0.1
physeq2 = filter_taxa(physeq.a.group, function(x) mean(x) > 0.1, TRUE)


# Transform sample counts to relative abundances
physeq3 = transform_sample_counts(physeq2, function(x) x / sum(x))


# Aggregate taxa at the Phylum level
glom <- tax_glom(physeq3, taxrank = 'Phylum')


# Convert the aggregated phyloseq object to a dataframe
data <- psmelt(glom)
# Convert Phylum column to character
data$Phylum <- as.character(data$Phylum)


# Rename phyla with abundance < 1% to "< 1% abund."
data$Phylum[data$Abundance < 0.01] <- "< 1% abund."


# Calculate medians of abundance for each Phylum
medians <- ddply(data, ~Phylum, function(x) c(median = median(x$Abundance)))


# Extract phyla with median abundance <= 0.01
remainder <- medians[medians$median <= 0.01, ]$Phylum


# Rename selected phyla to "Phyla < 1% abund."
data[data$Phylum %in% remainder, ]$Phylum <- "Phyla < 1% abund."


# Rename phyla with abundance < 1% to "Phyla < 1% abund."
data$Phylum[data$Abundance < 0.01] <- "Phyla < 1% abund."


# Plot the data with condensed phyla categories
p <- ggplot(data = data, aes(x = Group, y = Abundance, fill = Phylum))
p + geom_bar(aes(), stat = "identity", position = "stack") +
    scale_color_brewer(palette = "Paired") +
  theme_classic() +
  theme(text = element_text(size=15, colour = "black"),
```

```
        axis.ticks = element_line(colour = "black", size = 1.25),
        axis.line = element_line(colour = 'black', size = 1.25),
        axis.text.x = element_text(angle=45, hjust=1,
                                   colour = "black", size = 13),
        axis.text.y = element_text(angle=0, hjust=0.5,
                                   colour = "black",size = 13),
        axis.title.y = element_text(color="black", size=15,face="bold"),
        legend.position = "right")
```
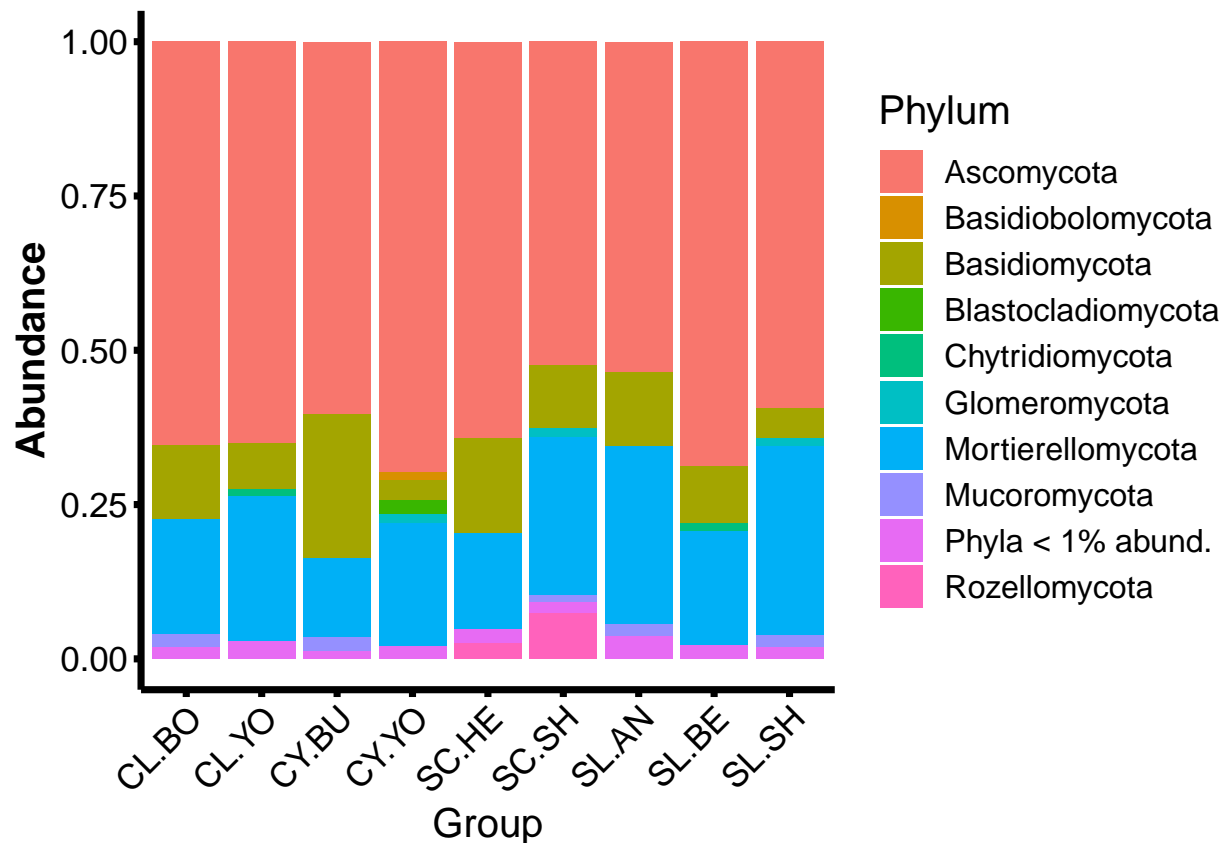


```
# Clean up by removing the alpha.object
rm(physeq2, physeq3, glom, data, medians, remainder, p)
```

Number of taxa in groups Examining the cumulative count of taxa within a particular taxonomic group for each distinct subgroup involves the summation of the total number of taxa within that group while eliminating any overlapping taxa based on their names within individual subgroups. This process aids in comprehending the dynamics of microbiome populations.

```
library("dplyr")
library("reshape2")
```

```
physeq <- merge_samples(physeq.a, "Group", fun = sum)
gentab_levels <- list()
observationThreshold <- 1
```

```r
# Define the taxonomic levels
genus_levels <- c("Phylum", "Class", "Order", "Family", "Genus", "Species")

# Something wrong with Kingdom Level, removed from the analysis
# loop through all the taxonomic levels
  for (level in genus_levels) {

    # create a factor variable for each level
    genfac <- factor(tax_table(physeq)[, level])

    # calculate the abundance of each genus within each sample
    gentab <- apply(otu_table(physeq), MARGIN = 1, function(x) {
      tapply(x, INDEX = genfac, FUN = sum, na.rm = TRUE, simplify = TRUE)
    })

    # calculate the number of samples in which each genus is observed above the threshold
    level_counts <- apply(gentab > observationThreshold, 2, sum)

    # create a data frame of level counts with genus names as row names
    BB <- as.data.frame(level_counts)
    BB$Group <- row.names(BB)

    # add the data frame to the gentab_levels list
    gentab_levels[[level]] <- BB
  }

# Combine all level counts data frames into one data frame
B2 <- gentab_levels %>% reduce(full_join, by = "Group")

# Set row names and column names
row.names(B2) <- B2$Group
B2$Group <- NULL
colnames(B2)[1:6] <- genus_levels

print(B2)
```

```
##        Phylum Class Order Family Genus Species
## CL.BO      14    38    80    162   233     263
## CL.YO      14    35    70    129   189     218
## CY.BU      14    37    70    137   191     210
## CY.YO      13    36    66    128   180     206
## SC.HE      13    41    87    165   245     282
## SC.SH      14    43    85    174   259     305
## SL.AN      14    36    81    156   230     268
## SL.BE      14    41    77    144   202     227
## SL.SH      14    36    78    154   241     276
```

```r
# write.csv(B2, file = "ITS/level_counts_by_group.csv", row.names = TRUE)
B2$Group = row.names(B2)
data_long <- melt(B2)
data_long <- melt(B2, id.vars = "Group",
                  variable.name = "Level",
                  value.name = "Taxa")
```

```r
# Plot the data as a line graph using ggplot
# Open a new PDF graphics device
# pdf(file = "Fig7C_line_graph_ITS.pdf", width=8,height=5)
ggplot(data_long, aes(x = Level, y = Taxa,
                      color = Group, group = Group)) +
  geom_line() +
  geom_point(size = 4) +
  labs(x = "Taxonomic Level", y = "Count", color = "Group") +
  theme_classic() +   scale_color_brewer(palette = "Paired") +
  theme(
    text = element_text(size = 19, colour = "black"),
    axis.ticks = element_line(colour = "black", size = 1.1),
    axis.line = element_line(colour = 'black', size = 1.5),
    axis.text.x = element_text(colour = "black", angle = 0, hjust = 0.5,
                               size = 13, face = "bold"),
    axis.text.y = element_text(colour = "black", angle = 0, hjust = 0.5,
                               size = 13, face = "bold"),
    axis.title.y = element_text(color = "black", size = 14, face = "bold"),
    axis.title.x = element_text(color = "black", size = 14, face = "bold"),
    legend.title = element_text(size = 13.5, face = "bold"),
    legend.text = element_text(size = 12),
    legend.key.size=unit(0.4,"cm")
  ) +
  scale_x_discrete(guide = guide_axis(n.dodge=2)) +
  scale_y_continuous(breaks=seq(0,400,by=100))
```
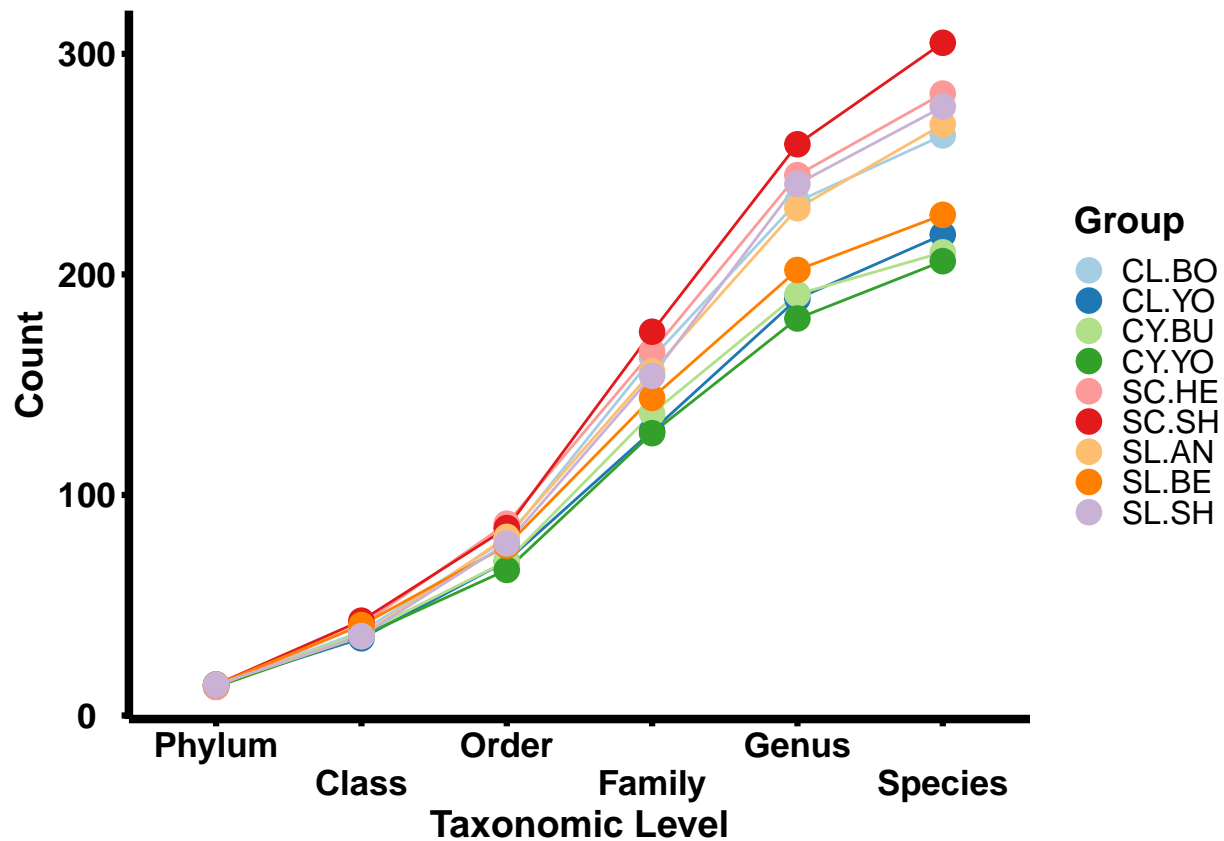
```
# Close the PDF device and save the plot to a file
# dev.off()

# Clean up by removing the alpha.object
rm(physeq, gentab_levels, observationThreshold, B2, data_long, genfac, genus_levels, level, level_count
```

Plotting the top 10 taxa at family level   Generating a visual representation through the plotting of the top 10 taxa at the family level offers a concise overview of the predominant microbial groups within a dataset. This approach is instrumental in identifying and emphasizing significant contributors to the structure and function of microbial communities across diverse groups or conditions.

```
## Transform normalised ASVs to proportions
proportions = transform_sample_counts(physeq.a.group, function(x) 100 * x/sum(x))

##  S
Top10ASVs = names(sort(taxa_sums(proportions), TRUE)[1:13])
Taxtab10 = cbind(tax_table(proportions), Family30 = NA)
Taxtab10[Top10ASVs, "Family30"] <- as(tax_table(proportions)[Top10ASVs, "Family"], "character")
tax_table(proportions) <- tax_table(Taxtab10)

Rgsm10 = prune_taxa(Top10ASVs, proportions)

# plotting
title = "Top 10 - Family level"
plot_bar(Rgsm10, "Group", fill = "Family", title = title) +
```

```
coord_flip() +
ylim(0, 50) +
scale_color_brewer(palette = "Paired") +
theme_classic() +
theme(text = element_text(size=15, colour = "black"),
      axis.ticks = element_line(colour = "black", size = 1.25),
      axis.line = element_line(colour = 'black', size = 1.25),
      axis.text.x = element_text(angle=45, hjust=1,
                                 colour = "black", size = 13),
      axis.text.y = element_text(angle=0, hjust=0.5,
                                 colour = "black",size = 13),
      axis.title.y = element_text(color="black", size=15,face="bold"),
      legend.position = "right",
      legend.text = element_text(size = 9.5),
      legend.key.height= unit(0.45, 'cm'),
      legend.key.width= unit(0.45, 'cm')
      )
```


Top 10 – Family level
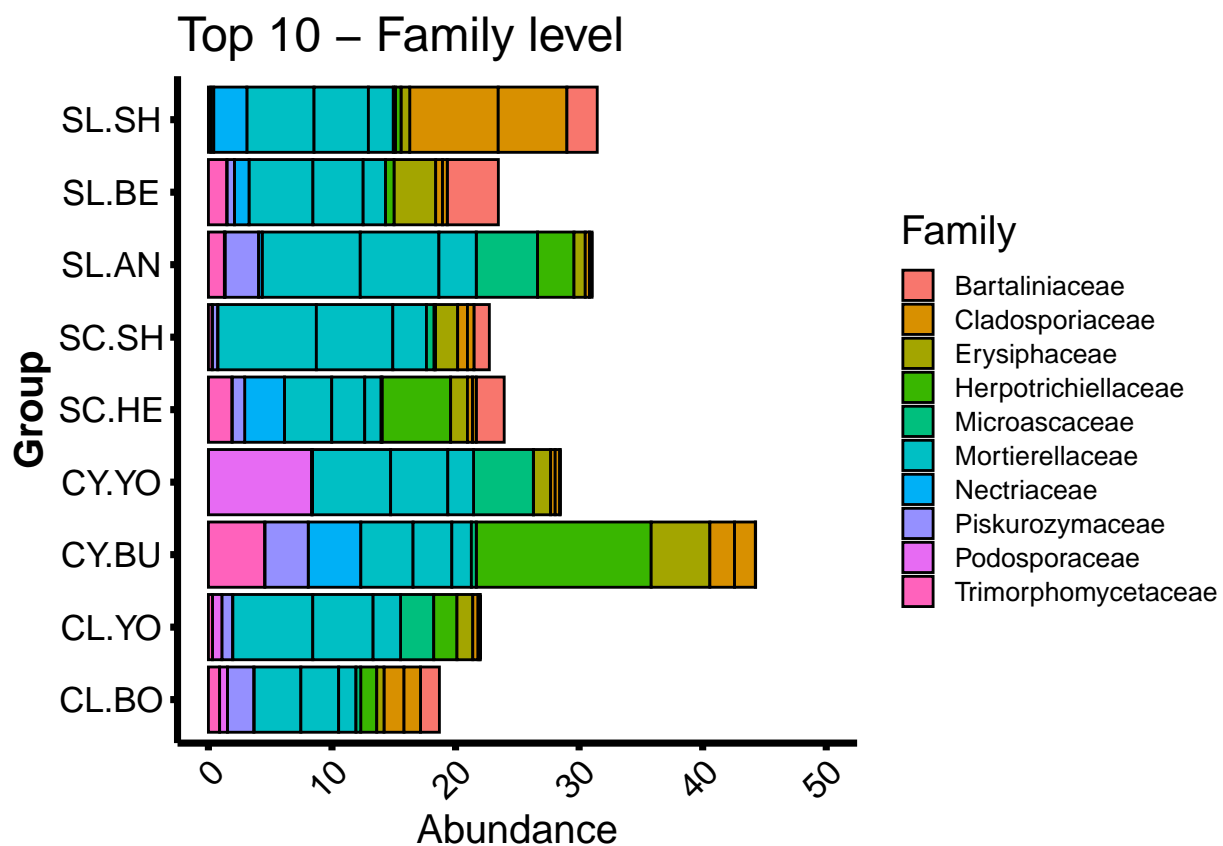
```
# Clean up by removing unnecessary objects
rm(proportions, Top10ASVs, Taxtab10, Rgsm10, title)
```

Upset plot using UpsetR

Venn diagrams are commonly used for visualizing sets, but they can become complex with more than five sets. UpSet graphs, on the other hand, offer a more efficient way to display intersections and complements,

especially for larger or multiple datasets. They provide a more intuitive and informative data representation.

```r
library("UpSetR")
library("reshape2")
library("plyr")
library("dplyr")
# BiocManager::install("microbiome")
library("microbiome")
```

```r
# Aggregate taxa at the genus level
B <- aggregate_taxa(physeq.a.group, "Genus", verbose = TRUE)
```

```
## [1] "Remove taxonomic information below the target level"
## [1] "Mark the potentially ambiguous taxa"
## [1] "-- split"
## [1] "-- sum"
## [1] "Create phyloseq object"
## [1] "Remove ambiguous levels"
## [1] "-- unique"
## [1] "-- Rename the lowest level"
## [1] "-- rownames"
## [1] "-- taxa"
## [1] "Convert to taxonomy table"
## [1] "Combine OTU and Taxon matrix into Phyloseq object"
## [1] "Add the metadata as is"
```

```r
# Remove undesired genera
# B2 <- subset_taxa(B, !get("Genus") %in% c("uncultured", "Unknown"))

# Remove unwanted taxon names
taxa_to_remove <- c("uncultured", "Unknown")
B2 <- subset_taxa(B, !get("Genus") %in% taxa_to_remove)

# Extract relevant data from the phyloseq object
sample_data <- sample_data(B2)
otu_table <- otu_table(B2)
abundance <- as.vector(otu_table)

# Create a tibble with the extracted data
D <- tibble(
  Sample = rep(sample_data$Group, each = nrow(otu_table)),
  ASV = rep(rownames(otu_table), times = ncol(otu_table)),
  Abundance = abundance
) %>%
  group_by(Sample) %>%
  mutate(rank = rank(desc(Abundance))) %>%
   filter(Abundance > 0) %>%
  ungroup() %>%
  select(Sample, Abundance, ASV)

# Remove the Abundance column
D$Abundance <- NULL
```

```r
# Rename the second column to "ASV"
names(D)[2] <- "ASV"
names(D)[1] <- "Group"

# Convert data from long to wide format
E <- dcast(D, ASV ~ Group)

# Define a binary function
binary_fun <- function(x) {
  x[is.na(x)] <- 0
  ifelse(x > 0, 1, 0)
}

col = brewer.pal(n = 9, name = "Paired")

# Apply the binary function to columns 2 to 10
temp_df <- apply(E[2:10], 2, binary_fun)
temp_df <- as.data.frame(temp_df)

# Create an UpSet plot
upset_plot <- upset(temp_df,
                    sets = colnames(temp_df),
                    sets.bar.color = (col),
                    order.by = "freq",
                    empty.intersections = "on",
                    mainbar.y.label = "Counts by Pattern of Conditions",
                    sets.x.label = "Counts by Condition",
                    matrix.color="blue",
                    mb.ratio = c(0.6, 0.4),
                    point.size= 2.75,
                    line.size = 1.25,
                    text.scale = 1.5
)


# Open a new PDF graphics device
# pdf(file = "Fig7D_UpSet_ITS.pdf", width=8,height=5)

# Print the UpSet plot
print(upset_plot)
```

```
# Close the PDF device and save the plot to a file
# dev.off()
```

Phylogenetic tree

Phylogenetic trees are branching diagrams that show the evolutionary relationships between different species or other entities. They are constructed by comparing the organisms' DNA, proteins, or other features. The more similar the features, the more closely related the organisms are thought to be. It can be used to answer how different species are related to each other, when different species diverged from each other, and what are the common ancestors of different species.

Here, we concentrate on *Fusarium*, a large genus of filamentous fungi. These fungi, often referred to as hyphomycetes, are widely found in soil and are associated with plants. While most species are harmless saprobes and form a significant part of the soil microbial community, some can produce mycotoxins in cereal crops that can harm human and animal health if they enter the food chain. Despite the majority of species being harmless, certain Fusarium species and subspecific groups rank among the most critical fungal pathogens of plants and animals.

```
# Load required libraries
library("phyloseq")    # For handling phylogenetic sequencing data
# BiocManager::install("ggtree")
library("ggtree")      # For tree visualisation
library("scales")      # For scaling transformations
```

```r
# Load the GlobalPatterns dataset and prune taxa
GP <- prune_taxa(taxa_sums(physeq.a) > 0, physeq.a)  # Remove taxa with zero sums
GP.chl <- subset_taxa(GP, Genus == "Fusarium")   # Subset data based on Phylum value

# Create a ggtree plot
p <- ggtree(GP.chl, ladderize = TRUE) +
  # geom_text2(aes(subset = !isTip, label = label), hjust = -1.5, size = 3.5) +
  geom_tiplab(aes(label = Species), as_ylab=TRUE) +
  geom_point(aes(x = x + hjust, color = Group,
                 shape = Group, size = Abundance), na.rm = TRUE) +
  scale_size_continuous(trans = log_trans(2)) +
  scale_shape_manual(values = c(15, 17, 3, 4, 16, 18, 21, 22, 23)) + # Set custom
  scale_color_brewer(palette = "Paired") +
  theme(text = element_text(size=13, colour = "black"),
        axis.ticks = element_line(colour = "black", size = 1.25),
        axis.line = element_line(colour = 'black', size = 1.25) ,
        axis.title.y = element_text(color="black", size=2.5,face="bold"),
        legend.text = element_text(size = 8),
        legend.key.size=unit(0.4,"cm"),
        legend.title = element_text(size = 10, face = "bold"),
        legend.position = "bottom"
  )

# Open a new PDF graphics device
# pdf(file = "Fig10_Tree_ITS.pdf", width=11,height=5)
# Print the ggtree plot
print(p)
```
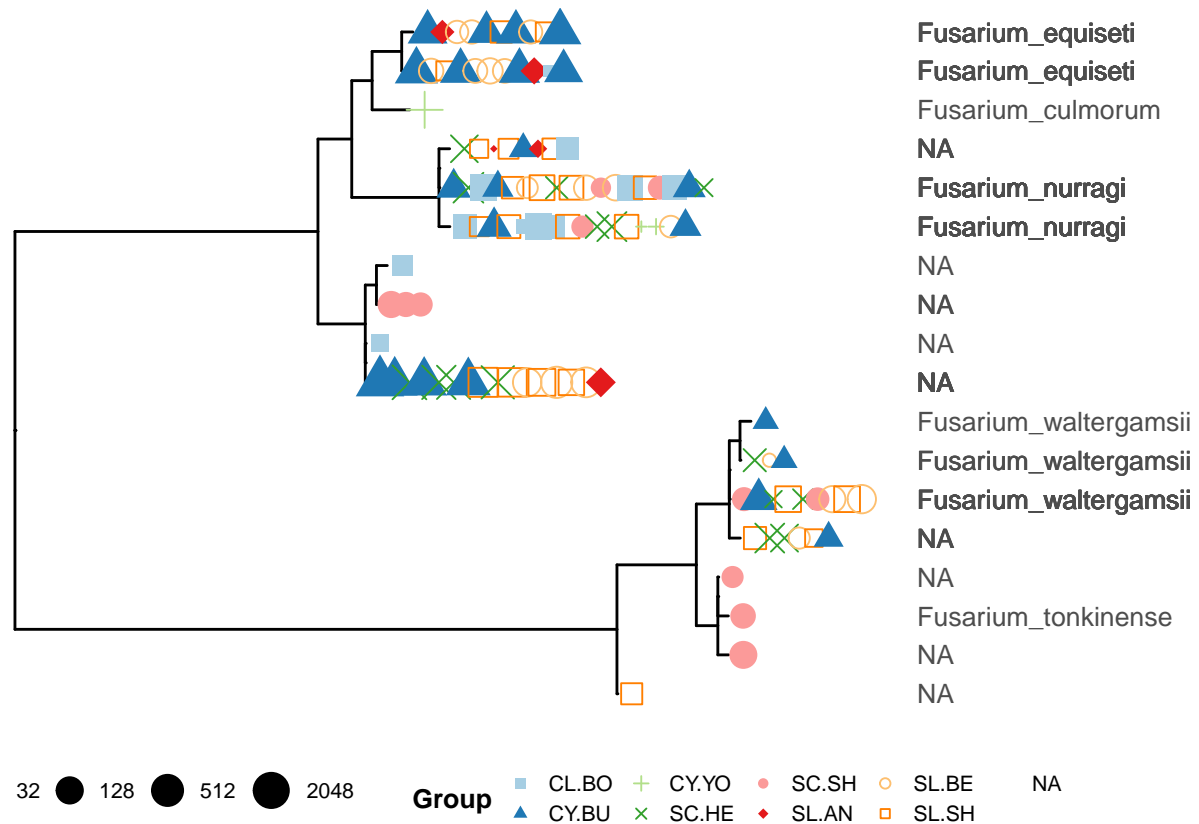
```
# Close the PDF device and save the plot to a file
# dev.off()
```

From the phylogenetic tree, it was obsered *Fusarium culmorum* and *Fusarium tonkinense* were identified in only one of the samples. However, there was an enrichment of *Fusarium equiseti* , *Fusarium nurragi and Fusarium waltergamsii.*

- *Fusarium equiseti* is a fungal species and plant pathogen on a varied range of crops. It is considered to be a weak pathogen on cereals and is occasionally found to be associated with 'Fusarium head blight' infected kernels. It is commonly found in tropical and subtropical areas.

- *Fusarium nurragi* has been isolated from heath species growing on the coasts in the Australian states of Victoria and Western Australia. Although *F. nurragi* has not been assigned to a Fusarium species complex, it is closely related to the *F. heterosporum* and *F. tricinctum* species complexes.

- *Fusarium waltergamsii* is a fungal species that was first isolated from the roots of wheat plants in Germany. It belongs to the *Fusarium oxysporum* species complex, which comprises many plant pathogens and opportunistic animal pathogens. *Fusarium waltergamsii* can cause wilt and root rot diseases in wheat and other cereals, reducing crop yield and quality.

```
# Plotting for the abundance of one specific fungi
# Subset the taxa to Genus from physeq.wheat
physeq.a.genus <- subset_taxa(physeq.a, Genus == "Fusarium")
physeq.a.equiseti <- subset_taxa(physeq.a, Species == "Fusarium_equiseti")
physeq.a.nurragi <- subset_taxa(physeq.a, Species == "Fusarium_nurragi")
physeq.a.waltergamsii <- subset_taxa(physeq.a, Species == "Fusarium_waltergamsii")
```

17

```r
# Calculate the total abundance of Fusarium for each sample
meta = physeq.a@sam_data
otudf = as.data.frame(t(as.data.frame(physeq.a.genus@otu_table)))
meta$Fusarium = rowSums(otudf)
otudf = as.data.frame(t(as.data.frame(physeq.a.equiseti@otu_table)))
meta$F.equiseti = rowSums(otudf)
otudf = as.data.frame(t(as.data.frame(physeq.a.nurragi@otu_table)))
meta$F.nurragi = rowSums(otudf)
otudf = as.data.frame(t(as.data.frame(physeq.a.waltergamsii@otu_table)))
meta$F.waltergamsii = rowSums(otudf)

# Plot a graph of the abundance of Fusarium for each sample grouped by Group:
p1 <- ggplot(subset(meta, Group %in% c("CL.BO","CL.YO",
                                       "CY.BU","CY.YO",
                                       "SC.HE","SC.SH",
                                       "SL.AN","SL.BE",
                                       "SL.SH")),
             aes(x = Group, y = Fusarium,  colour = interaction(Group))) +
  geom_point(alpha = 1, position = "jitter", size = 4) +
  geom_boxplot(alpha = 0, colour = "black", size = 0.8)+
  theme_classic() +   scale_color_brewer(palette = "Paired") +
  labs(x = "", y = "\n Fusarium") +
  theme(text = element_text(size=15, colour = "black"),
        axis.ticks = element_line(colour = "black", size = 1.25),
        axis.line = element_line(colour = 'black', size = 1.25),
        axis.text.x = element_text(angle=45, hjust=1,
                                   colour = "black", size = 13),
        axis.text.y = element_text(angle=0, hjust=0.5,
                                   colour = "black",size = 13),
        axis.title.y = element_text(color="black", size=12,face="bold"),
        legend.position = "none")

p2 <- ggplot(subset(meta, Group %in% c("CL.BO","CL.YO",
                                       "CY.BU","CY.YO",
                                       "SC.HE","SC.SH",
                                       "SL.AN","SL.BE",
                                       "SL.SH")),
             aes(x = Group, y = F.equiseti,  colour = interaction(Group))) +
  geom_point(alpha = 1, position = "jitter", size = 4) +
  geom_boxplot(alpha = 0, colour = "black", size = 0.8)+
  theme_classic() +   scale_color_brewer(palette = "Paired") +
  labs(x = "", y = "\n F.nurragi") +
  theme(text = element_text(size=15, colour = "black"),
        axis.ticks = element_line(colour = "black", size = 1.25),
        axis.line = element_line(colour = 'black', size = 1.25),
        axis.text.x = element_text(angle=45, hjust=1,
                                   colour = "black", size = 13),
        axis.text.y = element_text(angle=0, hjust=0.5,
                                   colour = "black",size = 13),
        axis.title.y = element_text(color="black", size=12,face="bold"),
        legend.position = "none")

p3 <- ggplot(subset(meta, Group %in% c("CL.BO","CL.YO",
```

```r
                                  "CY.BU","CY.YO",
                                  "SC.HE","SC.SH",
                                  "SL.AN","SL.BE",
                                  "SL.SH")),
             aes(x = Group, y = F.nurragi,  colour = interaction(Group))) +
  geom_point(alpha = 1, position = "jitter", size = 4) +
  geom_boxplot(alpha = 0, colour = "black", size = 0.8)+
  theme_classic() +   scale_color_brewer(palette = "Paired") +
  labs(x = "", y = "\n F.nurragi") +
  theme(text = element_text(size=15, colour = "black"),
        axis.ticks = element_line(colour = "black", size = 1.25),
        axis.line = element_line(colour = 'black', size = 1.25),
        axis.text.x = element_text(angle=45, hjust=1,
                                   colour = "black", size = 13),
        axis.text.y = element_text(angle=0, hjust=0.5,
                                   colour = "black",size = 13),
        axis.title.y = element_text(color="black", size=12,face="bold"),
        legend.position = "none")

p4 <- ggplot(subset(meta, Group %in% c("CL.BO","CL.YO",
                                  "CY.BU","CY.YO",
                                  "SC.HE","SC.SH",
                                  "SL.AN","SL.BE",
                                  "SL.SH")),
             aes(x = Group, y = F.waltergamsii,  colour = interaction(Group))) +
  geom_point(alpha = 1, position = "jitter", size = 4) +
  geom_boxplot(alpha = 0, colour = "black", size = 0.8)+
  theme_classic() +   scale_color_brewer(palette = "Paired") +
  labs(x = "", y = "\n F.waltergamsii") +
  theme(text = element_text(size=15, colour = "black"),
        axis.ticks = element_line(colour = "black", size = 1.25),
        axis.line = element_line(colour = 'black', size = 1.25),
        axis.text.x = element_text(angle=45, hjust=1,
                                   colour = "black", size = 13),
        axis.text.y = element_text(angle=0, hjust=0.5,
                                   colour = "black",size = 13),
        axis.title.y = element_text(color="black", size=12,face="bold"),
        legend.position = "none")

# Combine and Arrange the plots
fig <- ggarrange(p1, p2, p3, p4, labels = c("A", "B", "C", "D"), size = 8,
                 ncol = 2, nrow = 2)

# Add labels
fig <- annotate_figure(fig)

# Open a new PDF graphics device
# pdf(file = "Fig11_ITS.pdf", width=11,height=7.5)
# Print the figure
print(fig)


## $`1`
```
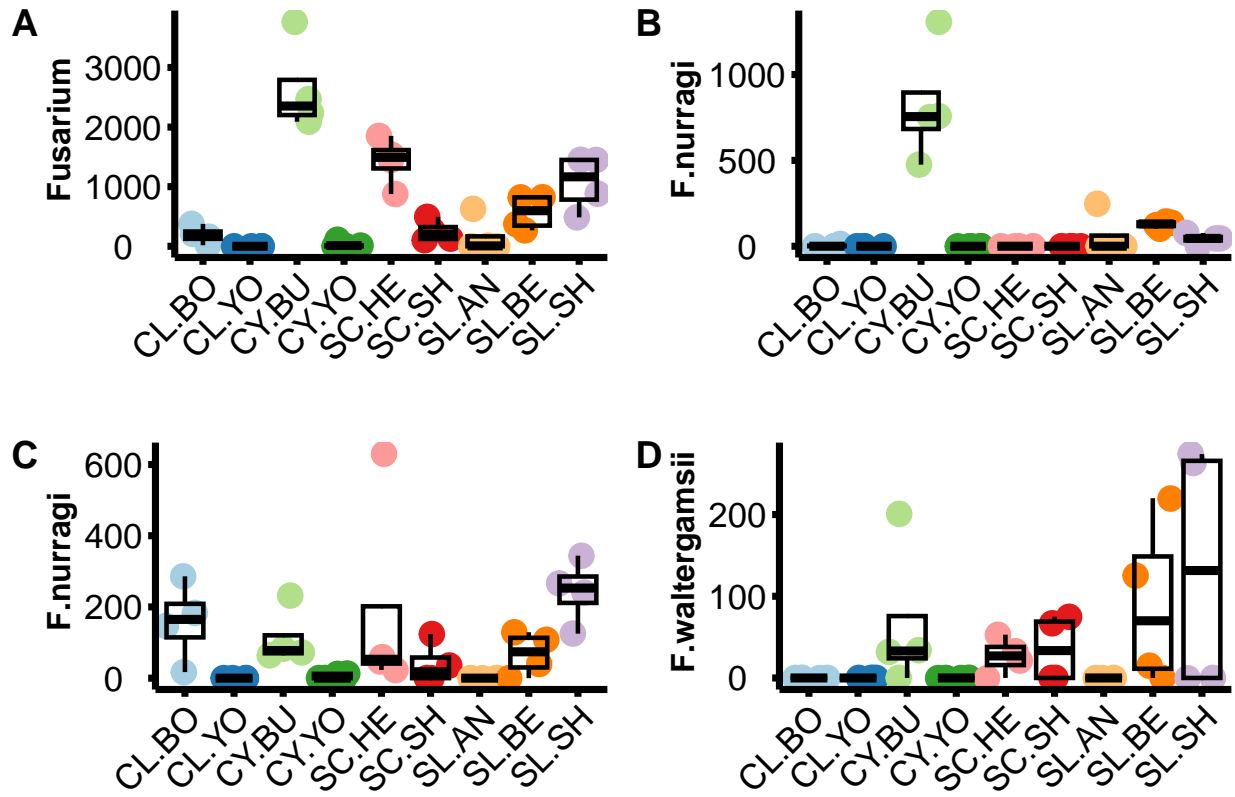
```
## 
## $'2'
```

A

```
## 
## attr(,"class")
## [1] "list"        "ggarrange"
```

```
# Close the PDF device and save the plot to a file
# dev.off()
```

```
sessionInfo()
```

```
## R version 4.3.2 (2023-10-31 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 11 x64 (build 22631)
## 
## Matrix products: default
## 
## 
## locale:
## [1] LC_COLLATE=English_United Kingdom.utf8
## [2] LC_CTYPE=English_United Kingdom.utf8
## [3] LC_MONETARY=English_United Kingdom.utf8
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United Kingdom.utf8
## 
## time zone: Europe/London
## tzcode source: internal
```

```
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] scales_1.3.0       ggtree_3.8.2       microbiome_1.22.0  UpSetR_1.4.0
##  [5] reshape2_1.4.4     plyr_1.8.9         RColorBrewer_1.1-3 ggpubr_0.6.0
##  [9] lubridate_1.9.3    forcats_1.0.0      stringr_1.5.0      dplyr_1.1.3
## [13] purrr_1.0.2        readr_2.1.4        tidyr_1.3.0        tibble_3.2.1
## [17] ggplot2_3.4.4      tidyverse_2.0.0    phyloseq_1.44.0    qiime2R_0.99.6
##
## loaded via a namespace (and not attached):
##   [1] bitops_1.0-7          gridExtra_2.3          permute_0.9-7
##   [4] rlang_1.1.1           magrittr_2.0.3         ade4_1.7-22
##   [7] compiler_4.3.2        mgcv_1.9-0             vctrs_0.6.3
##  [10] pkgconfig_2.0.3       crayon_1.5.2          fastmap_1.1.1
##  [13] backports_1.4.1       XVector_0.40.0        labeling_0.4.3
##  [16] utf8_1.2.3            rmarkdown_2.25        tzdb_0.4.0
##  [19] xfun_0.40             cachem_1.0.8          zlibbioc_1.46.0
##  [22] aplot_0.2.2           GenomeInfoDb_1.36.4   jsonlite_1.8.7
##  [25] biomformat_1.28.0     rhdf5filters_1.12.1   Rhdf5lib_1.22.1
##  [28] broom_1.0.5           parallel_4.3.2        cluster_2.1.4
##  [31] R6_2.5.1              stringi_1.7.12        car_3.1-2
##  [34] zCompositions_1.4.1   rpart_4.1.21          Rcpp_1.0.11
##  [37] iterators_1.0.14      knitr_1.44            base64enc_0.1-3
##  [40] IRanges_2.34.1        Matrix_1.6-1.1        splines_4.3.2
##  [43] nnet_7.3-19           igraph_1.5.1          timechange_0.2.0
##  [46] tidyselect_1.2.0      abind_1.4-5           rstudioapi_0.15.0
##  [49] yaml_2.3.7            vegan_2.6-4           codetools_0.2-19
##  [52] lattice_0.21-9        treeio_1.24.3         Biobase_2.60.0
##  [55] withr_2.5.2           Rtsne_0.16            evaluate_0.22
##  [58] foreign_0.8-85        gridGraphics_0.5-1    survival_3.5-7
##  [61] Biostrings_2.68.1     pillar_1.9.0          carData_3.0-5
##  [64] checkmate_2.2.0       DT_0.30               foreach_1.5.2
##  [67] stats4_4.3.2          NADA_1.6-1.1          ggfun_0.1.3
##  [70] generics_0.1.3        RCurl_1.98-1.12       truncnorm_1.0-9
##  [73] S4Vectors_0.38.2      hms_1.1.3             tidytree_0.4.5
##  [76] munsell_0.5.0         glue_1.6.2            lazyeval_0.2.2
##  [79] Hmisc_5.1-1           tools_4.3.2           data.table_1.14.8
##  [82] ggsignif_0.6.4        fs_1.6.3              cowplot_1.1.1
##  [85] rhdf5_2.44.0          grid_4.3.2            ape_5.7-1
##  [88] colorspace_2.1-0      patchwork_1.1.3       nlme_3.1-163
##  [91] GenomeInfoDbData_1.2.10 htmlTable_2.4.1     Formula_1.2-5
##  [94] cli_3.6.1             fansi_1.0.4           gtable_0.3.4
##  [97] yulab.utils_0.1.0     rstatix_0.7.2         digest_0.6.33
## [100] BiocGenerics_0.46.0   ggplotify_0.1.2       farver_2.1.1
## [103] htmlwidgets_1.6.2     memoise_2.0.1         htmltools_0.5.6
## [106] multtest_2.56.0       lifecycle_1.0.4       MASS_7.3-60
```