Metagenomics\_analysis

Jasmine Speranza

The analyses were run on two sample batches. The first batch consist in the C4U and H4U samples, the second one is represented by the samples collected during the FMT experiment which are H3U, H3T, H4T, H7A, H7C, C7A and C7C. These decisions was made basing on the different collection time.

Download and recall all the packages needed for the analysis

#Generic packages to install for all the analysis  
BiocManager::install("phyloseq")  
BiocManager::install("microbiome")  
install.packages("multcompView")  
library(microbiome)  
library(phyloseq)  
library(jsonlite)  
library(colorspace)  
  
#Packages to import your dataset on R  
library(readxl)  
library(dplyr)  
library(tibble)  
  
#Packages to decontaminate your dataset  
library(ggplot2)  
library(decontam)  
  
#Packages for DESeq study  
library(DESeq2)  
library(ggplot2)  
library(RColorBrewer)  
  
#Packages to normalize samples   
library(ggplot2)  
library(vegan)  
library(randomcoloR)  
library(RColorBrewer)  
  
#Packages to assess Absolute Abundance, Relative abundance, and plot Venn diagrams  
library(tidyverse)  
library(RColorBrewer)  
library(ggvenn)  
library(vegan)  
  
#Packages to assess the 10 most abundant phyla-families  
library(dplyr)  
library(ggplot2)  
library(RColorBrewer)  
  
#Packages for Alpha diversity analysis  
library(ggplot2)  
library(dplyr)  
library(tidyr)  
library(ggsignif)   
library(multcomp)   
library(stats)   
library(car)   
library(ggplot2)   
library(emmeans)   
library(multcompView)   
library(gridExtra)   
library(cowplot)   
  
#Packages for Beta diversity analysis  
library(dplyr)  
library(tidyr)  
library(ggplot2)  
library(ggrepel)  
library(cowplot)  
library(vegan)  
library(permute)  
library(lattice)

Import your data tables and create the phyloseq object (1st sample batch C4U vs H4U, 2nd sample batch FMT experiment samples)

#Import your tables with single samples per conditions, OTUs and Taxonomy  
otu\_mat<- read\_excel("/Users/u2093090/Desktop/Metagenomics analysis/R analysis /JS\_allFMTsamples & qPCR data.xls", sheet = "OUT\_checked")  
tax\_mat<- read\_excel("/Users/u2093090/Desktop/Metagenomics analysis/R analysis /JS\_allFMTsamples & qPCR data.xls", sheet = "Tax checked")  
samples\_df <- read\_excel("/Users/u2093090/Desktop/Metagenomics analysis/R analysis /JS\_allFMTsamples & qPCR data.xls", sheet = "Samples")  
  
#Define the row names from the otu column  
otu\_mat <- otu\_mat %>%  
 tibble::column\_to\_rownames("OTU\_ID")   
  
#The same for the two other matrices  
tax\_mat <- tax\_mat %>%   
 tibble::column\_to\_rownames("OTU\_ID")  
  
#Sample conditions  
samples\_df <- samples\_df %>%   
 tibble::column\_to\_rownames("Sample")   
  
#Transform otu and tax tables into matrices (sample table can be left as data frame)  
otu\_mat <- as.matrix(otu\_mat)  
tax\_mat <- as.matrix(tax\_mat)  
otu\_mat  
tax\_mat  
  
#Transform to phyloseq objects  
OTU = otu\_table(otu\_mat, taxa\_are\_rows = TRUE)  
TAX = tax\_table(tax\_mat)  
samples = sample\_data(samples\_df)  
  
#look at the tables  
otu\_mat  
tax\_mat  
samples  
  
#Create the phyloseq object  
phylo\_allsamples <- phyloseq(OTU, TAX, samples)  
phylo\_allsamples  
  
#To delete the initial part of taxa names (p\_\_, o\_\_, g\_\_)in the table  
phylo\_allsamples@tax\_table <- substring(phylo\_allsamples@tax\_table, 4)  
colnames(phylo\_allsamples@tax\_table) <- c("Kingdom",  
 "Phylum",  
 "Class",  
 "Order",  
 "Family",  
 "Genus",  
 "Species")  
View(phylo\_allsamples@tax\_table)

After the creation of phyloseq object, eliminate the contaminates from your data and create a new phyloseq object

head((sample\_data(phylo\_allsamples)))  
  
#Inspect library size  
df <- as.data.frame(sample\_data(phylo\_allsamples)) # Put sample\_data into a ggplot-friendly data.frame  
df$LibrarySize <- sample\_sums(phylo\_allsamples)  
df <- df[order(df$LibrarySize),]  
df$Index <- seq(nrow(df))  
ggplot(data=df, aes(x=Index, y=LibrarySize)) + geom\_point()  
  
#Find the contamination using the qPCR data plus the frequency method  
contamdf.freq <- isContaminant(phylo\_allsamples, method="frequency", conc="qPCR")  
head(contamdf.freq)  
view(contamdf.freq)   
table(contamdf.freq$contaminant)  
head(which(contamdf.freq$contaminant))  
  
#Take a look at the OTU detected as contaminates to understand if they are actually contamination  
plot\_frequency(phylo\_allsamples, taxa\_names(phylo\_allsamples)[c(62,141)], conc="qPCR") +   
 xlab("DNA Concentration (qPCR data)")  
  
#In this plot the dashed black line shows the model of a non-contaminant OTU feature for which frequency is expected to be independent of the input DNA concentration.   
#The red line shows the model of a contaminant sequence feature, for which frequency is expected to be inversely proportional to input DNA concentration, as contaminating DNA will make up a larger fraction of the total DNA in samples with very little total DNA. Clearly OTU 141 fits the red contamination model very well.  
  
#Let see if using another method we detect other contamination  
contamdf.freq <- isContaminant(phylo\_allsamples, method="auto", conc="qPCR")  
head(contamdf.freq)  
view(contamdf.freq)   
table(contamdf.freq$contaminant)  
head(which(contamdf.freq$contaminant))  
#I do not have a negative control so I can just try the method auto, and it gives to me the same output of the frequency method  
  
#Now that we have identified likely contaminants, let’s remove them from the phyloseq object:  
phylo\_allsamples  
phylo.decont <- prune\_taxa(!contamdf.freq$contaminant, phylo\_allsamples)  
phylo.decont

Run the DESeq analysis on the decontaminated phyloseq object

#my phyloseq object  
phylo.decont  
  
#Change the groups in the code to look at the comparison you are interested in  
phylo\_subset <- subset\_samples(phylo.decont, Group %in% c("H3U", "H7C"))  
  
#convert the phyloseq-format data into a DESeq data set basing on the group  
#The default multiple-inference correction is Benjamini-Hochberg,   
phylo.Deseq = phyloseq\_to\_deseq2(phylo\_subset, ~ Group)  
phylo.Deseq = DESeq(phylo.Deseq, test="Wald", fitType="parametric")  
  
#results visualization basing on the adjusted p-value  
res = results(phylo.Deseq, cooksCutoff = FALSE)  
alpha = 0.05  
sigtab = res[which(res$padj < alpha), ]  
sigtab = cbind(as(sigtab, "data.frame"), as(tax\_table(phylo\_subset)[rownames(sigtab), ], "matrix"))  
head(sigtab)  
view(sigtab)  
  
#graph to visualize the results  
theme\_set(theme\_bw())  
scale\_fill\_discrete <- function(palname = "Set1", ...) {  
 scale\_fill\_brewer(palette = palname, ...)  
}  
# Phylum order  
x = tapply(sigtab$log2FoldChange, sigtab$Phylum, function(x) max(x))  
x = sort(x, TRUE)  
sigtab$Phylum = factor(as.character(sigtab$Phylum), levels=names(x))  
# Genus order  
x = tapply(sigtab$log2FoldChange, sigtab$Genus, function(x) max(x))  
x = sort(x, TRUE)  
sigtab$Genus = factor(as.character(sigtab$Genus), levels=names(x))  
  
#Plot  
ggplot(sigtab, aes(x=Family, y=log2FoldChange, color=Phylum)) +   
 geom\_point(size=6) +   
 ggtitle("H3U vs H7C") +  
 theme(axis.text.x = element\_text(angle = -45, hjust = 0, vjust= 1))

Normalize the samples using the rarefaction curves

#look at the data  
sample\_names(phylo.decont)  
rank\_names(phylo.decont)  
sample\_variables(phylo.decont)  
  
#Generate a custom color palette with 92 colors  
color\_palette <- distinctColorPalette(92)  
  
#Look at the sample data through bar plot related to the taxa abundance of each sample (replicate)  
plot\_bar(phylo.decont)  
plot\_bar(phylo.decont, fill ="Phylum")   
  
### Things to consider:   
## 1. We need to exclude the OTUs with no or low abundant reads

#let's look at the total sum of sequences of each OTU  
taxa\_sums(phylo.decont)  
  
#remove OTUs with <1, <2 or <5 reads and create new database  
phylo.decont\_1 = filter\_taxa(phylo.decont, function (x) sum (x) >1, TRUE)  
phylo.decont\_2 = filter\_taxa(phylo.decont, function (x) sum (x) >2, TRUE)  
phylo.decont\_5 = filter\_taxa(phylo.decont, function (x) sum (x) >5, TRUE)  
  
#Look at the results  
phylo.decont   
phylo.decont\_1 **#I'm choosing this condition**  
phylo.decont\_2   
phylo.decont\_5   
  
## 2.Consider the rarefaction level - this will be key for diversity analysis  
  
# Total reads for each samples and creation of rarefactions curve to assess the sampling  
sort(sample\_sums(phylo.decont\_1))  
  
#Plot raref. curves through the rarecurve applied to the (otu\_table(phylo.decont\_1)).The t function traspone the OTU table into the format needed for the curve creation. The parameter 'step=50' and cex=0.5 are related to the curve creation   
rarecurve(data.frame(t(otu\_table(phylo.decont\_1))), step = 50, cex=0.5)  
  
#Re-sampling and exclusion of samples that have less than a specific number of reads. This reduces also the OTUs number of the other samples considered for the analysis.  
phylo.decont\_rarelow<-rarefy\_even\_depth(phylo.decont\_1, min(sample\_sums(phylo.decont\_1)), rngseed = 9237)  
phylo.decont\_rarehigh<-rarefy\_even\_depth(phylo.decont\_1,4600, rngseed = 9237)  
  
#Threshold used: 18000 for the 1st sample batch, 4600 for the 2nd sample batch.  
#Samples removed because they contained fewer reads than `sample.size`: for 1st sample batch 2 samples (C4U2, H4U2), for the 2nd sample batch 3 samples (C7C2,H3T4,H7A2)  
  
# Generate a color palette with 92 colors  
color\_palette <- colorRampPalette(brewer.pal(12, "Paired"))(92)

#plot data  
plot\_bar(phylo.decont\_rarehigh, fill= "Phylum") + scale\_fill\_manual(values = color\_palette)  
plot\_bar(phylo.decont\_rarelow, fill="Phylum") + scale\_fill\_manual(values = color\_palette)

Plot: Absolute Abundance, Relative abundance, Venn diagrams

#Change phyloseq object in dataframe  
phylo\_meta\_df <- psmelt(phylo.decont\_rarehigh)   
phylo.decont\_rarehigh  
number\_of\_taxa <- phylo\_meta\_df %>%   
 filter(Abundance > 0) %>%   
 group\_by(Group, Family) %>%   
 summarise(n = length(Abundance))  
  
view(number\_of\_taxa)  
  
# Determine the number of unique Phyla  
num\_phyla <- number\_of\_taxa %>%   
 pull(Family) %>%   
 unique() %>%   
 length()  
  
  
###VEEN DIAGRAM  
#Check phylum present in specific samples  
unique(number\_of\_taxa$Phylum[number\_of\_taxa$Group== "H7A"])  
unique(number\_of\_taxa$Phylum[number\_of\_taxa$Group== "H7C"])  
  
#Place the two sets of phyla in a list:  
venn\_data <- list(H7A = unique(number\_of\_taxa$Phylum[number\_of\_taxa$Group== "H7A"]),  
 H7C = unique(number\_of\_taxa$Phylum[number\_of\_taxa$Group== "H7C"]))  
  
#Plot the graph  
ggvenn(venn\_data, fill\_color = c('#CC66FF', '#F564E3' ))  
  
#Palette colour: c("C7A" = "#CA5E00", "C7C" = "#F0E442", 'H3U' ='#00BFC4', 'H3T'= '#00BA38', 'H4T'= '#56B4E9', 'H7A'='#CC66FF', 'H7C'= '#F564E3')  
  
#Check unique phylum between the 2 samples   
venn\_data$H7A[!venn\_data$H7A %in% venn\_data$H7C]   
venn\_data$H7C[!venn\_data$H7C %in% venn\_data$H7A]   
  
  
#Set up the color palette  
base\_palette <- brewer.pal(12, "Paired")  
color\_palette <- colorRampPalette(base\_palette)(num\_phyla)  
  
  
### ABSOLUTE ABUNDANCE  
abundance\_of\_taxa <- phylo\_meta\_df %>%   
 filter(Abundance > 0) %>%   
 group\_by(Group, Family) %>%   
 summarise(Abundance = sum(Abundance))  
  
# Plot it  
abundance\_of\_taxa %>%   
 ggplot(aes(x = Group, y = Abundance, fill = Family)) +  
 geom\_col(position = "stack", color = "black") +  
 scale\_fill\_manual(values = color\_palette) +  
 theme\_minimal() +  
 labs(title = "Absolute Abundance of Taxa", x = "Group", y = "Abundance")  
  
  
### RELATIVE ABUNDANCE  
abundance\_of\_taxa <- abundance\_of\_taxa %>%   
 group\_by(Group) %>%   
 mutate(relative = Abundance/sum(Abundance) \* 100)  
  
#plot it  
abundance\_of\_taxa %>%   
 ggplot(aes(x = Group, y = relative, fill = Family)) + scale\_fill\_manual(values = color\_palette) +  
 geom\_col(position = "stack") + geom\_col(position = "stack", color = "black") +  
 labs(title = "Relative Abundance of Taxa", x = "Group", y = "Abundance")  
  
###Heatmap  
plot\_heatmap(phylo.decont\_rarehigh)  
plot\_heatmap(phylo.decont\_rarelow, taxa.label="Phylum")  
plot\_heatmap(phylo.decont\_rarehigh, taxa.label="Phylum")

10 most Abundant Phyla or Families

# Calculate the absolute abundance for each phylum or family within each group  
abundance\_of\_taxa <- phylo\_meta\_df %>%   
 filter(Abundance > 0) %>%   
 group\_by(Group,Family) %>%   
 summarise(Abundance = sum(Abundance), .groups = 'drop')  
  
# Calculate total abundance for each phylum across all groups  
total\_abundance\_of\_phyla <- abundance\_of\_taxa %>%   
 group\_by(Family) %>%   
 summarise(TotalAbundance = sum(Abundance), .groups = 'drop')  
  
# Identify the top 10 most abundant phyla  
top\_10\_phyla <- total\_abundance\_of\_phyla %>%   
 arrange(desc(TotalAbundance)) %>%   
 slice\_head(n = 10) %>%   
 pull(Family)  
  
# Filter the dataset to include only the top 10 phyla  
top\_10\_abundance\_of\_taxa <- abundance\_of\_taxa %>%   
 filter(Family %in% top\_10\_phyla)  
  
# Determine the number of unique Phyla (which will be 10)  
num\_phyla <- top\_10\_abundance\_of\_taxa %>%   
 pull(Family) %>%   
 unique() %>%   
 length()  
  
# Set up the color palette  
base\_palette <- brewer.pal(12, "Paired")  
color\_palette <- colorRampPalette(base\_palette)(num\_phyla)  
  
  
# Plot absolute abundance  
top\_10\_abundance\_of\_taxa %>%   
 ggplot(aes(x = Group, y = Abundance, fill = Family)) +  
 geom\_col(position = "stack") +  
 geom\_col(position = "stack", color = "black") +  
 scale\_fill\_manual(values = color\_palette) +  
 theme\_minimal() +  
 labs(title = "Absolute Abundance of Top 10 Families", x = "Group", y = "Abundance")  
  
# Calculate relative abundance within each group  
relative\_abundance\_of\_taxa <- top\_10\_abundance\_of\_taxa %>%   
 group\_by(Group) %>%   
 mutate(relative = Abundance / sum(Abundance) \* 100) %>%   
 ungroup()  
  
# Plot relative abundance  
relative\_abundance\_of\_taxa %>%   
 ggplot(aes(x = Group, y = relative, fill = Family)) +  
 geom\_col(position = "stack") +  
 scale\_fill\_manual(values = color\_palette) +  
 theme\_minimal() +  
 labs(title = "Relative Abundance of Top 10 Families", x = "Group", y = "Relative Abundance (%)")

Alpha diversity analysis and Wilcox signed-rank test, specifically for the 1st sample batch (C4U vs H4U)

#all the alpha diversity parameters per each samples  
plot\_richness(phylo.decont\_rarehigh)  
#Focus on some of them  
plot\_richness(phylo.decont\_rarehigh, measures= c('Observed','Shannon','Simpson'))  
  
#Plot using a variable. In my analysis I have used the group one and I have added the color  
plot\_richness(phylo.decont\_rarehigh, x='Group', color = 'Group', measures= 'Observed')  
#adding some features  
plot\_richness(phylo.decont\_rarehigh,x='Group', color = 'Group', measures= 'Observed') + geom\_jitter(size=3, alpha=0.7)   
  
##Boxplot  
#Generate a data.frame with adiv measures  
adiv <- data.frame(  
 "Observed" = phyloseq::estimate\_richness(phylo.decont\_rarehigh, measures = "Observed"),  
 "Shannon" = phyloseq::estimate\_richness(phylo.decont\_rarehigh, measures = "Shannon"),  
 "Simpson" = phyloseq::estimate\_richness(phylo.decont\_rarehigh, measures = 'Simpson'),  
 #"Chao1"= phyloseq::estimate\_richness(phylo.decont\_rarehigh,measures = 'Chao1'),  
 #"Fisher"= phyloseq::estimate\_richness(phylo.decont\_rarehigh,measures = 'Fisher'),  
 #"InvSimpson"= phyloseq::estimate\_richness(phylo.decont\_rarehigh,measures = "InvSimpson"),  
 'Status' = phyloseq::sample\_data(phylo.decont\_rarehigh)$Group)  
head(adiv)   
adiv

#Summarize  
adiv%>%  
 group\_by(Status) %>%  
 dplyr::summarise(median\_observed = median(Observed),  
 median\_shannon = median(Shannon),  
 #median\_Chao1 = median(Chao1.Chao1),  
 median\_Simpson = median(Simpson)  
 #median\_InvSimpson = median(InvSimpson)  
 #median\_Fisher = median(Fisher)  
 )

#Stats between specific groups  
adiv\_subset <- subset(adiv, Status %in% c("C7A", "H4U"))  
adiv\_subset  
#Summarize  
adiv\_subset%>%  
 group\_by(Status) %>%  
 dplyr::summarise(median\_observed = median(Observed),  
 median\_shannon = median(Shannon),  
 #median\_Chao1 = median(Chao1.Chao1),  
 median\_Simpson = median(Simpson)  
 #median\_InvSimpson = median(InvSimpson)  
 #median\_Fisher = median(Fisher)  
 )  
  
#Test  
wilcox.test(Observed ~ Status, data = adiv\_subset, exact = FALSE, conf.int = TRUE)  
#wilcox.test(Fisher ~ Status, data = adiv\_subset, exact = FALSE, conf.int = TRUE)  
wilcox.test(Shannon ~ Status, data = adiv\_subset, exact = FALSE, conf.int = TRUE)  
#wilcox.test(Chao1.Chao1 ~ Status, data = adiv\_subset, exact = FALSE, conf.int = TRUE)  
wilcox.test(Simpson ~ Status, data = adiv\_subset, exact = FALSE, conf.int = TRUE)  
#wilcox.test(InvSimpson ~ Status, data = adiv\_subset, exact = FALSE, conf.int = TRUE)  
  
  
#Plot adiv measures  
adiv %>%  
 gather(key = metric, value = value, c("Observed", "Shannon", "Simpson")) %>%  
 mutate(metric = factor(metric, levels = c("Observed", "Shannon", "Simpson"))) %>%  
 ggplot(aes(x = Status, y = value, color = Status)) +  
 geom\_boxplot(outlier.color = NA, color = "black") +  
 geom\_jitter(aes(color = Status), height = 0, width = 0.2) +  
 scale\_color\_manual(values = c('C4U' = '#F8766D', 'H4U' = '#619CFF')) +  
 labs(x = "", y = "", title = "Alpha Diversity Analysis") +  
 facet\_wrap(~ metric, scales = "free") +  
 theme\_classic() +  
 theme(  
 legend.title = element\_text(face = "bold"),  
 legend.position = "right",  
 panel.grid.major = element\_line(color = "grey90"),  
 panel.grid.minor = element\_line(color = "grey95"),  
 strip.background = element\_blank(),  
 strip.text = element\_text(face = "bold"),   
 plot.title = element\_text(face = "bold", hjust = 0.5)   
 )   
  
#Test are not significant for the pair C4U vs H4U  
 # geom\_signif(  
 # comparisons = list(c('C4U', 'H4U')),  
 #map\_signif\_level = FALSE,  
 #test = "wilcox.test",  
 #y\_position = c(max(adiv$Shannon, na.rm = TRUE)) + 0.1,  
 #step\_increase = 0.1,  
 #annotations = c("", "", "") # Remove annotations  
 #)

Alpha diversity for the 2nd sample batch (FMT samples): Check normality distribution and homogeneity to be sure about ANOVA test. Run the ANOVA test and the Tukey post-hoc test then create the plots

#Code to verify which statistical test is better to use  
# Function to check assumptions and decide on ANOVA or Kruskal-Wallis  
check\_anova\_assumptions <- function(data, metric, group\_var) {  
 formula <- as.formula(paste(metric, "~", group\_var))  
   
 # Check normality  
 shapiro\_test <- shapiro.test(residuals(aov(formula, data = data)))  
   
 # Check homogeneity of variances  
 levene\_test <- leveneTest(formula, data = data)  
   
 # Collect p-values  
 shapiro\_p <- shapiro\_test$p.value  
 levene\_p <- levene\_test[1, "Pr(>F)"]  
   
 # Print results  
 cat("Shapiro-Wilk p-value for", metric, ":", shapiro\_p, "\n")  
 cat("Levene's Test p-value for", metric, ":", levene\_p, "\n")  
   
 # Decide on ANOVA or Kruskal-Wallis  
 if (shapiro\_p > 0.05 & levene\_p > 0.05) {  
 cat("Both assumptions met for", metric, "- Running ANOVA.\n")  
 anova\_result <- summary(aov(formula, data = data))  
 return(anova\_result)  
 } else {  
 cat("Assumptions not met for", metric, "- Running Kruskal-Wallis.\n")  
 kruskal\_result <- kruskal.test(formula, data = data)  
 return(kruskal\_result)  
 }  
}  
  
# Apply function to each metric  
metrics <- c("Observed", "Shannon", "Simpson")  
results <- list()  
  
for (metric in metrics) {  
 result <- check\_anova\_assumptions(adiv, metric, "Status")  
 results[[metric]] <- result  
}  
  
# Print results  
print(results)  
  
# Perform post-hoc Tukey's HSD test if ANOVA is significant  
posthoc\_results <- list()  
  
for (metric in metrics) {  
 if ("Pr(>F)" %in% colnames(results[[metric]][[1]]) && results[[metric]][[1]]$`Pr(>F)`[1] < 0.05) {  
 aov\_model <- aov(as.formula(paste(metric, "~Status")), data = adiv)  
 posthoc\_result <- TukeyHSD(aov\_model)  
 posthoc\_results[[metric]] <- posthoc\_result  
 }  
}  
  
# Print post-hoc results  
print(posthoc\_results)  
  
  
###Plot the results  
# Metrics data  
data <- data.frame(  
 Observed = c(52, 57, 28, 18, 58, 63, 29, 14, 46, 31, 28, 125, 58, 61, 43, 39, 36, 23, 13, 20, 38, 15, 21, 12, 33, 22),  
 Shannon = c(0.32035013, 0.84174186, 0.28648566, 0.15103254, 0.53529748, 0.59427553, 0.18093271, 0.10374595, 0.37159864, 0.18806128, 0.24174285, 1.38774814, 1.50439239, 1.14960511, 1.18332217, 0.22997076, 0.30415012, 0.13414634, 0.19967506, 0.27730534, 0.99606938, 0.18745446, 0.50663295, 0.05988349, 0.24966046, 0.14781359),  
 Simpson = c(0.08813440, 0.31353119, 0.10245520, 0.04457930, 0.15877420, 0.18419480, 0.04928488, 0.02929291, 0.11212127, 0.05015709, 0.08065624, 0.44159764, 0.57460586, 0.44675312, 0.48417108, 0.06111522, 0.09130293, 0.03697486, 0.07926295, 0.10867372, 0.41532873, 0.07080113, 0.28653837, 0.01600095, 0.06989830, 0.04370132),  
 Status = factor(c("C7A", "C7A", "C7A", "C7A", "C7C", "C7C", "C7C", "H3U", "H3U", "H3U", "H3U", "H3T", "H3T", "H3T", "H3T", "H4T", "H4T", "H4T", "H4T", "H7A", "H7A", "H7A", "H7A", "H7C", "H7C", "H7C"))  
)  
  
# Function to compute ANOVA e Tukey  
calculate\_anova\_tukey <- function(data, metric) {  
 formula <- as.formula(paste(metric, "~ Status"))  
 fit <- aov(formula, data = data)  
 anova\_res <- summary(fit)  
 tukey\_res <- TukeyHSD(fit)  
 list(anova = anova\_res, tukey = tukey\_res)  
}  
  
# Function to determinate if values are significant  
get\_significance <- function(p) {  
 if (p < 0.001) {  
 return("\*\*\*")  
 } else if (p < 0.01) {  
 return("\*\*")  
 } else if (p < 0.05) {  
 return("\*")  
 } else if (p < 0.1) {  
 return(".")  
 } else {  
 return("")  
 }  
}  
  
# Plot with ANOVA and Tukey results  
create\_plot <- function(data, metric, anova\_res, tukey\_res, colors) {  
 # Convert metric to a chr for ggplot  
 data$Metric <- data[[metric]]  
   
 # Create your plot  
 p <- ggplot(data, aes(x = Status, y = Metric)) +  
 geom\_boxplot(outlier.shape = NA, fill = "white") +  
 geom\_jitter(width = 0.2, aes(color = Status), size = 2) +  
 scale\_color\_manual(values = colors) +  
 labs(title = metric, y = metric) +  
 theme\_minimal() +  
 theme(  
 legend.position = "bottom",  
 legend.box = "horizontal",  
 plot.title = element\_text(hjust = 0.5, size = 16)  
 )  
   
 # Tukey  
 tukey\_results <- as.data.frame(tukey\_res$Status)  
 significant\_pairs <- subset(tukey\_results, `p adj` < 0.05)  
   
 y\_max <- max(data$Metric, na.rm = TRUE)  
 y\_offset <- 0  
   
 for (i in 1:nrow(significant\_pairs)) {  
 comparison <- rownames(significant\_pairs)[i]  
 groups <- unlist(strsplit(comparison, "-"))  
 group1 <- groups[1]  
 group2 <- groups[2]  
 p\_val <- significant\_pairs$`p adj`[i]  
 significance <- get\_significance(p\_val)  
   
 y <- y\_max + y\_offset  
 x1 <- which(levels(data$Status) == group1)  
 x2 <- which(levels(data$Status) == group2)  
   
 p <- p + annotate("segment", x = x1, xend = x2, y = y, yend = y, color = "black") +  
 annotate("text", x = mean(c(x1, x2)), y = y, label = significance, vjust = -0.1, size = 6, color = "black")  
   
 y\_offset <- y\_offset + 0.05 \* y\_max  
 }  
  
 return(p)  
}  
  
# Group colours  
colors <- c("C7A" = "#CA5E00", "C7C" = "#F0E442", 'H3U' = '#00BFC4', 'H3T' = '#00BA38', 'H4T' = '#56B4E9', 'H7A' = '#CC66FF', 'H7C' = '#F564E3')  
  
# Compute ANOVA and Tukey for each metric and plot graph   
metrics <- c("Observed", "Shannon", "Simpson")  
plots <- list()  
for (metric in metrics) {  
 res <- calculate\_anova\_tukey(data, metric)  
 plots[[metric]] <- create\_plot(data, metric, res$anova, res$tukey, colors)  
}  
  
# Take out the legend from one graph   
legend <- get\_legend(plots[[1]])  
  
# or each graph  
for (i in seq\_along(plots)) {  
 plots[[i]] <- plots[[i]] + theme(legend.position = "none")  
}  
  
# All graph together  
combined\_plot <- plot\_grid(  
 plot\_grid(plots[[1]], plots[[2]], plots[[3]], ncol = 3, align = "v"),  
 legend,  
 ncol = 1,  
 rel\_heights = c(1, 0.1)  
)  
  
#Graph with labels  
combined\_plot <- combined\_plot +   
 draw\_label("Alpha Diversity Analysis", x = 0.5, y = 1.05, vjust = 2, hjust = 0.3, size = 15) +  
 draw\_label("Status", x = 0.5, y = -0.02, vjust = 1, hjust = 0.5, size = 15)  
  
# Print graph  
print(combined\_plot)

Beta diversity analysis considering all the sample set basing on the group variable

# Define custom colors for groups  
custom\_colors <- c('C4U'= '#F8766D', 'H4U'= '#619CFF', "C7A" = "#CA5E00", "C7C" = "#F0E442", 'H3U' = '#00BFC4', 'H3T' = '#00BA38', 'H4T' = '#56B4E9', 'H7A' = '#CC66FF', 'H7C' = '#F564E3')  
  
# Calculation of Bray-Curtis distance for all conditions  
dist\_all <- phyloseq::distance(phylo.decont\_rarehigh, method = "bray")  
  
# Plot NMDS for all conditions  
ordination\_all\_NMDS <- ordinate(phylo.decont\_rarehigh, method = 'NMDS', distance = dist\_all)  
plot\_all\_NMDS <- plot\_ordination(phylo.decont\_rarehigh, ordination\_all\_NMDS, color="Group") +   
 theme\_classic() +  
 theme(strip.background = element\_blank()) +  
 stat\_ellipse(type = "norm", linetype = 2) +  
 scale\_fill\_manual(values = custom\_colors) +   
 scale\_color\_manual(values = custom\_colors) +   
 labs(title = "NMDS method",  
 x = "NMDS1", y = "NMDS2")  
  
# PCoA for all conditions with variance calculation  
ordination\_all\_PCoA <- ordinate(phylo.decont\_rarehigh, method = 'PCoA', distance = dist\_all)  
eigenvalues <- ordination\_all\_PCoA$values$Eigenvalues  
variance\_explained <- eigenvalues / sum(eigenvalues)  
percent\_variance\_explained <- round(variance\_explained \* 100, 2)  
  
# Plot PCoA  
plot\_all\_PCoA <- plot\_ordination(phylo.decont\_rarehigh, ordination\_all\_PCoA, color="Group") +   
 theme\_classic() +  
 theme(strip.background = element\_blank()) +  
 stat\_ellipse(type = "norm", linetype = 2) +  
 scale\_fill\_manual(values = custom\_colors) +   
 scale\_color\_manual(values = custom\_colors) +   
 labs(title = "PCoA Method",  
 x = paste0("PCoA1 (", percent\_variance\_explained[1], "%)"),  
 y = paste0("PCoA2 (", percent\_variance\_explained[2], "%)"))  
  
# Perform PERMANOVA on the Bray-Curtis distance matrix  
metadata\_all <- data.frame(sample\_data(phylo.decont\_rarehigh))  
permanova\_all <- adonis2(dist\_all ~ Group, data = metadata\_all)  
permanova\_p\_value <- as.numeric(as.character(permanova\_all$`Pr(>F)`[1]))  
permanova\_r2 <- as.numeric(permanova\_all$R2[1])  
permanova\_label <- paste("PERMANOVA: R2 =", round(permanova\_r2, 2), "- p =", signif(permanova\_p\_value, 3))  
print(permanova\_all)  
  
# Check for homogeneity of group dispersions (BETADISPER) on the Bray-Curtis distance matrix  
betadisper\_all <- betadisper(dist\_all, metadata\_all$Group)  
anova\_betadisper\_all <- anova(betadisper\_all)  
betadisper\_p\_value <- as.numeric(as.character(anova\_betadisper\_all$`Pr(>F)`[1]))  
betadisper\_label <- paste("BETADISPER: p =", signif(betadisper\_p\_value, 3))  
print(anova\_betadisper\_all)  
  
# Add statistical annotations to the plots  
plot\_all\_NMDS <- plot\_all\_NMDS +  
 annotate("text", x = Inf, y = Inf, label = permanova\_label, hjust = 1.1, vjust = 2.1, size = 4) +  
 annotate("text", x = Inf, y = Inf, label = betadisper\_label, hjust = 1.1, vjust = 4.1, size = 4)  
  
plot\_all\_PCoA <- plot\_all\_PCoA +  
 annotate("text", x = Inf, y = Inf, label = permanova\_label, hjust = 1.1, vjust = 2.1, size = 4) +  
 annotate("text", x = Inf, y = Inf, label = betadisper\_label, hjust = 1.1, vjust = 4.1, size = 4)  
  
# Combine NMDS and PCoA plots  
combined\_plot <- plot\_grid(plot\_all\_NMDS, plot\_all\_PCoA, labels = c("A", "B"))  
  
# Add a title above the combined plot  
title <- ggdraw() + draw\_label("Beta-diversity analysis", fontface = 'bold', size = 18)  
  
# Combine title and plot  
final\_plot <- plot\_grid(title, combined\_plot, ncol = 1, rel\_heights = c(0.1, 1))  
  
# Display the final plot  
print(final\_plot)

Beta diversity analysis considering specific samples and using the Pairwise and Permutation Test

### Comparison among the conditions using the Pairwaise permutation test and adjusting the p value through BH:  
# Function to perform pairwise permutation test  
perform\_pairwise\_permutation <- function(groups, phylo.decont\_rarehigh, colors) {  
 phylo\_subset <- subset\_samples(phylo.decont\_rarehigh, Group %in% groups)  
 dist\_subset <- phyloseq::distance(phylo\_subset, method = "bray")  
   
 # PCoA  
 ord\_pcoa <- ordinate(phylo\_subset, method = 'PCoA', distance = dist\_subset)  
 eigenvalues <- ord\_pcoa$values$Eigenvalues  
 variance\_explained <- eigenvalues / sum(eigenvalues)  
 percent\_variance\_explained <- round(variance\_explained \* 100, 2)  
   
 plot\_pcoa <- plot\_ordination(phylo\_subset, ord\_pcoa, color="Group") +   
 theme\_classic() +  
 theme(strip.background = element\_blank()) +  
 geom\_point(size = 4) +  
 geom\_text\_repel(aes(label = sample\_names(phylo\_subset))) + # Add labels to points  
 scale\_color\_manual(values = colors) +   
 labs(title = paste("PCoA Analysis for", paste(groups, collapse = " vs ")),  
 x = paste0("PCoA1 (", percent\_variance\_explained[1], "%)"),  
 y = paste0("PCoA2 (", percent\_variance\_explained[2], "%)"))  
   
 # Permutation test  
 metadata\_subset <- data.frame(sample\_data(phylo\_subset))  
 perm\_test\_result <- adonis2(dist\_subset ~ Group, data = metadata\_subset, permutations = 999)  
 p\_value <- as.numeric(as.character(perm\_test\_result$`Pr(>F)`[1]))  
   
 list(plot\_pcoa = plot\_pcoa, p\_value = p\_value)  
}  
  
# Function to perform pairwise comparisons using permutation test  
pairwise\_comparisons\_permutation <- function(groups, phylo.decont\_rarehigh, colors) {  
 comb <- combn(groups, 2)  
 results <- list()  
   
 for (i in 1:ncol(comb)) {  
 group\_pair <- comb[, i]  
 comparison <- perform\_pairwise\_permutation(group\_pair, phylo.decont\_rarehigh, colors)  
   
 results[[paste(group\_pair, collapse = " vs ")]] <- comparison  
 }  
   
 results  
}  
  
# List of groups  
groups <- unique(sample\_data(phylo.decont\_rarehigh)$Group)  
  
# Perform pairwise comparisons using permutation test  
results\_permutation <- pairwise\_comparisons\_permutation(groups, phylo.decont\_rarehigh, custom\_colors)  
  
# Extract p-values from results  
p\_values\_permutation <- sapply(results\_permutation, function(res) res$p\_value)  
  
# Apply multiple testing correction (Benjamini-Hochberg)  
p\_adjusted\_permutation <- p.adjust(p\_values\_permutation, method = "BH")  
  
# Display corrected p-values  
for (comparison in names(results\_permutation)) {  
 cat("\nComparison:", comparison)  
 cat("\nP-value:", results\_permutation[[comparison]]$p\_value)  
 cat("\nAdjusted P-value:", p\_adjusted\_permutation[comparison], "\n")  
 print(results\_permutation[[comparison]]$plot\_pcoa)  
}

Beta diversity analysis considering specific samples and using the Pairwise and Mann-Whitney Test:

### Comparison among the conditions using the Mann-Whitney U test and adjusting the p value through BH:  
# Function to perform Mann-Whitney U test  
perform\_mann\_whitney <- function(groups, phylo.decont\_rarehigh, colors) {  
 phylo\_subset <- subset\_samples(phylo.decont\_rarehigh, Group %in% groups)  
 dist\_subset <- phyloseq::distance(phylo\_subset, method = "bray")  
   
 # PCoA  
 ord\_pcoa <- ordinate(phylo\_subset, method = 'PCoA', distance = dist\_subset)  
 eigenvalues <- ord\_pcoa$values$Eigenvalues  
 variance\_explained <- eigenvalues / sum(eigenvalues)  
 percent\_variance\_explained <- round(variance\_explained \* 100, 2)  
   
 plot\_pcoa <- plot\_ordination(phylo\_subset, ord\_pcoa, color="Group") +   
 theme\_classic() +  
 theme(strip.background = element\_blank()) +  
 geom\_point(size = 4) +  
 geom\_text\_repel(aes(label = sample\_names(phylo\_subset))) +   
 scale\_color\_manual(values = colors) +   
 labs(title = paste("PCoA Analysis for", paste(groups, collapse = " vs ")),  
 x = paste0("PCoA1 (", percent\_variance\_explained[1], "%)"),  
 y = paste0("PCoA2 (", percent\_variance\_explained[2], "%)"))  
   
 # Mann-Whitney U test  
 metadata\_subset <- data.frame(sample\_data(phylo\_subset))  
 group1\_indices <- which(metadata\_subset$Group == groups[1])  
 group2\_indices <- which(metadata\_subset$Group == groups[2])  
   
 # Ensure there are enough observations in both groups  
 if (length(group1\_indices) < 2 || length(group2\_indices) < 2) {  
 p\_value <- NA # Not enough observations for the test  
 } else {  
 group1\_distances <- as.vector(as.dist(dist\_subset)[outer(group1\_indices, group1\_indices, function(x, y) x < y)])  
 group2\_distances <- as.vector(as.dist(dist\_subset)[outer(group2\_indices, group2\_indices, function(x, y) x < y)])  
 p\_value <- wilcox.test(group1\_distances, group2\_distances)$p.value  
 }  
   
 list(plot\_pcoa = plot\_pcoa, p\_value = p\_value)  
}  
  
# Function to perform pairwise comparisons  
pairwise\_comparisons\_mann\_whitney <- function(groups, phylo.decont\_rarehigh, colors) {  
 comb <- combn(groups, 2)  
 results <- list()  
   
 for (i in 1:ncol(comb)) {  
 group\_pair <- comb[, i]  
 comparison <- perform\_mann\_whitney(group\_pair, phylo.decont\_rarehigh, colors)  
   
 results[[paste(group\_pair, collapse = " vs ")]] <- comparison  
 }  
   
 results  
}  
  
# List of groups  
groups <- unique(sample\_data(phylo.decont\_rarehigh)$Group)  
  
# Perform pairwise comparisons using Mann-Whitney U test  
results\_mann\_whitney <- pairwise\_comparisons\_mann\_whitney(groups, phylo.decont\_rarehigh, custom\_colors)  
  
# Extract p-values from results  
p\_values\_mann\_whitney <- sapply(results\_mann\_whitney, function(res) res$p\_value)  
  
# Apply multiple testing correction (Benjamini-Hochberg)  
p\_adjusted\_mann\_whitney <- p.adjust(p\_values\_mann\_whitney, method = "BH")  
  
# Display results  
for (comparison in names(results\_mann\_whitney)) {  
 cat("\nComparison:", comparison)  
 cat("\nP-value:", results\_mann\_whitney[[comparison]]$p\_value)  
 cat("\nAdjusted P-value:", p\_adjusted\_mann\_whitney[comparison], "\n")  
 print(results\_mann\_whitney[[comparison]]$plot\_pcoa)  
}