scp leonar25@bell.rcac.purdue.edu:**/scratch/bell/leonar25/qiime/manifest\_import**/rooted-tree.qza ~/Desktop/cluster

taxonomy.qza

#step 1, cd into the proper directory. This directory must already exist

cd $RCAC\_SCRATCH

cd 2021LL02/

**Getting ready to run qiime:**

sinteractive -A microbiome -t 5:00:00 -n12

module load bioinfo

module load Qiime/2-2021.11

module list

Things I need in folder before starting:

* .fastq files
* metadata.tsv file
* manifest file

Get the databases to classify the sequences

wget \

-O "gg-13-8-99-515-806-nb-classifier.qza" \

"https://data.qiime2.org/2021.11/common/gg-13-8-99-515-806-nb-classifier.qza"

**Import data into qiime**

qiime tools import \

--type 'SampleData[PairedEndSequencesWithQuality]' \

--input-path manifest.txt \

--input-format PairedEndFastqManifestPhred33V2 \

--output-path demux.qza

Convert .qza to .qzv

qiime demux summarize \

--i-data demux.qza \

--o-visualization demux.qzv

check demux.qzv to see where to truncate seq in following step:

scp [leonar25@bell.rcac.purdue.edu:**/scratch/bell/leonar25/qiime/manifest\_import**/\*\*.qzv](mailto:leonar25@bell.rcac.purdue.edu:/scratch/bell/leonar25/qiime/manifest_import/**.qzv) ~/Desktop/cluster

**Dada2**

qiime dada2 denoise-paired \

--i-demultiplexed-seqs demux.qza \

--p-trim-left-f 0 \

--p-trim-left-r 0 \

--p-trunc-len-f 245 \

--p-trunc-len-r 230 \

--p-n-threads 0 \

--o-representative-sequences rep-seqs.qza \

--o-table table.qza \

--o-denoising-stats stats-dada2.qza \

--verbose

Convert .qza to .qzv

qiime metadata tabulate \

--m-input-file stats-dada2.qza \

--o-visualization stats-dada2.qzv

qiime feature-table summarize \

--i-table table.qza \

--o-visualization table.qzv \

--m-sample-metadata-file metadata.txt

qiime feature-table tabulate-seqs \

--i-data rep-seqs.qza \

--o-visualization rep-seqs.qzv

**Alpha and Beta Diversity**

**TAXONOMY**

qiime feature-classifier classify-sklearn \

--i-classifier gg-13-8-99-515-806-nb-classifier.qza \

--i-reads rep-seqs.qza \

--o-classification taxonomy.qza

qiime metadata tabulate \

--m-input-file taxonomy.qza \

--o-visualization taxonomy.qzv

qiime taxa barplot \

--i-table table.qza \

--i-taxonomy taxonomy.qza \

--m-metadata-file LL\_metadata.txt \

--o-visualization taxa-bar-plots.qzv

**Ancom**

Filtered table with only experiment 1=

qiime feature-table filter-samples \

--i-table table.qza \

--m-metadata-file LL\_metadata.txt \

--p-where "[experiment]='1'" \

--o-filtered-table exp-1-table.qza

ANCOM operates on a FeatureTable[Composition] QIIME 2 artifact, which is based on frequencies of features on a per-sample basis, but cannot tolerate frequencies of zero. To build the composition artifact, a FeatureTable[Frequency] artifact must be provided to add-pseudocount (an imputation method), which will produce the FeatureTable[Composition] artifact.:

qiime composition add-pseudocount \

--i-table exp-1-table.qza \

--o-composition-table comp-exp-1-table.qza

We can then run ANCOM on the group column to determine what features differ in abundance across the samples of the different diet groups.

qiime composition ancom \

--i-table comp-exp-1-table.qza \

--m-metadata-file LL\_metadata.txt \

--m-metadata-column group \

--o-visualization ancom-group.qzv

We’re also often interested in performing a differential abundance test at a specific taxonomic level. To do this, we can collapse the features in our FeatureTable[Frequency] **at the taxonomic level of interest**, and then re-run the above steps. (i.e. level 6 of the Greengenes taxonomy-Genus level).

qiime taxa collapse \

--i-table exp-1-table.qza \

--i-taxonomy taxonomy.qza \

--p-level 6 \

--o-collapsed-table exp-1-table-l6.qza

qiime composition add-pseudocount \

--i-table exp-1-table-l6.qza \

--o-composition-table comp-exp-1-table-l6.qza

qiime composition ancom \

--i-table comp-exp-1-table-l6.qza \

--m-metadata-file LL\_metadata.txt \

--m-metadata-column group \

--o-visualization l6-ancom-group.qzv

Create a phylogenetic tree. (Phylogenetic method)

qiime phylogeny align-to-tree-mafft-fasttree \

--i-sequences rep-seqs.qza \

--o-alignment aligned-rep-seqs.qza \

--o-masked-alignment masked-aligned-rep-seqs.qza \

--o-tree unrooted-tree.qza \

--o-rooted-tree rooted-tree.qza

Alpha Rarefaction

qiime diversity alpha-rarefaction \

--i-table table.qza \

--i-phylogeny rooted-tree.qza \

--p-max-depth 72000 \

--m-metadata-file LL\_metadata.txt \

--o-visualization alpha-rarefaction.qzv

**Alpha and Beta Diversity**

Core-metrics (gives general alpha and beta diversity graphs)

qiime diversity core-metrics-phylogenetic \

--i-phylogeny rooted-tree.qza \

--i-table table.qza \

--p-sampling-depth 10897 \

--m-metadata-file LL\_metadata.txt \

--output-dir core-metrics-results

Alpha-group-significance

qiime diversity alpha-group-significance \

--i-alpha-diversity core-metrics-results/faith\_pd\_vector.qza \

--m-metadata-file LL\_metadata.txt \

--o-visualization core-metrics-results/faith-pd-group-significance.qzv

qiime diversity alpha-group-significance \

--i-alpha-diversity core-metrics-results/evenness\_vector.qza \

--m-metadata-file LL\_metadata.txt \

--o-visualization core-metrics-results/evenness-group-significance.qzv

qiime diversity alpha-group-significance \

--i-alpha-diversity core-metrics-results/shannon\_vector.qza \

--m-metadata-file LL\_metadata.txt \

--o-visualization core-metrics-results/shannon-group-significance.qzv

qiime diversity alpha-group-significance \

--i-alpha-diversity core-metrics-results/observed\_features\_vector.qza \

--m-metadata-file LL\_metadata.txt \

--o-visualization core-metrics-results/observed\_features-group-significance.qzv

ALPHA CORRELATION

qiime diversity alpha-correlation \

--i-alpha-diversity core-metrics-results/faith\_pd\_vector.qza \

--m-metadata-file LL\_metadata.txt \

--p-method spearman \

--o-visualization core-metrics-results/faith-correlation.qzv

qiime diversity alpha-correlation \

--i-alpha-diversity core-metrics-results/evenness\_vector.qza \

--m-metadata-file LL\_metadata.txt \

--p-method spearman \

--o-visualization core-metrics-results/evenness-correlation.qzv

qiime diversity alpha-correlation \

--i-alpha-diversity core-metrics-results/shannon\_vector.qza \

--m-metadata-file LL\_metadata.txt \

--p-method spearman \

--o-visualization core-metrics-results/shannon-correlation.qzv

qiime diversity alpha-correlation \

--i-alpha-diversity core-metrics-results/observed\_features\_vector.qza \

--m-metadata-file LL\_metadata.txt \

--o-visualization core-metrics-results/observed\_features-correlation.qzv

**Random Forest (Sample Classifier)**

qiime sample-classifier classify-samples \

--i-table table.qza \

--m-metadata-file LL\_metadata.txt \

--m-metadata-column group \

--p-optimize-feature-selection \

--p-parameter-tuning \

--p-estimator RandomForestClassifier \

--p-n-estimators 20 \

--p-random-state 123 \

--output-dir LL-classifier

qiime metadata tabulate \

--m-input-file LL\_metadata.txt \

--o-visualization LL-classifier/predictions.qzv

qiime metadata tabulate \

--m-input-file LL\_metadata.txt \

--o-visualization LL-classifier/probabilities.qzv

qiime metadata tabulate \

--m-input-file LL\_metadata.txt \

--o-visualization LL-classifier/feature\_importance.qzv

qiime feature-table filter-features \

--i-table table.qza \

--m-metadata-file LL\_metadata.txt \

--o-filtered-table LL-classifier/important-feature-table.qza

qiime sample-classifier heatmap \

--i-table table.qza \

--i-importance LL-classifier/feature\_importance.qza \

--m-sample-metadata-file LL\_metadata.txt \

--m-sample-metadata-column group \

--p-group-samples \

--p-feature-count 30 \

--o-filtered-table LL-classifier/important-feature-table-top-30.qza \

--o-heatmap LL-classifier/important-feature-heatmap.qzv

qiime sample-classifier heatmap \

--i-table table.qza \

--i-importance LL-classifier/feature\_importance.qza \

--m-sample-metadata-file LL\_metadata.txt \

--m-sample-metadata-column group \

--m-feature-metadata-file taxonomy.qza \

--m-feature-metadata-column Taxon \

--p-group-samples \

--p-feature-count 30 \

--o-filtered-table LL-classifier/important-feature-table-taxonomy-top-30.qza \

--o-heatmap LL-classifier/important-feature-heatmap-taxonomy.qzv

Copy all files to computer:

scp leonar25@bell.rcac.purdue.edu:**/scratch/bell/leonar25/qiime/manifest\_import**/LL-classifier/\*\*.qz\* ~/Desktop/cluster

**R Code:**

**Alpha Diversity:**

##############################################################

# title: "Alpha diversity in R - qiime2 output"

# author: "ANSC595"

# date: "March 16, 2021"

##############################################################

#The first step is very important. You need to set your working

#directory. Just like in unix we have to `cd` to where our data is, the

#same thing is true for R.

# To get these files you need to scp them from the cluster:

#

# first on your laptop cd to the directory where you want to save them.

# Then use this code for our example dataset today:

# mkdir core-metrics-results/

# scp <user.name>@bell.rcac.purdue.edu:/depot/microbiome/data/ANSC595/class\_materials/moving\_pictures\_pipeline/qiime\_out/\*.qz\* .

# scp <user.name>@bell.rcac.purdue.edu:/depot/microbiome/data/ANSC595/class\_materials/moving\_pictures\_pipeline/qiime\_out/core-metrics-results/\*.qz\* core-metrics-results/

# scp <user.name>@bell.rcac.purdue.edu:/depot/microbiome/data/ANSC595/class\_materials/moving\_pictures\_pipeline/qiime\_out/\*.tsv .

##############################################

#So where `~/Desktop/ANSC595/moving-pictures` is in the code below, you

#need to enter the path to where you saved the tutorial or your data.

setwd('~/Desktop/ANSC595Project/')

# Modified from the original online version available at

# http://rpubs.com/dillmcfarlan/R\_microbiotaSOP

# and Tutorial: Integrating QIIME2 and R for data visualization

# and analysis using qiime2R

# https://forum.qiime2.org/t/tutorial-integrating-qiime2-and-r-for-data-visualization-and-analysis-using-qiime2r/4121

##Goal

# The goal of this tutorial is to demonstrate basic analyses of microbiota data to determine if and how communities differ by variables of interest. In general, this pipeline can be used for any microbiota data set that has been clustered into operational taxonomic units (OTUs).

#

# This tutorial assumes some basic statistical knowledge. Please consider if your data fit the assumptions of each test (normality? equal sampling? Etc.). If you are not familiar with statistics at this level, we strongly recommend collaborating with someone who is. The incorrect use of statistics is a pervasive and serious problem in the sciences so don't become part of the problem! That said, this is an introductory tutorial and there are many, many further analyses that can be done with microbiota data. Hopefully, this is just the start for your data!

##Data

# The data used here are from the qiime2 moving pictures tutorial.

# Please see their online tutorial for an explanation of the dataset.

##Files

# We will use the following files created using the qiime2 moving pictures tutorial.

# core-metrics-results/evenness\_vector.qza (alpha diversity)

# core-metrics-results/faith\_pd\_vector.qza (alpha diversity)

# core-metrics-results/observed\_features\_vector.qza (alpha diversity)

# core-metrics-results/shannon\_vector.qza (alpha diversity)

# sample-metadata.tsv (metadata)

# Data manipulation

## Load Packages

if (!requireNamespace("devtools", quietly = TRUE)){install.packages("devtools")}

devtools::install\_github("jbisanz/qiime2R") # current version is 0.99.20

library(tidyverse)

library(qiime2R)

library(ggpubr)

##Load Data

# In the code, the text before = is what the file will be called in R.

# Make this short but unique as this is how you will tell R to use this

# file in later commands.

# header: tells R that the first row is column names, not data

# row.names: tells R that the first column is row names, not data

# sep: tells R that the data are tab-delimited.

# If you had a comma-delimited file, you would us sep=","

# Load data

meta<-read\_q2metadata("LL\_metadata.txt")

str(meta)

colnames(meta)[4] <- "day.of.diet"

str(meta)

evenness = read\_qza("FromCluster/Core-metrics- Sampling\_Depth\_10897/qza/evenness\_vector.qza")

evenness<-evenness$data %>% rownames\_to\_column("SampleID") # this moves the sample names to a new column that matches the metadata and allows them to be merged

observed\_features = read\_qza("FromCluster/Core-metrics- Sampling\_Depth\_10897/qza/observed\_features\_vector.qza")

observed\_features<-observed\_features$data %>% rownames\_to\_column("SampleID") # this moves the sample names to a new column that matches the metadata and allows them to be merged

shannon = read\_qza("FromCluster/Core-metrics- Sampling\_Depth\_10897/qza/shannon\_vector.qza")

shannon<-shannon$data %>% rownames\_to\_column("SampleID") # this moves the sample names to a new column that matches the metadata and allows them to be merged

faith\_pd = read\_qza("FromCluster/Core-metrics- Sampling\_Depth\_10897/qza/faith\_pd\_vector.qza")

faith\_pd<-faith\_pd$data %>% rownames\_to\_column("SampleID") # this moves the sample names to a new column that matches the metadata and allows them to be merged\

## Clean up the data

# You can look at your data by clicking on it in the upper-right

# quadrant "Environment"

# You always need to check the data types in your tables to make

# sure they are what you want. We will now change some data types

# in the meta now

str(meta)

#observed\_features$observed\_features\_num <- lapply(observed\_features$observed\_features, as.numeric)

#observed\_features$observed\_features <- as.numeric(observed\_features$observed\_features)

str(observed\_features)

###Alpha Diversity tables

# These tables will be merged for convenience and added to the

# metadata table as the original tutorial was organized.

alpha\_diversity = merge(x=faith\_pd, y=evenness, by.x = "SampleID", by.y = "SampleID")

alpha\_diversity = merge(alpha\_diversity, observed\_features, by.x = "SampleID", by.y = "SampleID")

alpha\_diversity = merge(alpha\_diversity, shannon, by.x = "SampleID", by.y = "SampleID")

meta = merge(meta, alpha\_diversity, by.x = "SampleID", by.y = "SampleID")

row.names(meta) = meta$SampleID

#meta = meta[,-1]

str(meta)

#Alpha-diversity

# Alpha-diversity is within sample diversity. It is how many

# different species (OTUs) are in each sample (richness) and how

# evenly they are distributed (evenness), which together are diversity.

# Each sample has one value for each metric.

##Explore alpha metrics

# Now we will start to look at our data. We will first start with

# alpha-diversity and richness.

#

# You want the data to be roughly normal so that you can run ANOVA

# or t-tests. If it is not normally distributed, you will need to

# consider if you should normalize the data or usenon-parametric

# tests such as Kruskal-Wallis.

# Here, we see that none of the data are normally distributed,

# with the exception of "Faith" and "Observed Features".

#Plots

hist(meta$shannon, main="Shannon diversity", xlab="", breaks=10)

hist(meta$faith\_pd, main="Faith phylogenetic diversity", xlab="", breaks=10)

hist(meta$pielou\_e, main="Evenness", xlab="", breaks=10)

hist(as.numeric(meta$observed\_features), main="Observed Features", xlab="", breaks=10)

#Plots the qq-plot for residuals

ggqqplot(meta$shannon, title = "Shannon")

ggqqplot(meta$faith\_pd, title = "Faith PD")

ggqqplot(meta$pielou\_e, title = "Evenness")

ggqqplot(meta$observed\_features, title = "Observed Features")

# To test for normalcy statistically, we can run the Shapiro-Wilk

# test of normality.

shapiro.test(meta$shannon)

shapiro.test(meta$pielou\_e)

shapiro.test(meta$faith\_pd)

shapiro.test(meta$observed\_features)

# The null hypothesis of these tests is that “sample distribution

# is normal”. If the test is significant, the distribution is non-normal.

# We see that, as expected from the graphs, shannon and evenness

# are normally distributed.

#Overall, for alpha-diversity:

# ANOVA, t-test, or general linear models with the normal distribution

# are used when the data is roughly normal. Transforming the data to

# achieve a normal distribution could also be completed.

#

# Kruskal-Wallis, Wilcoxon rank sum test, or general linear models

# with another distribution are used when the data is not normal or if

# the n is low, like less than 30.

# Our main variables of interest are

# body site: gut, tongue, right palm, left palm

# subject: 1 and 2

# month-year: 10-2008, 1-2009, 2-2009, 3-2009, 4-2009

## Categorical variables

# Now that we know which tests can be used, let's run them.

## Normally distributed metrics

# Since it's the closest to normalcy, we will use \*\*Evenness\*\* as an

#example. First, we will test body site, which is a categorical variable

# with more than 2 levels. Thus, we run ANOVA. If age were only two

# levels, we could run a t-test

# Does body site impact the Evenness of the microbiota?

#Run the ANOVA and save it as an object

aov.evenness.body\_site = aov(pielou\_evenness ~ body.site, data=meta)

#Call for the summary of that ANOVA, which will include P-values

summary(aov.evenness.body\_site)

#To do all the pairwise comparisons between groups and correct for multiple comparisons, we run Tukey's honest significance test of our ANOVA.

TukeyHSD(aov.evenness.body\_site)

# We clearly see that the evenness between hands and gut are different.

# When we plot the data, we see that evenness decreases in the gut

# compared to palms.

levels(meta$body.site)

#Re-order the groups because the default is alphabetical order

meta$body.site.ord = factor(meta$body.site, c("left palm", "right palm", "gut", "tongue"))

levels(meta$body.site.ord)

#Plot

boxplot(pielou\_evenness ~ group, data=meta, ylab="Peilou evenness")

evenness <- ggplot(meta, aes(group, pielou\_evenness)) +

geom\_boxplot(aes(color = group)) +

#ylim(c(0.5,1)) +

theme\_q2r() +

theme(axis.text.x = element\_text(angle = 90, hjust = 1, vjust = 0.5))

ggsave("output/evenness.png", evenness, height = 3, width = 3)

# Now, the above graph is kind of not correct. Our test and our graphic do not exactly match. ANOVA and Tukey are tests based on the mean, but the boxplot plots the median. Its not wrong, its just not the best method. Unfortunately plotting the average and standard deviation is a little complicated.

evenness\_summary <- meta %>% # the names of the new data frame and the data frame to be summarised

group\_by(body.site.ord) %>% # the grouping variable

summarise(mean\_evenness = mean(pielou\_evenness), # calculates the mean of each group

sd\_evenness = sd(pielou\_evenness), # calculates the standard deviation of each group

n\_evenness = n(), # calculates the sample size per group

se\_evenness = sd(pielou\_evenness)/sqrt(n())) # calculates the standard error of each group

# We can now make a bar plot of means vs body site, with standard

# deviations or standard errors as the error bar. The following code

# uses the standard deviations.

evenness\_se <- ggplot(evenness\_summary, aes(body.site.ord, mean\_evenness, fill = body.site.ord)) +

geom\_col() +

geom\_errorbar(aes(ymin = mean\_evenness - se\_evenness, ymax = mean\_evenness + se\_evenness), width=0.2) +

theme\_q2r() +

theme(axis.text.x = element\_text(angle = 90, hjust = 1, vjust = 0.5)) +

theme(legend.title = element\_blank()) +

labs(y="Pielou's evenness ± s.e.", x = "")

ggsave("output/evenness\_se.png", evenness\_se, height = 3, width = 3)

## \*\*Non-normally distributed metrics\*\*

# We will use \*\*Faith's phylogenetic diversity\*\* here. Since body site

# is categorical, we use Kruskal-Wallis (non-parametric equivalent of

# ANOVA). If we have only two levels, we would run Wilcoxon rank sum

# test (non-parametric equivalent of t-test)

kruskal.test(faith\_pd ~ body.site.ord, data=meta)

# We can test pairwise within the age groups with Wilcoxon Rank Sum

# Tests. This test has a slightly different syntax than our other tests

pairwise.wilcox.test(meta$faith\_pd, meta$body.site.ord, p.adjust.method="BH")

# Like evenness, we see that pd also increases with age.

#Plot

boxplot(faith\_pd ~ group, data=meta, ylab="Faith phylogenetic diversity")

# or with ggplot2

faith\_pd <- ggplot(meta, aes(group, faith\_pd)) +

geom\_boxplot(aes(color = group)) +

#ylim(c(0.5,1)) +

theme\_q2r() +

theme(axis.text.x = element\_text(angle = 90, hjust = 1, vjust = 0.5)) +

theme(legend.title = element\_blank()) +

labs(y="Faith Phylogenetic Diversity", x = "")

ggsave("output/Faith\_pd.png", faith\_pd, height = 3, width = 3)

##Continuous variables

# For continuous variables, we use general linear models, specifying

# the distribution that best fits our data.

# \*\*Normally distributed metrics\*\*

# Since days.since.experiment.start is a continuous variable, we run a

# general linear model. We will again use evenness as our roughly normal

# metric. The default of `glm` and `lm` is the normal distribution so we

# don't have to specify anything.

# Does days.since.experiment.start impact evenness of the microbiota?

glm.evenness.time = glm(pielou\_evenness ~ days.since.experiment.start, data=meta)

summary(glm.evenness.time)

#The output let's us know that the intercept of our model is significantly different from 0 but our slope (\*e.g.\* our variable of interest) is not. This makes sense when we look at the data.

plot(pielou\_evenness ~ days.since.experiment.start, data=meta)

#Add the glm best fit line

plot(pielou\_evenness ~ days.since.experiment.start, data=meta) + abline(glm.evenness.time)

# \*\*Non-normally distributed metrics\*\*

# We will again use a \*general linear model\* for our non-normally

# distributed metric Faith\_pd. However, this time, we change the

# distribution from normal to something that fits the data better.

# But which distribution should we choose? In statistics, there is no

# one "best" model. There are only good and better models. We will use

# the plot() function to compare two models and pick the better one.

# First, the Gaussian (normal) distribution, which we already know is a bad fit.

gaussian.faith.time = glm(faith\_pd ~ days.since.experiment.start, data=meta, family="gaussian")

plot(gaussian.faith.time, which=c(1,2))

# Quasipoisson (log) distribution

qp.faith.time = glm(faith\_pd ~ days.since.experiment.start, data=meta, family="quasipoisson")

plot(qp.faith.time, which=c(1,2))

# What we're looking for is no pattern in the Residuals vs. Fitted graph

# ("stars in the sky"), which shows that we picked a good distribution

# family to fit our data. We also want our residuals to be normally

# distributed, which is shown by most/all of the points falling on the

# line in the Normal Q-Q plot.

# While it's still not perfect, the quasipoisson fits much better.

# In the residuals vs fitted graph, the y axis is from -2 to 4 whereas

# the axis with gaussian was from -5 to 10. So, we will use quasipoisson

# and see that ADG does not to correlate to Chao richness.

summary(qp.faith.time)

# Plotting this we see that, indeed, there is a trend toward correlation between Faith\_pd and days.since.experiment.start.

#Plot

plot(log(faith\_pd) ~ days.since.experiment.start, data=meta, ylab="ln(Faith Phylo. Diversity)")

plot(log(faith\_pd) ~ days.since.experiment.start, data=meta, ylab="ln(Faith Phylo. Diversity)") + abline(qp.faith.time)

#Plot

observed.features <- ggplot(meta, aes(group, observed\_features)) +

geom\_boxplot(aes(color = group)) +

#ylim(c(0.5,1)) +

theme\_q2r() +

theme(axis.text.x = element\_text(angle = 90, hjust = 1, vjust = 0.5))

ggsave("output/observed.features.png", evenness, height = 3, width = 3)

#Plot

evenness <- ggplot(meta, aes(group, shannon\_entropy)) +

geom\_boxplot(aes(color = group)) +

#ylim(c(0.5,1)) +

theme\_q2r() +

theme(axis.text.x = element\_text(angle = 90, hjust = 1, vjust = 0.5))

ggsave("output/shannon.png", evenness, height = 3, width = 3)

Beta Diversity:

#Install qiime2R if you need to:

install.packages("remotes")

#Load the packages

install.packages("remotes")

remotes::install\_github("jbisanz/qiime2R")

library(qiime2R)

library(tidyverse)

##############################################

#Set UP

#

#These are the things that we need from Qiime:

#

#sample-metadata.tsv

#core-metrics-results/bray\_curtis\_pcoa\_results.qza

#core-metrics-results/weighted\_unifrac\_pcoa\_results.qza

#core-metrics-results/rarefied\_table.qza

#rooted-tree.qza

#taxonomy.qza

#core-metrics-results/evenness\_vector.qza

#core-metrics-results/faith\_pd\_vector.qza

#core-metrics-results/observed\_otus\_vector.qza

#core-metrics-results/shannon\_vector.qza

#

# To get these files you need to scp them from the cluster:

#

# first on your laptop cd to the directory where you want to save them.

# Then use this code for our example dataset today:

# mkdir core-metrics-results/

# scp john2185@bell.rcac.purdue.edu:/depot/microbiome/data/2021\_ANSC595/john2185/qiime/moving\_pictures\_pipeline/\* .

# scp john2185@bell.rcac.purdue.edu:/depot/microbiome/data/2021\_ANSC595/john2185/qiime/moving\_pictures\_pipeline/core-metrics-results/\* core-metrics-results/.

##############################################

setwd('~/Desktop/ANSC595Project/')

list.files()

if(!dir.exists("output"))

dir.create("output")

metadata<-read\_q2metadata("LL\_metadata.txt")

#metadata2 <- read.delim("sample-metadata.tsv", sep = "\t", header = T, quote = "", stringsAsFactors = F)

#metadata2 <- metadata2[-1,]

str(metadata)

levels(metadata$`group`)

row.names(metadata) <- metadata[ ,1]

#metadata <- metadata[,-1]

row.names(metadata)

setwd('~/Desktop/ANSC595Project/FromCluster/Core-metrics-Sampling\_Depth\_10897/qza')

list.files()

bc\_PCoA<-read\_qza("bray\_curtis\_pcoa\_results.qza")

body\_colors <- c("Black", "Blue", "Green", "Gray","Pink","Yellow", "Red")

bc\_meta <- bc\_PCoA$data$Vectors %>%

select(SampleID, PC1, PC2) %>%

inner\_join(metadata, by = c("SampleID" = "SampleID"))

my\_column <- "group"

#my\_column <- "DietTreatment"

ggplot(bc\_meta, aes(x=PC1, y=PC2, color=get(my\_column))) +

geom\_point() + #alpha controls transparency and helps when points are overlapping

theme\_q2r() +

xlab(paste0("PC1 (", round(100\*bc\_PCoA$data$ProportionExplained[1], digits = 2), "%)")) +

ylab(paste0("PC2 (", round(100\*bc\_PCoA$data$ProportionExplained[2], digits = 2), "%)")) +

scale\_color\_manual(values=body\_colors, name = my\_column)

ggsave(paste0("output/BC-basic\_", my\_column,".pdf"), height=2, width=3, device="pdf") # save a PDF 3 inches by 4 inches

centroids\_BC <- aggregate(cbind(PC1,PC2)~get(my\_column),bc\_meta,mean)

colnames(centroids\_BC)[1] <- "group"

ggplot(bc\_meta, aes(x=PC1, y=PC2, color=get(my\_column))) +

geom\_point() + #alpha controls transparency and helps when points are overlapping

geom\_point(data=centroids\_BC, size = 3) +

theme\_q2r() +

stat\_ellipse(level = 0.95, type = "t") +

xlab(paste0("PC1 (", round(100\*bc\_PCoA$data$ProportionExplained[1], digits = 2), "%)")) +

ylab(paste0("PC2 (", round(100\*bc\_PCoA$data$ProportionExplained[2], digits = 2), "%)")) +

scale\_color\_manual(values=body\_colors, name = my\_column)

ggsave(paste0("output/BC-ellipse\_", my\_column,".pdf"), height=3, width=4.5, device="pdf") # save a PDF 3 inches by 4 inches

ggplot(bc\_meta, aes(x=PC1, y=PC2, color=get(my\_column))) +

geom\_point(aes(shape= subject)) + #alpha controls transparency and helps when points are overlapping

#geom\_point(data=centroids, size = 3) +

theme\_q2r() +

#stat\_ellipse(level = 0.95, type = "t") +

xlab(paste0("PC1 (", round(100\*bc\_PCoA$data$ProportionExplained[1], digits = 2), "%)")) +

ylab(paste0("PC2 (", round(100\*bc\_PCoA$data$ProportionExplained[2], digits = 2), "%)"))

#scale\_color\_manual(values=corn\_colors, name = my\_column)

ggsave(paste0("output/BC-ellipse\_", my\_column,"-subject.pdf"), height=3, width=4.5, device="pdf") # save a PDF 3 inches by 4 inches

##SAME thing but with weighted UniFrac

setwd('~/Desktop/ANSC595Project/FromCluster/Core-metrics-Sampling\_Depth\_10897/qza')

Wuni\_PCoA<-read\_qza("weighted\_unifrac\_pcoa\_results.qza")

Wuni\_meta <- Wuni\_PCoA$data$Vectors %>%

select(SampleID, PC1, PC2) %>%

inner\_join(metadata, by = c("SampleID" = "SampleID"))

centroids\_Wuni <- aggregate(cbind(PC1,PC2)~get(my\_column),Wuni\_meta,mean)

ggplot(Wuni\_meta, aes(x=PC1, y=PC2, color=get(my\_column))) +

geom\_point() + #alpha controls transparency and helps when points are overlapping

#geom\_point(data=centroids, size = 3) +

theme\_q2r() +

stat\_ellipse(level = 0.95, type = "t") +

xlab(paste0("PC1 (", round(100\*Wuni\_PCoA$data$ProportionExplained[1], digits = 2), "%)")) +

ylab(paste0("PC2 (", round(100\*Wuni\_PCoA$data$ProportionExplained[2], digits = 2), "%)")) +

scale\_color\_manual(values=body\_colors, name = "Body Site")

setwd('~/Desktop/ANSC595Project/')

ggsave(paste0("output/Wuni-ellipse\_", my\_column,".pdf"), height=3, width=4.5, device="pdf") # save a PDF 3 inches by 4 inches

##SAME thing but with unweighted\_unifrac\_pcoa\_results.qza

setwd('~/Desktop/ANSC595Project/FromCluster/Core-metrics-Sampling\_Depth\_10897/qza')

unweighted\_PCoA<-read\_qza("unweighted\_unifrac\_pcoa\_results.qza")

unweighted\_meta <- unweighted\_PCoA$data$Vectors %>%

select(SampleID, PC1, PC2) %>%

inner\_join(metadata, by = c("SampleID" = "SampleID"))

centroids\_unweighted <- aggregate(cbind(PC1,PC2)~get(my\_column),unweighted\_meta,mean)

ggplot(unweighted\_meta, aes(x=PC1, y=PC2, color=get(my\_column))) +

geom\_point() + #alpha controls transparency and helps when points are overlapping

#geom\_point(data=centroids, size = 3) +

theme\_q2r() +

stat\_ellipse(level = 0.95, type = "t") +

xlab(paste0("PC1 (", round(100\*unweighted\_PCoA$data$ProportionExplained[1], digits = 2), "%)")) +

ylab(paste0("PC2 (", round(100\*unweighted\_PCoA$data$ProportionExplained[2], digits = 2), "%)")) +

scale\_color\_manual(values=body\_colors, name = "Body Site")

setwd('~/Desktop/ANSC595Project/')

ggsave(paste0("output/unweighted-ellipse\_", my\_column,".pdf"), height=3, width=4.5, device="pdf") # save a PDF 3 inches by 4 inches

##SAME thing but with Jaccard

setwd('~/Desktop/ANSC595Project/FromCluster/Core-metrics-Sampling\_Depth\_10897/qza')

Jaccard\_PCoA<-read\_qza("jaccard\_pcoa\_results.qza")

Jaccard\_meta <- Jaccard\_PCoA$data$Vectors %>%

select(SampleID, PC1, PC2) %>%

inner\_join(metadata, by = c("SampleID" = "SampleID"))

centroids\_Jaccard <- aggregate(cbind(PC1,PC2)~get(my\_column),Jaccard\_meta,mean)

ggplot(Jaccard\_meta, aes(x=PC1, y=PC2, color=get(my\_column))) +

geom\_point() + #alpha controls transparency and helps when points are overlapping

#geom\_point(data=centroids, size = 3) +

theme\_q2r() +

stat\_ellipse(level = 0.95, type = "t") +

xlab(paste0("PC1 (", round(100\*Jaccard\_PCoA$data$ProportionExplained[1], digits = 2), "%)")) +

ylab(paste0("PC2 (", round(100\*Jaccard\_PCoA$data$ProportionExplained[2], digits = 2), "%)")) +

scale\_color\_manual(values=body\_colors, name = "Body Site")

setwd('~/Desktop/ANSC595Project/')

ggsave(paste0("output/Jaccard-ellipse\_", my\_column,".pdf"), height=3, width=4.5, device="pdf") # save a PDF 3 inches by 4 inches

**Taxonomy**

## This script goes through making a taxa bar plot and then running

## DESeq2 to find differentially abundant ASVs

# for help installing phyloseq, see this website

# https://bioconductor.org/packages/release/bioc/html/phyloseq.html

# to install phyloseq:

# if (!requireNamespace("BiocManager", quietly = TRUE))

# install.packages("BiocManager")

#BiocManager::install("phyloseq")

library(qiime2R)

library(phyloseq)

library(zoo)

library(tidyverse)

##############################################

#Set UP

#

#These are the things that we need from Qiime:

#

#sample-metadata.tsv

#core-metrics-results/rarefied\_table.qza

#rooted-tree.qza

#taxonomy.qza

##############################################

setwd('~/Desktop/ANSC595Project/')

list.files()

if(!dir.exists("output"))

dir.create("output")

metadata<-read\_q2metadata("LL\_metadata.txt")

str(metadata)

row.names(metadata) <- metadata[ ,1]

#metadata <- metadata[,-1]

row.names(metadata)

##Qiime2r method of reading in the taxonomy files

taxonomy<-read\_qza("taxonomy.qza")

head(taxonomy$data)

tax.clean<-parse\_taxonomy(taxonomy$data)

head(tax.clean)

#All this is OK except that in future use of the taxonomy table,

#these ASVs will be ignored because they are not classified. Why

#are ASVs not classified? Its because there is not a close enough

#match in the database. Just because there is not a good match in

#the database does not mean they don’t exist, so I wanted to make

#sure this data was not lost. So in my new code, from lines 200 – 224

#I make it so that ASVs that are unclassified at any level are

#classified as the lowest taxonomic level for which there is a

#classification.

#Next, all these `NA` classifications with the last level that was

#classified

tax.clean[is.na(tax.clean)] <- ""

for (i in 1:nrow(tax.clean)){

if (tax.clean[i,2] == ""){

kingdom <- paste("Kingdom\_", tax.clean[i,1], sep = "")

tax.clean[i, 2:7] <- kingdom

} else if (tax.clean[i,3] == ""){

phylum <- paste("Phylum\_", tax.clean[i,2], sep = "")

tax.clean[i, 3:7] <- phylum

} else if (tax.clean[i,4] == ""){

class <- paste("Class\_", tax.clean[i,3], sep = "")

tax.clean[i, 4:7] <- class

} else if (tax.clean[i,5] == ""){

order <- paste("Order\_", tax.clean[i,4], sep = "")

tax.clean[i, 5:7] <- order

} else if (tax.clean[i,6] == ""){

family <- paste("Family\_", tax.clean[i,5], sep = "")

tax.clean[i, 6:7] <- family

} else if (tax.clean[i,7] == ""){

tax.clean$Species[i] <- paste("Genus",tax.clean$Genus[i], sep = "\_")

}

}

#################################################################

##Taxa barplot

#################################################################

physeq <- qza\_to\_phyloseq(

features="rarefied\_table.qza",

tree="rooted-tree.qza",

taxonomy = "taxonomy.qza",

metadata = "LL\_metadata.txt"

)

#First get the OTU table from physeq

physeq\_otu\_table <- data.frame(otu\_table(physeq), check.names = F)

tax.clean = tax.clean[row.names(tax.clean) %in% rownames(physeq\_otu\_table),]

metadata.filtered = metadata[row.names(metadata) %in% colnames(physeq\_otu\_table),]

#Assign as variables to be feed into phyloseq

OTU.physeq = otu\_table(as.matrix(physeq\_otu\_table), taxa\_are\_rows=TRUE)

#our edited and formatted taxonomy table from the top of this script

tax.physeq = tax\_table(as.matrix(tax.clean))

meta.physeq = sample\_data(metadata.filtered)

#We then merge these into an object of class phyloseq.

physeq\_bar\_plot = phyloseq(OTU.physeq, tax.physeq, meta.physeq)

# 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", "gray", "black"

)

#If you want different taxonomic level, find and replace the taxonomic level listed here

my\_level <- c("Phylum", "Family", "Genus")

my\_column <- "group" #this is the metadata column that we will use in the taxa barplot

rm(taxa.summary)

abund\_filter <- 0.05 # Our abundance threshold

#ml ="Genus"

for(ml in my\_level){

print(ml)

taxa.summary <- physeq\_bar\_plot %>%

tax\_glom(taxrank = ml, NArm = FALSE) %>% # agglomerate at `ml` level

transform\_sample\_counts(function(x) {x/sum(x)} ) %>% # Transform to rel. abundance

psmelt() %>% # Melt to long format

group\_by(get(my\_column), get(ml)) %>%

summarise(Abundance.average=mean(Abundance))

taxa.summary <- as.data.frame(taxa.summary)

colnames(taxa.summary)[1] <- my\_column

colnames(taxa.summary)[2] <- ml

physeq.taxa.max <- taxa.summary %>%

group\_by(get(ml)) %>%

summarise(overall.max=max(Abundance.average))

physeq.taxa.max <- as.data.frame(physeq.taxa.max)

colnames(physeq.taxa.max)[1] <- ml

# merging the phyla means with the metadata #

physeq\_meta <- merge(taxa.summary, physeq.taxa.max)

physeq\_meta\_filtered <- filter(physeq\_meta, overall.max>abund\_filter)

#str(physeq\_meta\_filtered)

# Plot

ggplot(physeq\_meta\_filtered, aes(x = get(my\_column), y = Abundance.average, fill = get(ml))) +

#facet\_grid(.~LitterTreatment) +

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)) +

theme(legend.title = element\_blank()) +

ylab("Relative Abundance") +

xlab(my\_column) +

ggtitle(paste0(ml, " (>", abund\_filter \* 100,"%) in at least 1 sample"))

ggsave(paste0("output/", ml, "BarPlot\_", my\_column, ".png"), height = 5, width = 4)

}

#################################################################

###Differential Abundance with DESeq2

#################################################################

#Adapted from https://joey711.github.io/phyloseq-extensions/DESeq2.html

#First load DESeq2.

#If you need help with DESeq2 install, see this website

# https://bioconductor.org/packages/release/bioc/html/DESeq2.html

if (!requireNamespace("BiocManager", quietly = TRUE))

install.packages("BiocManager")

BiocManager::install("DESeq2")

library("DESeq2")

#To use DESeq, we need no zeros in our OTU table. So we will edit the table by multiplying by 2 and + 1

#First get the OTU table from physeq

physeq\_otu\_table <- data.frame(otu\_table(physeq), check.names = FALSE)

OTU.clean2 <- physeq\_otu\_table + 1

#Now make the phyloseq object:

OTU.physeq = otu\_table(as.matrix(OTU.clean2), taxa\_are\_rows=TRUE)

tax.physeq = tax\_table(as.matrix(tax.clean))

meta.physeq = sample\_data(metadata.filtered)

#We then merge these into an object of class phyloseq.

physeq\_deseq = phyloseq(OTU.physeq, tax.physeq, meta.physeq)

#The following two lines actually do all the complicated DESeq2 work. The function phyloseq\_to\_deseq2 converts your phyloseq-format microbiome data into a DESeqDataSet with dispersions estimated, using the experimental design formula, also shown (the ~body.site term). The DESeq function does the rest of the testing, in this case with default testing framework, but you can actually use alternatives.

diagdds = phyloseq\_to\_deseq2(physeq\_deseq, ~ group)

diagdds = DESeq(diagdds, test="Wald", fitType="parametric")

#the test type of "Wald" tests for significance of coefficients in a Negative Binomial GLM. This is generally a pretty good assumption for sequencing experiments. This was designed with RNA-seq in mind, but also pretty good for 16S sequencing.

###Investigate test results table

#The following results function call creates a table of the results of the tests. Very fast. The hard work was already stored with the rest of the DESeq2-related data in our latest version of the diagdds object (see above). I then order by the adjusted p-value, removing the entries with an NA value. The rest of this example is just formatting the results table with taxonomic information for nice(ish) display in the HTML output.

#Contrast: this argument specifies what comparison to extract from the object to build a results table. There are exactly three elements:

# 1. the name of a factor in the design formula,

# 2. the name of the numerator level for the fold change, and

# 3. the name of the denominator level for the fold change (simplest case)

alpha = 0.05

my\_contrast = c("group", "FF", "FR")

#my\_contrast = c("body.site", "gut", "right palm")

#my\_contrast = c("body.site", "gut", "tongue")

#my\_contrast = c("body.site", "tongue", "left palm")

#my\_contrast = c("body.site", "tongue", "right palm")

#my\_contrast = c("body.site", "right palm", "left palm")

res = results(diagdds, contrast = my\_contrast, cooksCutoff = FALSE)

sigtab = res[which(res$padj < alpha), ]

#sigtab\_test <- as(sigtab, "data.frame")

sigtab = cbind(as(sigtab, "data.frame"), as(tax\_table(physeq\_deseq)[rownames(sigtab), ], "matrix"))

#head(sigtab)

###Volcano Plot

with(res, plot(log2FoldChange, -log10(padj), pch=20, main="Volcano plot", xlim=c(-15,15)))

# Add colored points: blue if padj<0.01, red if log2FC>1 and padj<0.05)

with(subset(res, padj<.01 ), points(log2FoldChange, -log10(pvalue), pch=20, col="blue"))

with(subset(res, padj<.01 & abs(log2FoldChange)>2), points(log2FoldChange, -log10(pvalue), pch=20, col="red"))

#Let's look at the OTUs that were significantly different between the two treatment groups. The following makes a nice ggplot2 summary of 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))

DESeq\_fig = ggplot(sigtab, aes(x=Genus, y = log2FoldChange, color=Phylum)) +

geom\_point(size=3) +

ylab(paste0("(", my\_contrast[2], "/", my\_contrast[3], ")\n", "log2FoldChange")) +

scale\_color\_manual(values = my\_colors[c(4,6,8,10,12,14,16,18,20)]) +

#ylim(0,8) +

theme(axis.text.x = element\_text(angle = -90, hjust = 0, vjust=0.5))

ggsave(paste0("output/DESeq2-FF\_vs\_FR", my\_contrast[2], "-", my\_contrast[3], ".png"), DESeq\_fig, height = 3, width = 10)