# Corn Bran Particle Size Modulates the Gut Microbiome Profile and Composition: Results from an *In Vitro* Fermentation Study

## Supplementary Material 2

## R code

### Rarefaction Curve

```
#Set working directory appropriately
setwd("~/Desktop/ANSC_project")

library(readr)
library(stringr)
library(ggplot2)
library(tidyr)
library(dplyr)

#Rarefaction curve
rm(V4rfact)

#Read in metadata
metadata <- "data/size_time.metadata.txt" #metadata
meta <- read.table(file = metadata, header = TRUE)
View(meta)
#Renames the levels in Size and Time correctly
meta$Size <- factor(meta$Size, levels = c("1", "2", "3", "14", "15", "16", "17"), labels=
c("Blank", "FOS", "Initial", "180-250", "250-300", "300-500", "500-800"))
meta$Time <- factor(meta$Time, levels = c("0", "12", "24", "48"), labels= c("0hrs", "12hrs",
"24hrs", "48hrs"))
#Stores the sample name info as the rownames of the dataframe rather (uses the Group/sample
name as the rowname)
rownames(meta) <- meta$Group

#Saves this cleaned metadata as dataframe for future reference
save(meta, file="tables/meta.Rda")
#Get ready and cleaned meta dataframe saved in table folder
load("tables/meta.Rda")
View(meta)

#Read in data on rarefaction
V4rfact <- read_tsv(file = "data/stability.opti_mcc.groups.rarefaction")%>%
  select(-contains("lci-"), -contains("hci-")) %>%
  gather(-numsampled, key=sample, value=coverage) %>%
  mutate(sample=str_replace_all(sample, pattern="0.03-", replacement="")) %>%
  drop_na()
View(V4rfact)

#Merge rarefaction file with metadata
V4meta_rare <- meta%>%
  #sample_n(20) %>%
  merge(., V4rfact, by.x= "Group", by.y = "sample")

############################## Get rarefaction curve #############################

#graph rarefaction plot with vertical line where subsampling cutoff is for size
ggplot(V4meta_rare, aes(x=numsampled, y=coverage, group=Group, color=Size)) +
  geom_line()+
  geom_vline(xintercept=3000) +
  coord_cartesian(xlim=c(0,20000)) +
  labs(x="Number of Sequences Sampled per Subject",
       y="Number of OTUs per Subject") +
  theme_classic()
ggsave("graphs/Rarefaction_Size.png")

#graph rarefaction plot with vertical line where subsampling cutoff is for time
ggplot(V4meta_rare, aes(x=numsampled, y=coverage, group=Group, color=Time)) +
  geom_line()+
  geom_vline(xintercept=3000) +
  coord_cartesian(xlim=c(0,20000)) +
  labs(x="Number of Sequences Sampled per Subject",
       y="Number of OTUs per Subject") +
  theme_classic()
ggsave("graphs/Rarefaction_Time.png")

##############################################################################Al
```

## Alpha diversity & Repeated ANOVA

```
#Set working directory appropriately
setwd("~/Desktop/ANSC_project")

#Loading necessary packages
library(readr)
library(stringr)
library(ggplot2)
library(tidyr)

#Reading in tables: Diversity metrics
alpha_div <- read.table(file = "data/stability.opti_mcc.groups.ave-std.summary", sep = "\t", header = T)
View(alpha_div)
#Cleaning the table by removing unneeded rows and columns
alpha_div <- alpha_div[,c(-1)] #to remove the first column (label)

#Read in metadata
metadata <- "data/size_time.metadata.txt" #metadata
meta <- read.table(file = metadata, header = TRUE)
View(meta)
#Renames the levels in Size and Time correctly
meta$Size <- factor(meta$Size, levels = c("1", "2", "3", "14", "15", "16", "17"), labels= c("Blank", "FOS", "Initial", "180-250", "250-300", "300-500", "500-800"))
meta$Time <- factor(meta$Time, levels = c("0", "12", "24", "48"), labels= c("0hrs", "12hrs", "24hrs", "48hrs"))
#Stores the sample name info as the rownames of the dataframe rather (uses the Group/sample name as the rowname)
rownames(meta) <- meta$Group

#Merge alpha diversity and meta dataframes together
alpha_div_merge <- merge(meta, alpha_div, by.x = "Group", by.y = "group")
#Delete entries/rows with “method= std” and leave those with “method=ave”
alpha_div_merge <- subset(alpha_div_merge, method=="ave")
View(alpha_div_merge)

unique(alpha_div_merge$Time)
unique(alpha_div_merge$Size)

#################### Comparing samples for specific time point ####################

#Checking Alpha Diversity boxplots for different Size fractions compared across each Time point
qplot(Time, shannon, geom = "boxplot", colour = Size, data = alpha_div_merge, size = I(0.3))
```

```
#Get figures for manuscript for different Size fractions compared at each Time point
chao <- ggplot(alpha_div_merge, aes(Time, chao)) +
  geom_boxplot(aes(color = Size)) +
ggsave("graphs/chao_size by time.png")

shannoneven <- ggplot(alpha_div_merge, aes(Time, shannoneven)) +
  geom_boxplot(aes(color = Size)) +
ggsave("graphs/shannoneven_size by time_time.png")

shannon <- ggplot(alpha_div_merge, aes(Time, shannon)) +
  geom_boxplot(aes(color = Size)) +
ggsave("graphs/shannon_size by time.png")

#Run the ANOVAs for statistics for different Size fractions compared at each Time point
Size <- unique(alpha_div_merge$Size)
Time <- unique(alpha_div_merge$Time)

#Size with Chao
ad_metrics <- c("chao")
for(t in Time){
  print(t)
  for(m in ad_metrics){
    print(m)
    aov_temp <- aov(get(m) ~ Size, data = subset(alpha_div_merge, Time == t))
    summary(aov_temp)
    anova_summary <- as.data.frame(summary(aov_temp)[[1]])
    write.table(anova_summary, file = paste0("graphs/anova_chao_Size", t, ".txt"), sep = "\t",
quote = FALSE)
  }
}

#Size with Shannon
ad_metrics <- c("shannon")
for(t in Time){
  print(t)
  for(m in ad_metrics){
    print(m)
    aov_temp <- aov(get(m) ~ Size, data = subset(alpha_div_merge, Time == t))
    summary(aov_temp)
    anova_summary <- as.data.frame(summary(aov_temp)[[1]])
    write.table(anova_summary, file = paste0("graphs/anova_shannon_Size", t, ".txt"), sep = "\t",
quote = FALSE)
  }
}

#Size with Shannoneven
```

```
ad_metrics <- c("shannoneven")
for(t in Time){
  print(t)
  for(m in ad_metrics){
    print(m)
    aov_temp <- aov(get(m) ~ Size, data = subset(alpha_div_merge, Time == t))
    summary(aov_temp)
    anova_summary <- as.data.frame(summary(aov_temp)[[1]])
    write.table(anova_summary, file = paste0("graphs/anova_shannoneven_Size", t, ".txt"), sep = "\t", quote = FALSE)
  }
}

################# Comparing different time points for each sample #################

#Checking Alpha Diversity boxplots for different Time points compared across each Size fraction
qplot(Size, chao, geom = "boxplot", colour = Time, data = alpha_div_merge, size = I(0.3))
qplot(Size, shannoneven, geom = "boxplot", colour = Time, data = alpha_div_merge, size = I(0.3))
qplot(Size, shannon, geom = "boxplot", colour = Time, data = alpha_div_merge, size = I(0.3))

#Get figures for manuscript for different Time points compared across each Size fraction
chao <- ggplot(alpha_div_merge, aes(Size, chao)) +
  geom_boxplot(aes(color = Time))
ggsave("graphs/chao_size_time.png")

shannoneven <- ggplot(alpha_div_merge, aes(Size, shannoneven)) +
  geom_boxplot(aes(color = Time))
ggsave("graphs/shannoneven_size_time.png")

shannon <- ggplot(alpha_div_merge, aes(Size, shannon)) +
  geom_boxplot(aes(color = Time))
ggsave("graphs/shannon_size_time.png")

#Run the ANOVAs for statistics for different Time points compared for each Size fraction
Size <- unique(alpha_div_merge$Size)
Time <- unique(alpha_div_merge$Time)

#Time with Chao
ad_metrics <- c("chao")
for(s in Size){
  print(s)
  for(m in ad_metrics){
    print(m)
    aov_temp <- aov(get(m) ~ Time, data = subset(alpha_div_merge, Size == s))
    summary(aov_temp)
    anova_summary <- as.data.frame(summary(aov_temp)[[1]])
    write.table(anova_summary, file = paste0("graphs/anova_chao_Time", s, ".txt"), sep = "\t", quote = FALSE)
  }
}

#Time with Shannoneven
ad_metrics <- c("shannoneven")
for(s in Size){
  print(s)
  for(m in ad_metrics){
    print(m)
    aov_temp <- aov(get(m) ~ Time, data = subset(alpha_div_merge, Size == s))
    summary(aov_temp)
    anova_summary <- as.data.frame(summary(aov_temp)[[1]])
    write.table(anova_summary, file = paste0("graphs/anova_shannoneven_Time", s, ".txt"), sep = "\t", quote = FALSE)
  }
}

#Time with Shannon
ad_metrics <- c("shannon")
for(s in Size){
  print(s)
  for(m in ad_metrics){
    print(m)
    aov_temp <- aov(get(m) ~ Time, data = subset(alpha_div_merge, Size == s))
    summary(aov_temp)
    anova_summary <- as.data.frame(summary(aov_temp)[[1]])
    write.table(anova_summary, file = paste0("graphs/anova_shannon_Time", s, ".txt"), sep = "\t", quote = FALSE)
  }
}

##################################################################################
```

## PERMANOVA

```
#Set working directory appropriately
setwd("~/Desktop/ANSC_project")

library(vegan)
library(ggplot2)
library(tidyr)
library(dplyr)

################################# FUNCTIONS #################################
pairwise.adonis <- function(x,factors, sim.method, p.adjust.m)
{
  library(vegan)
  co = as.matrix(combn(unique(factors),2))
  pairs = c()
  F.Model =c()
  R2 = c()
  p.value = c()

  for(elem in 1:ncol(co)){
    ad = adonis(x[factors %in% c(as.character(co[1,elem]),as.character(co[2,elem])),] ~
                 factors[factors %in% c(as.character(co[1,elem]),as.character(co[2,elem]))] , method
=sim.method, permutations = 9999);
    pairs = c(pairs,paste(co[1,elem],'vs',co[2,elem]));
    F.Model =c(F.Model,ad$aov.tab[1,4]);
    R2 = c(R2,ad$aov.tab[1,5]);
    p.value = c(p.value,ad$aov.tab[1,6])
  }
  p.adjusted = p.adjust(p.value,method=p.adjust.m)
  pairw.res = data.frame(pairs,F.Model,R2,p.value,p.adjusted)
  return(pairw.res)
}

veganCovEllipse <- function (cov, center = c(0,0), scale = 1, npoints = 100){
  theta <- (0:npoints) * 2 * pi/npoints
  Circle <- cbind(cos(theta), sin(theta))
  t(center + scale * t(Circle %*% chol(cov)))
}

##################################### DATA ###################################

#Read in OTU table from scratch to keep Group as a column
otu_table <-"data/stability.opti_mcc.0.03.subsample.shared" #rarefied OTU table
otu_subsample <- read.table(otu_table, header = TRUE)
View(otu_subsample)
```

```
#Stores the sample name info as the rownames of the dataframe rather (Uses the Group/sample name as the rowname)
rownames(otu_subsample) <- otu_subsample$Group
#Cleans the dataframe by removing unneeded columns
otu_subsample <- otu_subsample[-c(0,1)]
otu_subsample <- otu_subsample[-c(0,2)]
View(otu_subsample)

#Read in metadata
metadata <- "data/size_time.metadata.txt" #metadata
meta <- read.table(file = metadata, header = TRUE)
View(meta)
#Renames the levels in Size and Time correctly
meta$Size <- factor(meta$Size, levels = c("1", "2", "3", "14", "15", "16", "17"), labels= c("Blank", "FOS", "Initial", "180-250", "250-300", "300-500", "500-800"))
meta$Time <- factor(meta$Time, levels = c("0", "12", "24", "48"), labels= c("0hrs", "12hrs", "24hrs", "48hrs"))
#Stores the sample name info as the rownames of the dataframe rather (uses the Group/sample name as the rowname)
rownames(meta) <- meta$Group

#Make sure that the meta table and the otu table have the same samples
meta <- meta[meta$Group %in% rownames(otu_subsample),]
otu_subsample <- otu_subsample[rownames(otu_subsample) %in% meta$Group,]

#Merge otu_subsample and metadata
meta_otu_subsample <- merge(meta, otu_subsample, by.x = 'Group', by.y = 'Group')
View(meta_otu_subsample)
str(meta_otu_subsample) #To make sure that Size and Time are “Factor”

#To count how many rows and columns are there in the dataframe
ncol(meta_otu_subsample)
nrow(meta_otu_subsample)

#To conduct pairwise PERMANOVA analysis by size and time (need to specify the exact columns of the OTUs since need only numeric variables for function to work)
pairwise.adonis(meta_otu_subsample[,4:1917], meta_otu_subsample$Time, sim.method="bray", p.adjust.m = "fdr")
pairwise.adonis(meta_otu_subsample[,4:1917], meta_otu_subsample$Size, sim.method="bray", p.adjust.m = "fdr")
################################################################################
```

**NMDS script for Beta Diversity, AMOVA & HOMOVA**

```
#Set working directory appropriately
setwd("~/Desktop/ANSC_project")
```

```
library(vegan)
library(ggplot2)
library(tidyr)
library(dplyr)

############################### FUNCTIONS ####################################
pairwise.adonis <- function(x,factors, sim.method, p.adjust.m)
{
  library(vegan)
  co = as.matrix(combn(unique(factors),2))
  pairs = c()
  F.Model =c()
  R2 = c()
  p.value = c()

  for(elem in 1:ncol(co)){
    ad = adonis(x[factors %in% c(as.character(co[1,elem]),as.character(co[2,elem])),] ~
                 factors[factors %in% c(as.character(co[1,elem]),as.character(co[2,elem]))] , method
=sim.method, permutations = 9999);
    pairs = c(pairs,paste(co[1,elem],'vs',co[2,elem]));
    F.Model =c(F.Model,ad$aov.tab[1,4]);
    R2 = c(R2,ad$aov.tab[1,5]);
    p.value = c(p.value,ad$aov.tab[1,6])
  }
  p.adjusted = p.adjust(p.value,method=p.adjust.m)
  pairw.res = data.frame(pairs,F.Model,R2,p.value,p.adjusted)
  return(pairw.res)
}

veganCovEllipse <- function (cov, center = c(0,0), scale = 1, npoints = 100){
  theta <- (0:npoints) * 2 * pi/npoints
  Circle <- cbind(cos(theta), sin(theta))
  t(center + scale * t(Circle %*% chol(cov)))
}

############################### DATA ####################################
#Read in OTU table
otu_table <-"data/stability.opti_mcc.0.03.subsample.shared" #rarefied OTU table
otu_subsample <- read.table(otu_table, header = TRUE)
View(otu_subsample)
#Stores the sample name info as the rownames of the dataframe rather (Uses the Group/sample
name as the rowname)
rownames(otu_subsample) <- otu_subsample$Group
otu_subsample <- otu_subsample[-c(0,1)] #remove unneeded rows and columns
otu_subsample <- otu_subsample[-c(0,1)] #remove unneeded rows and columns
otu_subsample <- otu_subsample[-c(0,1)] #remove unneeded rows and columns
```

```
#Saves this cleaned otu_subsample as a table for future reference
save(otu_subsample, file="tables/ otu_subsample.Rda")

#Read in metadata
metadata <- "data/size_time.metadata.txt" #metadata
meta <- read.table(file = metadata, header = TRUE)
View(meta)
#Renames the levels in Size and Time correctly
meta$Size <- factor(meta$Size, levels = c("1", "2", "3", "14", "15", "16", "17"), labels=
c("Blank", "FOS", "Initial", "180-250", "250-300", "300-500", "500-800"))
meta$Time <- factor(meta$Time, levels = c("0", "12", "24", "48"), labels= c("0hrs", "12hrs",
"24hrs", "48hrs"))
#Stores the sample name info as the rownames of the dataframe rather (uses the Group/sample
name as the rowname)
rownames(meta) <- meta$Group

#Makes sure that the meta table and the otu table have the same samples
meta <- meta[meta$Group %in% rownames(otu_subsample),]
otu_subsample <- otu_subsample[rownames(otu_subsample) %in% meta$Group,]

# This calculates the distance matrix using Bray-Curtis distances with vegan
dist.matr.bray <- vegdist(otu_subsample, method = 'bray')
# This is vegan's function to make an NMDS ordination using k=2 dimensions
mds <- metaMDS(dist.matr.bray, k = 2,trymax = 1000, autotransform = FALSE)
#Calculation of the ordination stress
mds$stress

#Creates nmds dataframe from mds distance matrices
nmds <-as.data.frame(mds$points)
nmds$Group <- rownames(nmds)
View(nmds)

####################### NMDS PLOTS for all time points #######################
#Merging nmds and metadata
metanmds <- merge(meta, nmds, by.x = 'Group', by.y = 'Group')
View(metanmds)
str(metanmds) #To make sure that Size and Time are “Factor”

#General plots with basic facets
ggplot(metanmds, aes(x=MDS1, y=MDS2)) + geom_point(aes(color=Size))
ggplot(metanmds, aes(x=MDS1, y=MDS2)) + geom_point(aes(color=Time))
ggplot(metanmds, aes(x=MDS1, y=MDS2)) + geom_point(aes(color=Time, shape=Size))
ggplot(metanmds, aes(x=MDS1, y=MDS2)) + geom_point(aes(color=Size, shape=Time))

#Plot for manuscript
ggplot(metanmds, aes(x=MDS1, y=MDS2)) + geom_point(aes(color=Size)) +
  labs(x='MDS1', y= 'MDS2', caption = paste('Ordination stress: ', round(mds$stress, digits = 2)))
ggsave("graphs/nmds_size.png", height = 5, width = 7)

ggplot(metanmds, aes(x=MDS1, y=MDS2)) + geom_point(aes(color=Time)) +
  labs(x='MDS1', y= 'MDS2', caption = paste('Ordination stress: ', round(mds$stress, digits = 2)))
ggsave("graphs/nmds_time.png", height = 5, width = 7)

ggplot(metanmds, aes(x=MDS1, y=MDS2)) + geom_point(aes(color=Size, shape=Time)) +
  labs(x='MDS1', y= 'MDS2')
ggsave("graphs/nmds_size_time.png", height = 5, width = 7)

ggplot(metanmds, aes(x=MDS1, y=MDS2)) + geom_point(aes(color=Time, shape=Size)) +
  labs(x='MDS1', y= 'MDS2', caption = paste('Ordination stress: ', round(mds$stress, digits = 2)))
ggsave("graphs/nmds_time_size.png", height = 5, width = 7)

########################### NMDS PLOTS at 24 hours ##########################

#Creates meta only for samples at 24 hrs
meta_24 <- meta[-c(1:21,34:45,52:57),] #remove all time points other 24hrs
View(meta_24)

#Merging nmds and meta_24
metanmds <- merge(meta_24, nmds, by.x = 'Group', by.y = 'Group')
View(metanmds)
str(metanmds) #To make sure that Size and Time are “Factor”

#General plots with basic facets
ggplot(metanmds, aes(x=MDS1, y=MDS2)) + geom_point(aes(color=Size))

#Plot for manuscript with specifying limits for x-axis and y-axis
ggplot(metanmds, aes(x=MDS1, y=MDS2)) + geom_point(aes(color=Size)) + xlim(-0.2,0.45) +
ylim(-0.15,0.1)
  labs(x='MDS1', y= 'MDS2', caption = paste('Ordination stress: ', round(mds$stress, digits = 2)))
ggsave("graphs/nmds_size_24.png", height = 5, width = 7)

########################### NMDS PLOTS at 48 hours ##########################

#Creates meta only for samples at 48 hrs
meta_48 <- meta[-c(1:33, 46:51),] #remove all time points other 48hrs
View(meta_48)

#Creates nmds only for samples at 48 hrs
nmds_48 <- nmds[-
c(1,2,4,5,7,8,10,11,13,14,16,17,19,20,22,23,25,26,28,29,31,32,34,35,37,38,40,41,43,44,46,47,49
,50,52,53,55,56,57),] #remove all time points other 48hrs
View(meta_48)
```

```
#Merging nmds and meta_48
metanmds <- merge(meta_48, nmds_48, by.x = 'Group', by.y = 'Group')
View(metanmds)
str(metanmds) #To make sure that Size and Time are “Factor”

#General plots with basic facets
ggplot(metanmds, aes(x=MDS1, y=MDS2)) + geom_point(aes(color=Size))

#Plot for manuscript
ggplot(metanmds, aes(x=MDS1, y=MDS2)) + geom_point(aes(color=Size)) + xlim(-0.2,0.45) +
ylim(-0.15,0.1) +  labs(x='MDS1', y= 'MDS2', caption = paste('Ordination stress: ',
round(mds$stress, digits = 2)))
ggsave("graphs/nmds_size_48.png", height = 5, width = 7)

###############################################################################
```

## Bar Graphs for Relative Abundance

```
#Set working directory appropriately
setwd("~/Desktop/ANSC_project")

#Load libraries
library(ggplot2)
library(vegan)
library(dplyr)
library(scales)
library(grid)
library(reshape2)
library(tidyr)

#Get otu_subsample dataframe saved in the table folder
load("tables/otu_subsample.Rda")
View(otu_subsample)
#Get meta dataframe saved in table folder
load("tables/meta.Rda")
View(meta)
#Get nmds dataframe saved in table folder
load("tables/nmds.Rda")
View(nmds)
#Get metanmds dataframe saved in table folder
load("tables/metanmds.Rda")
View(metanmds)

#Assign variables for the paths of the data to import for Phylotype table and taxa
sharedfile <- "data/phylotype.tx.1.subsample.shared" #Phylo table
taxfile <- "data/phylotype.cons.correct.taxonomy" #Phylo taxa
taxonomy <-read.table(taxfile, header = T)
View(taxonomy)
#Clean the taxonomy dataframe by separating the taxa
taxonomy <- separate(data = taxonomy, col = Taxonomy, into = c("kingdom", "phylum",
"class", "family", "order", "genus", "species"), sep = ";")
str(taxonomy)

# Set colors for plotting
my_colors <- c(
  '#a6cee3','#1f78b4','#b2df8a','#33a02c','#fb9a99','#e31a1c',
  '#fdbf6f','#ff7f00','#cab2d6','#6a3d9a','#ffff99','#b15928',
  "#CBD588", "#5F7FC7", "orange","#DA5724", "#508578", "#CD9BCD",
   "#AD6F3B", "#673770","#D14285", "#652926", "#C84248",
  "#8569D5", "#5E738F","#D1A33D", "#8A7C64", "#599861", "black"
)
```

```
rm(otu.summary)
otu.summary <- prop.table(as.matrix(otu_subsample), 1)
otu_abund <- colSums(otu.summary)
otu.summary <- rbind(otu_abund, otu.summary)
otu.summary_sorted <- otu.summary[,order(otu.summary[1,], decreasing = TRUE)]

########################### Bar Graph at genus level ###########################

#Top 15 most abundant genera
num_genera <- 15 # enter the number of genera you want
melt_otu <- melt(otu.summary_sorted[,c(1:num_genera)])
colnames(melt_otu) <- c("Sample", "OTU", "Abundance")
View(melt_otu)
str(melt_otu)

#Putting it all together
#Merge melt_otu and metanmds
meta_otu <- merge(metanmds, melt_otu, by.x = "Group", by.y = "Sample")
View(meta_otu)
#Merge meta_otu and taxonomy tables
meta_otu_tax <- merge(meta_otu, taxonomy, by.x = "OTU", by.y = "OTU")
View(meta_otu_tax)
str(meta_otu_tax)
summary(meta_otu_tax$Group)

#Sorting based on MDS1 from negative to positive (NMDS axis 1)
meta_otu_tax <- meta_otu_tax[order(meta_otu_tax$MDS1),]

#ordering samples based on NMDS axis 1
meta_otu_tax$Group <- factor(meta_otu_tax$Group,
levels=unique(as.character(meta_otu_tax$Group)))

#MAKE A GRAPH! Plot individuals not group means
ggplot(meta_otu_tax, aes(x = Group, y = Abundance, fill = genus)) +
  geom_bar(stat = "identity") +
  scale_fill_manual(values = my_colors) +
  # Remove x axis title
  theme(axis.title.x = element_blank()) +
  ylim(c(0,1)) +
  guides(fill = guide_legend(reverse = F, keywidth = .5, keyheight = .5, ncol = 1)) +
  theme(legend.text=element_text(size=8)) +
  #theme(legend.position="bottom") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, vjust = 0.5)) +
  ylab(paste0("Relative Abundance (top ", num_genera, " genera)")) +
  ggtitle("Genus Composition by Size & Time sorted by NMDS Axis 1")
ggsave("graphs/GenusBarPlotNMDS1_Phylo.png", width = 10, height = 4)
```

```
#################### Bar Graphs at genus level at 24 hours ###################

#Top 15 most abundant genera
num_genera <- 15 # enter the number of genera you want
melt_otu <- melt(otu.summary_sorted[,c(1:num_genera)])
colnames(melt_otu) <- c("Sample", "OTU", "Abundance")
View(melt_otu)
str(melt_otu)

#Creates metanmds only for samples at 24 hours
metanmds_24 <- metanmds[-c(55, 56, 57, 1, 4, 7, 10, 13, 16, 19, 22, 25, 28, 31, 34, 37, 40, 43,
46, 49, 52, 3, 6, 9, 12, 15, 18, 21, 24, 27, 30, 33, 36, 39, 42, 45, 48, 51, 54),] #remove all time
points other 24 hrs
View(metanmds_24)
#Save metanmds_24 for later
save(metanmds_24, file="tables/ metanmds_24.Rda")

#Putting it all together
#Merge melt_otu and metanmds
meta_otu <- merge(metanmds_24, melt_otu, by.x = "Group", by.y = "Sample")
View(meta_otu)
#Merge meta_otu and taxonomy tables
meta_otu_tax <- merge(meta_otu, taxonomy, by.x = "OTU", by.y = "OTU")
View(meta_otu_tax)
str(meta_otu_tax)
summary(meta_otu_tax$Group)

#Sorting based on Size
meta_otu_tax <- meta_otu_tax[order(meta_otu_tax$Size.x),]

#ordering samples based on NMDS axis 1
meta_otu_tax$Group <- factor(meta_otu_tax$Group,
levels=unique(as.character(meta_otu_tax$Group)))

#MAKE A GRAPH! Plot individuals not group means
ggplot(meta_otu_tax, aes(x = Group, y = Abundance, fill = genus)) +
  geom_bar(stat = "identity") +
  scale_fill_manual(values = my_colors) +
  # Remove x axis title
  theme(axis.title.x = element_blank()) +
  ylim(c(0,1)) +
  guides(fill = guide_legend(reverse = F, keywidth = .5, keyheight = .5, ncol = 1)) +
  theme(legend.text=element_text(size=8)) +
  #theme(legend.position="bottom") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, vjust = 0.5)) +
  ylab(paste0("Relative Abundance of top 15 most abundant genera")) +
  ggtitle("Genus Composition by Size at 24hrs")
ggsave("graphs/GenusBarPlotNMDS1_24hrs_Phylo.png", width = 10, height = 4)

#################### Bar Graphs at genus level at 48 hours ###################

#Top 15 most abundant genera
num_genera <- 15 # enter the number of genera you want
melt_otu <- melt(otu.summary_sorted[,c(1:num_genera)])
colnames(melt_otu) <- c("Sample", "OTU", "Abundance")
View(melt_otu)
str(melt_otu)

#Creates metanmds only for samples at 48 hours
metanmds_48 <- metanmds[-c(55, 56, 57, 1, 4, 7, 10, 13, 16, 19, 22, 25, 28, 31, 34, 37, 40, 43,
46, 49, 52, 2, 5, 8, 11, 14, 17, 20, 23, 26, 29, 32, 35, 38, 41, 44, 47, 50, 53),]
View(metanmds_48)
#Save metanmds_48 for later
save(metanmds_48, file="tables/ metanmds_48.Rda")

#Putting it all together
#Merge melt_otu and metanmds
meta_otu <- merge(metanmds_24, melt_otu, by.x = "Group", by.y = "Sample")
View(meta_otu)
#Merge meta_otu and taxonomy tables
meta_otu_tax <- merge(meta_otu, taxonomy, by.x = "OTU", by.y = "OTU")
View(meta_otu_tax)
str(meta_otu_tax)
summary(meta_otu_tax$Group)

#Sorting based on Size
meta_otu_tax <- meta_otu_tax[order(meta_otu_tax$Size.x),]

#ordering samples based on NMDS axis 1
meta_otu_tax$Group <- factor(meta_otu_tax$Group,
levels=unique(as.character(meta_otu_tax$Group)))

#MAKE A GRAPH! Plot individuals not group means
ggplot(meta_otu_tax, aes(x = Group, y = Abundance, fill = genus)) +
  geom_bar(stat = "identity") +
  scale_fill_manual(values = my_colors) +
  # Remove x axis title
  theme(axis.title.x = element_blank()) +
  ylim(c(0,1)) +
  guides(fill = guide_legend(reverse = F, keywidth = .5, keyheight = .5, ncol = 1)) +
  theme(legend.text=element_text(size=8)) +
  #theme(legend.position="bottom") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, vjust = 0.5)) +
  ylab(paste0("Relative Abundance of top 15 most abundant genera")) +
  ggtitle("Genus Composition by Size at 48hrs")
ggsave("graphs/GenusBarPlotNMDS1_48hrs_Phylo.png", width = 10, height = 4)

########################## Bar Graph at phylum level ##########################

#All phyla at all time points
melt_otu <- melt(otu.summary_sorted)
colnames(melt_otu) <- c("Sample", "OTU", "Abundance")
View(melt_otu)
str(melt_otu)

#Putting it all together
#Merge melt_otu and metanmds
meta_otu <- merge(metanmds, melt_otu, by.x = "Group", by.y = "Sample")
View(meta_otu)
#Merge meta_otu and taxonomy tables
meta_otu_tax <- merge(meta_otu, taxonomy, by.x = "OTU", by.y = "OTU")
View(meta_otu_tax)
str(meta_otu_tax)
summary(meta_otu_tax$Group)

#Sorting based on Size
meta_otu_tax <- meta_otu_tax[order(meta_otu_tax$Size.x),]

#ordering samples based on NMDS axis 1
meta_otu_tax$Group <- factor(meta_otu_tax$Group,
levels=unique(as.character(meta_otu_tax$Group)))

#MAKE A GRAPH! Plot individuals not group means
ggplot(meta_otu_tax, aes(x = Group, y = Abundance, fill = phylum)) +
  geom_bar(stat = "identity") +
  scale_fill_manual(values = my_colors) +
  # Remove x axis title
  theme(axis.title.x = element_blank()) +
  ylim(c(0,1)) +
  guides(fill = guide_legend(reverse = F, keywidth = .5, keyheight = .5, ncol = 1)) +
  theme(legend.text=element_text(size=8)) +
  #theme(legend.position="bottom") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, vjust = 0.5)) +
  ylab(paste0("Relative Abundance of phyla")) +
  ggtitle("Phylum Composition by Size & Time")
ggsave("graphs/PhylumBarPlot_Phylo.png", width = 10, height = 4)
```

```
#################### Bar Graphs at phylum level at 24 hours ######################

melt_otu <- melt(otu.summary_sorted)
colnames(melt_otu) <- c("Sample", "OTU", "Abundance")
View(melt_otu)
str(melt_otu)

#Creates metanmds only for samples at 24 hours
metanmds_24 <- metanmds[-c(55, 56, 57, 1, 4, 7, 10, 13, 16, 19, 22, 25, 28, 31, 34, 37, 40, 43,
46, 49, 52, 3, 6, 9, 12, 15, 18, 21, 24, 27, 30, 33, 36, 39, 42, 45, 48, 51, 54),]
View(metanmds_24)

#Putting it all together
#Merge melt_otu and metanmds
meta_otu <- merge(metanmds_24, melt_otu, by.x = "Group", by.y = "Sample")
View(meta_otu)
#Merge meta_otu and taxonomy tables
meta_otu_tax <- merge(meta_otu, taxonomy, by.x = "OTU", by.y = "OTU")
View(meta_otu_tax)
str(meta_otu_tax)
summary(meta_otu_tax$Group)

#Sorting based on Size
meta_otu_tax <- meta_otu_tax[order(meta_otu_tax$Size.x),]

#ordering samples based on NMDS axis 1
meta_otu_tax$Group <- factor(meta_otu_tax$Group,
levels=unique(as.character(meta_otu_tax$Group)))

#MAKE A GRAPH! Plot individuals not group means
ggplot(meta_otu_tax, aes(x = Group, y = Abundance, fill = phylum)) +
  geom_bar(stat = "identity") +
  scale_fill_manual(values = my_colors) +
  # Remove x axis title
  theme(axis.title.x = element_blank()) +
  ylim(c(0,1)) +
  guides(fill = guide_legend(reverse = F, keywidth = .5, keyheight = .5, ncol = 1)) +
  theme(legend.text=element_text(size=8)) +
  #theme(legend.position="bottom") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, vjust = 0.5)) +
  ylab(paste0("Relative Abundance of Phyla")) +
  ggtitle("Phylum Composition by Size at 24hrs")
ggsave("graphs/PhylumBarPlot_24hrs_Phylo.png", width = 10, height = 4)

#################### Bar Graphs at phylum level at 48 hours ######################
```

```r
melt_otu <- melt(otu.summary_sorted)
colnames(melt_otu) <- c("Sample", "OTU", "Abundance")
View(melt_otu)
str(melt_otu)

#Creates metanmds only for samples at 48 hours
metanmds_48 <- metanmds[-c(55, 56, 57, 1, 4, 7, 10, 13, 16, 19, 22, 25, 28, 31, 34, 37, 40, 43,
46, 49, 52, 2, 5, 8, 11, 14, 17, 20, 23, 26, 29, 32, 35, 38, 41, 44, 47, 50, 53),]
View(metanmds_48)

#Putting it all together
#Merge melt_otu and metanmds
meta_otu <- merge(metanmds_48, melt_otu, by.x = "Group", by.y = "Sample")
View(meta_otu)
#Merge meta_otu and taxonomy tables
meta_otu_tax <- merge(meta_otu, taxonomy, by.x = "OTU", by.y = "OTU")
View(meta_otu_tax)
str(meta_otu_tax)
summary(meta_otu_tax$Group)

#Sorting based on Size
meta_otu_tax <- meta_otu_tax[order(meta_otu_tax$Size.x),]

#ordering samples based on NMDS axis 1
meta_otu_tax$Group <- factor(meta_otu_tax$Group,
levels=unique(as.character(meta_otu_tax$Group)))

#MAKE A GRAPH! Plot individuals not group means
ggplot(meta_otu_tax, aes(x = Group, y = Abundance, fill = phylum)) +
  geom_bar(stat = "identity") +
  scale_fill_manual(values = my_colors) +
  # Remove x axis title
  theme(axis.title.x = element_blank()) +
  ylim(c(0,1)) +
  guides(fill = guide_legend(reverse = F, keywidth = .5, keyheight = .5, ncol = 1)) +
  theme(legend.text=element_text(size=8)) +
  #theme(legend.position="bottom") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, vjust = 0.5)) +
  ylab(paste0("Relative Abundance of Phyla")) +
  ggtitle("Phylum Composition by Size at 48hrs")
ggsave("graphs/PhylumBarPlot_48hrs_Phylo.png", width = 10, height = 4)

################################################################################
```

## LefSe Analysis

```
#Set working directory appropriately
setwd("~/Desktop/ANSC_project")

library(MASS)
library(tidyr)
library(ggplot2)
library(dplyr)

#Read in files
summary_lefse <-"data/phylotype.tx.1.subsample.1.lefse_summary"
#read in lefse data
lefse_out <- read.csv(summary_lefse, sep="\t", header=T)
View(lefse_out)

lefse_NMDS_filt <- lefse_out[lefse_out$Class != "-",]
View(lefse_NMDS_filt)

#log transfrom the LDA values
lefse_NMDS_filt$logLDA <- log(lefse_NMDS_filt$LDA)

#read in the taxonomy file and separate the taxa into columns
taxfile <- "data/phylotype.cons.taxonomy"
taxonomy<- read.table(file=taxfile, sep="\t", header=T)
taxonomy <- separate(data = taxonomy, col = Taxonomy, into = c("kingdom", "phylum",
"class", "family", "order", "genus", "species"), sep = ";")
View(taxonomy)

#merge the lefse and taxonomy by OTU
lefse_tax <- merge(lefse_NMDS_filt, taxonomy, by.x="OTU", by.y="OTU")
View(lefse_tax)

#order the data.frame by logLDA
lefse_tax$genus<- reorder(lefse_tax$genus, lefse_tax$LDA)
str(lefse_tax)
View(lefse_tax)

#Creates lefse_tax_14_17 only for samples with finest and coarsest corn bran sizes
lefse_tax_14_17 <- lefse_tax[-
c(4,7,8,13,18,19,23,30,37,39,2,27,14,17,24,29,32,3,6,11,15,21,25,34,36,9,20,31,38),]
View(lefse_tax_14_17)

#Renames the levels in the “Size” variable correctly
lefse_tax_14_17$Class <- factor(lefse_tax_14_17$Class, levels = c("14", "17"), labels= c("180-
250", "500-800"))
```

```
# Set colors for plotting
my_colors <- c(
  '#a6cee3','#1f78b4','#b2df8a','#33a02c','#fb9a99','#e31a1c',
  '#fdbf6f','#ff7f00','#cab2d6','#6a3d9a','#ffff99','#b15928',
  "#CBD588", "#5F7FC7", "orange","#DA5724", "#508578", "#CD9BCD",
  "#AD6F3B", "#673770","#D14285", "#652926", "#C84248",
  "#8569D5", "#5E738F","#D1A33D", "#8A7C64", "#599861", "black"
)

#Plot and save for manuscript
ggplot(lefse_tax_14_17, aes(x = genus, y = LDA, fill = Class)) +
  geom_bar(stat = "identity") +
  facet_grid(Class~.) +
  #scale_fill_manual(values = my_colors) +
  # Remove x axis title
  #theme(axis.title.x = element_blank()) +
  #ylim(c(0,1)) +
  guides(fill = guide_legend(reverse = F, keywidth = .5, keyheight = .5)) +
  theme(legend.text=element_text(size=8)) +
  #theme(legend.position="bottom") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, vjust = 0.5, size = 10)) +
  coord_flip()
ggsave(file="graphs/lefse_pwpt.png", width=6, height=4)
#dev.off()

##########################################################################
```