Phage_transfer

Feargal Ryan

May 4, 2017

16S analysis

```
library(ggplot2)
library(phyloseq)
library(ape)
library(gridExtra)
library(DESeq2)
library(xlsx)
library(plyr)
library(vegan)

raw_cnts = read.table("mouse2_raw_counts.txt",header=TRUE,row.names =
1,sep="\t")
meta_data = read.table("mouse2_meta_data.txt",header=TRUE,row.names = 1,sep =
"\t")
tree = read.tree("mouse2_OTU.phylo.tre")
taxa = read.table("mouse2_taxa.txt",header=TRUE, row.names = 1, sep="\t")
family_colors = read.xlsx("mouse_family_colors.xlsx",sheetIndex = 1)
rownames(family_colors) = family_colors$family


meta_data$Group_4 = factor(meta_data$Group_3,levels =
c("Pre_treatment_1","Pre_treatment_2","Post_AB",

"CT_1","CT_2","CT_3","CT_4","CT_5","CT_6","CT_7",

"PT_1","PT_2","PT_3","PT_4","PT_5","PT_6","PT_7"))

meta_data$Group_nu = factor(meta_data$Group_nu,levels =
c("CT000","CT00","CTAb","CT_1","CT_2","CT_3","CT_4","CT_5","CT_6","CCe_6",

"PT_1","PT_2","PT_3","PT_4","PT_5","PT_6","PCe_6"))


cnts_filt = raw_cnts[apply(raw_cnts>0,1,sum)>=round(ncol(raw_cnts)*0.05),]
cnts_filt_prop = prop.table(as.matrix(cnts_filt),2)
cnts_filt_prop = cnts_filt_prop * 100

norm_cnts = raw_cnts
norm_means = apply(norm_cnts,1,mean)
```

```r
a_s = norm_means
all_size = a_s
all_size[a_s>=as.numeric(summary(a_s)[5])] = "Biggest" # 75th percentile
all_size[a_s>=as.numeric(summary(a_s)[3]) & a_s<as.numeric(summary(a_s)[5]) ]
= "Second Biggest"# Median
all_size[a_s>=as.numeric(summary(a_s)[2]) & a_s<as.numeric(summary(a_s)[3]) ]
= "Second Smallest"# 25th percentile
all_size[a_s<as.numeric(summary(a_s)[2])] = "Smallest"# Median
all_size = gsub("Second Smallest",5,all_size)
all_size = gsub("Second Biggest",7,all_size)
all_size = gsub("Biggest",10,all_size)
all_size = as.numeric(gsub("Smallest",2,all_size))
names(all_size) = names(a_s)

OTU = otu_table(cnts_filt_prop,taxa_are_rows =TRUE)
map_data = sample_data(meta_data)
physeq = phyloseq(OTU,map_data,tree)

OTU_raw = otu_table(raw_cnts,taxa_are_rows =TRUE)
physeq_raw = phyloseq(OTU_raw,map_data,tree)
alpha_div = estimate_richness(physeq_raw)

braycurtis_dist = phyloseq::distance(physeq,method="bray")
braycurtis_pc = pcoa(braycurtis_dist)

unifrac_dist = phyloseq::distance(physeq,method="unifrac")
unifrac_pc = pcoa(unifrac_dist)

wunifrac_dist = phyloseq::distance(physeq,method="wunifrac")
wunifrac_pc = pcoa(wunifrac_dist)

adonis(braycurtis_dist ~ meta_data$AB)
adonis(unifrac_dist ~ meta_data$AB)
adonis(wunifrac_dist ~ meta_data$AB)

mpt = meta_data[grep("[PC]T",meta_data$PT),]
cpt = cnts_filt_prop[,rownames(mpt)]

OTU = otu_table(cpt,taxa_are_rows =TRUE)
map_data = sample_data(mpt)
physeq = phyloseq(OTU,map_data,tree)

braycurtis_dist = phyloseq::distance(physeq,method="bray")

unifrac_dist = phyloseq::distance(physeq,method="unifrac")

wunifrac_dist = phyloseq::distance(physeq,method="wunifrac")
```

```
adonis(braycurtis_dist ~ mpt$PT)
adonis(unifrac_dist ~ mpt$PT)
adonis(wunifrac_dist ~ mpt$PT)

BC_data.df = data.frame(meta_data,alpha_div,
                        "BC1" = braycurtis_pc$vectors[,1],
                        "BC2" = braycurtis_pc$vectors[,2],
                        "BC3" = braycurtis_pc$vectors[,3],
                        "UF1" = unifrac_pc$vectors[,1],
                        "UF2" = unifrac_pc$vectors[,2],
                        "WU1" = wunifrac_pc$vectors[,1],
                        "WU2" = wunifrac_pc$vectors[,2])

BC_points = ggplot(BC_data.df,aes(x = BC1, y =
BC2,color=Group_nu))+geom_point()+
  xlab(paste("PC1:",round(braycurtis_pc$values$Relative_eig[1]*100,2),"
variation"))+
  ylab(paste("PC2:",round(braycurtis_pc$values$Relative_eig[2]*100,2),"
variation"))+
  ggtitle("Bray Curtis OTU level mouse 2") +
theme(panel.background=element_blank())+
  stat_ellipse(aes(x = BC1,y =BC2,
fill=Group_nu),geom="polygon",level=0.7,alpha=0.3)

BC_labels = ggplot(BC_data.df,aes(x = BC1, y =
BC2,color=Group_nu,label=Group_nu))+geom_text()+
  xlab(paste("PC1:",round(braycurtis_pc$values$Relative_eig[1]*100,2),"
variation"))+
  ylab(paste("PC2:",round(braycurtis_pc$values$Relative_eig[2]*100,2),"
variation"))+
  ggtitle("Bray Curtis OTU level mouse 2") +
theme(panel.background=element_blank())+
  stat_ellipse(aes(x = BC1,y =BC2,
fill=Group_nu),geom="polygon",level=0.7,alpha=0.3)

WU_points = ggplot(BC_data.df,aes(x = WU1, y =
WU2,color=Group_nu))+geom_point()+
  xlab(paste("PC1:",round(wunifrac_pc$values$Relative_eig[1]*100,2),"
variation"))+
  ylab(paste("PC2:",round(wunifrac_pc$values$Relative_eig[2]*100,2),"
variation"))+
  ggtitle("Weighted Unifrac OTU level mouse 2") +
theme(panel.background=element_blank())+
  stat_ellipse(aes(x = WU1,y =WU2,
fill=Group_nu),geom="polygon",level=0.7,alpha=0.3)

WU_labels = ggplot(BC_data.df,aes(x = WU1, y =
WU2,color=Group_nu,label=Group_nu))+geom_text()+
  xlab(paste("PC1:",round(wunifrac_pc$values$Relative_eig[1]*100,2),"
```

```
variation"))+
  ylab(paste("PC2:",round(wunifrac_pc$values$Relative_eig[2]*100,2),"
variation"))+
  ggtitle("Weighted Unifrac OTU level mouse 2") +
theme(panel.background=element_blank())+
  stat_ellipse(aes(x = WU1,y =WU2,
fill=Group_nu),geom="polygon",level=0.7,alpha=0.3)

grid.arrange(BC_points,WU_points,ncol=2)
grid.arrange(BC_points,WU_points,nrow=2)
BC_labels
WU_labels

UF_points = ggplot(BC_data.df,aes(x = UF1, y =
UF2,color=Group_nu))+geom_point()+
  xlab(paste("PC1:",round(unifrac_pc$values$Relative_eig[1]*100,2),"
variation"))+
  ylab(paste("PC2:",round(unifrac_pc$values$Relative_eig[2]*100,2),"
variation"))+
  ggtitle("Unifrac OTU level mouse 2") +
theme(panel.background=element_blank())+
  stat_ellipse(aes(x = UF1,y =UF2,
fill=Group_nu),geom="polygon",level=0.7,alpha=0.3)

UF_labels = ggplot(BC_data.df,aes(x = UF1, y =
UF2,color=Group_nu,label=Group_nu))+geom_text()+
  xlab(paste("PC1:",round(unifrac_pc$values$Relative_eig[1]*100,2),"
variation"))+
  ylab(paste("PC2:",round(unifrac_pc$values$Relative_eig[2]*100,2),"
variation"))+
  ggtitle("Unifrac OTU level mouse 2") +
theme(panel.background=element_blank())+
  stat_ellipse(aes(x = UF1,y =UF2,
fill=Group_nu),geom="polygon",level=0.7,alpha=0.3)

UF_points
UF_labels

BC_data.df$Group_4 = factor(BC_data.df$Group_3,levels =
c("Pre_treatment_1","Pre_treatment_2","Post_AB",

"CT_1","CT_2","CT_3","CT_4","CT_5","CT_6","CT_7",

"PT_1","PT_2","PT_3","PT_4","PT_5","PT_6","PT_7"))
ggplot(BC_data.df,aes(x=Group_nu,y=Shannon,fill=Group_nu))+geom_boxplot()+
  theme_classic()+ggtitle("Alpha Diversity")+ theme(axis.text.x =
element_text(angle = 45, hjust = 1))

wilcox.test(BC_data.df[BC_data.df$Group_4=="CT_1","Shannon"],
```

```
            BC_data.df[BC_data.df$Group_4=="PT_1","Shannon"])

wilcox.test(BC_data.df[BC_data.df$Group_4=="CT_2","Shannon"],
            BC_data.df[BC_data.df$Group_4=="PT_2","Shannon"])

wilcox.test(BC_data.df[BC_data.df$Group_4=="CT_3","Shannon"],
            BC_data.df[BC_data.df$Group_4=="PT_3","Shannon"])

wilcox.test(BC_data.df[BC_data.df$Group_4=="CT_4","Shannon"],
            BC_data.df[BC_data.df$Group_4=="PT_4","Shannon"])

wilcox.test(BC_data.df[BC_data.df$Group_4=="CT_5","Shannon"],
            BC_data.df[BC_data.df$Group_4=="PT_5","Shannon"])

wilcox.test(BC_data.df[BC_data.df$Group_4=="CT_6","Shannon"],
            BC_data.df[BC_data.df$Group_4=="PT_6","Shannon"])

wilcox.test(BC_data.df[BC_data.df$Group_4=="CT_7","Shannon"],
            BC_data.df[BC_data.df$Group_4=="PT_7","Shannon"])

wilcox.test(BC_data.df[BC_data.df$Group_4=="CT_1","Shannon"],
            BC_data.df[BC_data.df$Group_4=="PT_1","Shannon"])

### Comparing to Pre treatment 2 ###
## 1 ##
wilcox.test(BC_data.df[BC_data.df$Group_nu=="CT00","Shannon"],
            BC_data.df[BC_data.df$Group_nu=="PT_1","Shannon"])

wilcox.test(BC_data.df[BC_data.df$Group_nu=="CT00","Shannon"],
            BC_data.df[BC_data.df$Group_nu=="CT_1","Shannon"])
## 2 ##
wilcox.test(BC_data.df[BC_data.df$Group_nu=="CT00","Shannon"],
            BC_data.df[BC_data.df$Group_nu=="PT_2","Shannon"])

wilcox.test(BC_data.df[BC_data.df$Group_nu=="CT00","Shannon"],
            BC_data.df[BC_data.df$Group_nu=="CT_2","Shannon"])
## 3 ##
wilcox.test(BC_data.df[BC_data.df$Group_nu=="CT00","Shannon"],
            BC_data.df[BC_data.df$Group_nu=="PT_3","Shannon"])

wilcox.test(BC_data.df[BC_data.df$Group_nu=="CT00","Shannon"],
            BC_data.df[BC_data.df$Group_nu=="CT_3","Shannon"])
## 4 ##
wilcox.test(BC_data.df[BC_data.df$Group_nu=="CT00","Shannon"],
            BC_data.df[BC_data.df$Group_nu=="PT_4","Shannon"])

wilcox.test(BC_data.df[BC_data.df$Group_nu=="CT00","Shannon"],
            BC_data.df[BC_data.df$Group_nu=="CT_4","Shannon"])
## 5 ##
```

```r
wilcox.test(BC_data.df[BC_data.df$Group_nu=="CT00","Shannon"],
            BC_data.df[BC_data.df$Group_nu=="PT_5","Shannon"])

wilcox.test(BC_data.df[BC_data.df$Group_nu=="CT00","Shannon"],
            BC_data.df[BC_data.df$Group_nu=="CT_5","Shannon"])
## 6 ##
wilcox.test(BC_data.df[BC_data.df$Group_nu=="CT00","Shannon"],
            BC_data.df[BC_data.df$Group_nu=="PT_6","Shannon"])

wilcox.test(BC_data.df[BC_data.df$Group_nu=="CT00","Shannon"],
            BC_data.df[BC_data.df$Group_nu=="CT_6","Shannon"])
## 7 ##
wilcox.test(BC_data.df[BC_data.df$Group_nu=="CT00","Shannon"],
            BC_data.df[BC_data.df$Group_nu=="PCe_6","Shannon"])

wilcox.test(BC_data.df[BC_data.df$Group_nu=="CT00","Shannon"],
            BC_data.df[BC_data.df$Group_nu=="CCe_6","Shannon"])

gm_mean = function(x, na.rm=TRUE){
  exp(sum(log(x[x > 0]), na.rm=na.rm) / length(x))
}

## CT1 vs PT1 ##
TP1_map = meta_data[which(meta_data$Group_3 %in% c("CT_1","PT_1")),]
TP1_cnts = raw_cnts[,rownames(TP1_map)]
TP1_cnts_90 = TP1_cnts[apply(TP1_cnts>0,1,sum)>=round(ncol(TP1_cnts)*0.1),]

TP1_colData = data.frame(TP1_map)
TP1_dds = DESeqDataSetFromMatrix(TP1_cnts_90,TP1_colData,formula(~ Group_3))
TP1_geoMeans = apply(counts(TP1_dds), 1, gm_mean)
TP1_ddsSF = estimateSizeFactors(TP1_dds, geoMeans=TP1_geoMeans)
TP1_ddsDE = DESeq(TP1_ddsSF,fitType="local")
TP1_res = results(TP1_ddsDE,contrast=c("Group_3","CT_1","PT_1"))
TP1_res_noNa = TP1_res[complete.cases(TP1_res),]
nrow(TP1_res_noNa[TP1_res_noNa$padj<0.1,])

## CT2 vs PT2 ##
TP2_map = meta_data[which(meta_data$Group_3 %in% c("CT_2","PT_2")),]
TP2_cnts = raw_cnts[,rownames(TP2_map)]
TP2_cnts_90 = TP2_cnts[apply(TP2_cnts>0,1,sum)>=round(ncol(TP2_cnts)*0.1),]

TP2_colData = data.frame(TP2_map)
TP2_dds = DESeqDataSetFromMatrix(TP2_cnts_90,TP2_colData,formula(~ Group_3))
TP2_geoMeans = apply(counts(TP2_dds), 1, gm_mean)
TP2_ddsSF = estimateSizeFactors(TP2_dds, geoMeans=TP2_geoMeans)
TP2_ddsDE = DESeq(TP2_ddsSF,fitType="local")
TP2_res = results(TP2_ddsDE,contrast=c("Group_3","CT_2","PT_2"))
TP2_res_noNa = TP2_res[complete.cases(TP2_res),]
```

```
## CT3 vs PT3 ##
TP3_map = meta_data[which(meta_data$Group_3 %in% c("CT_3","PT_3")),]
TP3_cnts = raw_cnts[,rownames(TP3_map)]
TP3_cnts_90 = TP3_cnts[apply(TP3_cnts>0,1,sum)>=round(ncol(TP3_cnts)*0.1),]

TP3_colData = data.frame(TP3_map)
TP3_dds = DESeqDataSetFromMatrix(TP3_cnts_90,TP3_colData,formula(~ Group_3))
TP3_geoMeans = apply(counts(TP3_dds), 1, gm_mean)
TP3_ddsSF = estimateSizeFactors(TP3_dds, geoMeans=TP3_geoMeans)
TP3_ddsDE = DESeq(TP3_ddsSF,fitType="local")
TP3_res = results(TP3_ddsDE,contrast=c("Group_3","CT_3","PT_3"))
TP3_res_noNa = TP3_res[complete.cases(TP3_res),]
nrow(TP3_res_noNa[TP3_res_noNa$padj<0.1,])

## CT4 vs PT4 ##

TP4_map = meta_data[which(meta_data$Group_3 %in% c("CT_4","PT_4")),]
TP4_cnts = raw_cnts[,rownames(TP4_map)]
TP4_cnts_90 = TP4_cnts[apply(TP4_cnts>0,1,sum)>=round(ncol(TP4_cnts)*0.1),]

TP4_colData = data.frame(TP4_map)
TP4_dds = DESeqDataSetFromMatrix(TP4_cnts_90,TP4_colData,formula(~ Group_3))
TP4_geoMeans = apply(counts(TP4_dds), 1, gm_mean)
TP4_ddsSF = estimateSizeFactors(TP4_dds, geoMeans=TP4_geoMeans)
TP4_ddsDE = DESeq(TP4_ddsSF,fitType="local")
TP4_res = results(TP4_ddsDE,contrast=c("Group_3","CT_4","PT_4"))
TP4_res_noNa = TP4_res[complete.cases(TP4_res),]
nrow(TP4_res_noNa[TP4_res_noNa$padj<0.05,])
TP4_sig = TP4_res_noNa[TP4_res_noNa$padj<0.05,]
TP4_tw = data.frame(TP4_res_noNa[TP4_res_noNa$padj<0.05,])
TP4_tw = data.frame(taxa[rownames(TP4_tw),],TP4_tw)

TP4_toplot = data.frame(TP4_res_noNa)
TP4_s = norm_means[rownames(TP4_toplot)]
TP4_toplot$size = TP4_s
TP4_toplot = TP4_toplot[order(TP4_toplot$size,decreasing=FALSE),]
TP4_sizes = all_size[rownames(TP4_toplot)]



TP4_f = as.character(droplevels(taxa[rownames(TP4_toplot),"Family"]))
TP4_fs = as.character(droplevels(taxa[rownames(TP4_sig),"Family"]))
TP4_f = gsub("Candidatus_Saccharibacteria_unclassified","Other",TP4_f)
TP4_f = gsub("Deferribacteraceae","Other",TP4_f)
TP4_f = gsub("Erysipelotrichaceae","Other",TP4_f)

table(levels(as.factor(TP4_f)) %in% family_colors$family)
table(levels(as.factor(TP4_fs)) %in% family_colors$family)
```

```
TP4_c = as.character(family_colors[levels(as.factor(TP4_f)),"colors"])


TP4_volcano_plot = ggplot(data = TP4_toplot,aes(x =
TP4_toplot$log2FoldChange,y = -
log10(TP4_toplot$padj),size=as.factor(TP4_sizes),color=TP4_f))+

geom_point()+geom_hline(yintercept=1.30103)+scale_color_manual("Family",value
s=TP4_c)+theme_classic()+scale_size_manual(values=c(5,7,10))
TP4_volcano_plot


TP4_meta = meta_data[meta_data$Group_4 %in%
c("Pre_treatment_1","Pre_treatment_2","Post_AB","CT_4","PT_4"),]
TP4_tp = data.frame(t(cnts_filt_prop[,rownames(TP4_meta)]),TP4_meta)

for (seqid in rownames(TP4_tw)){
  print(ggplot(data = TP4_tp, aes(x = Group_4, y = get(seqid), fill =
Group_3))+geom_boxplot()+ylab(seqid)+theme_classic())

}

TP4_upinCT4 = TP4_tw[TP4_tw$log2FoldChange>0,]
TP4_upinPT4 = TP4_tw[TP4_tw$log2FoldChange<0,]

## CT4 vs Pre_treatment_2
TP4_map = meta_data[which(meta_data$Group_3 %in%
c("CT_4","Pre_treatment_2")),]
TP4_cnts = raw_cnts[,rownames(TP4_map)]
TP4_cnts_90 = TP4_cnts[apply(TP4_cnts>0,1,sum)>=round(ncol(TP4_cnts)*0.1),]
TP4_colData = data.frame(TP4_map)
TP4_dds = DESeqDataSetFromMatrix(TP4_cnts_90,TP4_colData,formula(~ Group_3))
TP4_geoMeans = apply(counts(TP4_dds), 1, gm_mean)
TP4_ddsSF = estimateSizeFactors(TP4_dds, geoMeans=TP4_geoMeans)
TP4_ddsDE = DESeq(TP4_ddsSF,fitType="local")
TP4_res = results(TP4_ddsDE,contrast=c("Group_3","CT_4","Pre_treatment_2"))
TP4_res_noNa = TP4_res[complete.cases(TP4_res),]
TP4_sig = TP4_res_noNa[TP4_res_noNa$padj<0.05,]
TP4_upinCT4_sigPreT2 = TP4_upinCT4[rownames(TP4_upinCT4) %in%
rownames(TP4_sig),]

## PT4 vs Pre_treatment_2
TP4_map = meta_data[which(meta_data$Group_3 %in%
c("PT_4","Pre_treatment_2")),]
TP4_cnts = raw_cnts[,rownames(TP4_map)]
TP4_cnts_90 = TP4_cnts[apply(TP4_cnts>0,1,sum)>=round(ncol(TP4_cnts)*0.1),]
TP4_colData = data.frame(TP4_map)
TP4_dds = DESeqDataSetFromMatrix(TP4_cnts_90,TP4_colData,formula(~ Group_3))
TP4_geoMeans = apply(counts(TP4_dds), 1, gm_mean)
```

```
TP4_ddsSF = estimateSizeFactors(TP4_dds, geoMeans=TP4_geoMeans)
TP4_ddsDE = DESeq(TP4_ddsSF,fitType="local")
TP4_res = results(TP4_ddsDE,contrast=c("Group_3","PT_4","Pre_treatment_2"))
TP4_res_noNa = TP4_res[complete.cases(TP4_res),]
TP4_sig = TP4_res_noNa[TP4_res_noNa$padj<0.05,]
TP4_upinPT4_sigPreT2 = TP4_upinPT4[rownames(TP4_upinPT4) %in%
rownames(TP4_sig),]

preT2 = c(rownames(TP4_upinPT4_sigPreT2),rownames(TP4_upinCT4_sigPreT2))

TP4_meta = meta_data[meta_data$Group_4 %in%
c("Pre_treatment_1","Pre_treatment_2","Post_AB","CT_4","PT_4"),]
TP4_tp = data.frame(t(cnts_filt_prop[,rownames(TP4_meta)]),TP4_meta)

TP4_tw2 = TP4_tw[preT2,]

for (seqid in rownames(TP4_tw2)){
  print(ggplot(data = TP4_tp, aes(x = Group_4, y = get(seqid), fill =
Group_4))+geom_boxplot()+ylab(seqid)+theme_classic())

}



## CT5 vs PT5 ##
TP5_map = meta_data[which(meta_data$Group_4 %in% c("CT_5","PT_5")),]
TP5_cnts = raw_cnts[,rownames(TP5_map)]
TP5_cnts_90 = TP5_cnts[apply(TP5_cnts>0,1,sum)>=round(ncol(TP5_cnts)*0.1),]

TP5_colData = data.frame(TP5_map)
TP5_dds = DESeqDataSetFromMatrix(TP5_cnts_90,TP5_colData,formula(~ Group_4))
TP5_geoMeans = apply(counts(TP5_dds), 1, gm_mean)
TP5_ddsSF = estimateSizeFactors(TP5_dds, geoMeans=TP5_geoMeans)
TP5_ddsDE = DESeq(TP5_ddsSF,fitType="local")
TP5_res = results(TP5_ddsDE,contrast=c("Group_4","CT_5","PT_5"))
TP5_res_noNa = TP5_res[complete.cases(TP5_res),]
nrow(TP5_res_noNa[TP5_res_noNa$padj<0.1,])
TP5_sig = TP5_res_noNa[TP5_res_noNa$padj<0.05,]
TP5_tw = data.frame(TP5_res_noNa[TP5_res_noNa$padj<0.05,])
TP5_tw = data.frame(taxa[rownames(TP5_tw),],TP5_tw)
TP5_toplot = data.frame(TP5_res_noNa)

## Volcano plot ##
TP5_s = norm_means[rownames(TP5_toplot)]
TP5_toplot$size = TP5_s
TP5_toplot = TP5_toplot[order(TP5_toplot$size,decreasing=FALSE),]
TP5_sizes = all_size[rownames(TP5_toplot)]



TP5_f = as.character(droplevels(taxa[rownames(TP5_toplot),"Family"]))
```

```
TP5_fs = as.character(droplevels(taxa[rownames(TP5_sig),"Family"]))
TP5_f = gsub("Candidatus_Saccharibacteria_unclassified","Other",TP5_f)
TP5_f = gsub("Deferribacteraceae","Other",TP5_f)
TP5_f = gsub("Erysipelotrichaceae","Other",TP5_f)

table(levels(as.factor(TP5_f)) %in% family_colors$family)
table(levels(as.factor(TP5_fs)) %in% family_colors$family)

TP5_c = as.character(family_colors[levels(as.factor(TP5_f)),"colors"])

TP5_volcano_plot = ggplot(data = TP5_toplot,aes(x =
TP5_toplot$log2FoldChange,y = -
log10(TP5_toplot$padj),size=as.factor(TP5_sizes),color=TP5_f))+

geom_point()+geom_hline(yintercept=1.30103)+scale_color_manual("Family",value
s=TP5_c)+theme_classic()+scale_size_manual(values=c(5,7,10))
TP5_volcano_plot



TP5_meta = meta_data[meta_data$Group_4 %in%
c("Pre_treatment_1","Pre_treatment_2","Post_AB","CT_5","PT_5"),]
TP5_tp = data.frame(t(cnts_filt_prop[,rownames(TP5_meta)]),TP5_meta)

for (seqid in rownames(TP5_tw)){
  print(ggplot(data = TP5_tp, aes(x = Group_4, y = get(seqid), fill =
Group_4))+geom_boxplot()+ylab(seqid)+theme_classic())

}

TP5_upinCT5 = TP5_tw[TP5_tw$log2FoldChange>0,]
TP5_upinPT5 = TP5_tw[TP5_tw$log2FoldChange<0,]

## CT5 vs Pre_treatment_2
TP5_map = meta_data[which(meta_data$Group_3 %in%
c("CT_5","Pre_treatment_2")),]
TP5_cnts = raw_cnts[,rownames(TP5_map)]
TP5_cnts_90 = TP5_cnts[apply(TP5_cnts>0,1,sum)>=round(ncol(TP5_cnts)*0.1),]
TP5_colData = data.frame(TP5_map)
TP5_dds = DESeqDataSetFromMatrix(TP5_cnts_90,TP5_colData,formula(~ Group_3))
TP5_geoMeans = apply(counts(TP5_dds), 1, gm_mean)
TP5_ddsSF = estimateSizeFactors(TP5_dds, geoMeans=TP5_geoMeans)
TP5_ddsDE = DESeq(TP5_ddsSF,fitType="local")
TP5_res = results(TP5_ddsDE,contrast=c("Group_3","CT_5","Pre_treatment_2"))
TP5_res_noNa = TP5_res[complete.cases(TP5_res),]
TP5_sig = TP5_res_noNa[TP5_res_noNa$padj<0.05,]
TP5_upinCT5_sigPreT2 = TP5_upinCT5[rownames(TP5_upinCT5) %in%
rownames(TP5_sig),]

## PT5 vs Pre_treatment_2
```

```
TP5_map = meta_data[which(meta_data$Group_3 %in%
c("PT_5","Pre_treatment_2")),]
TP5_cnts = raw_cnts[,rownames(TP5_map)]
TP5_cnts_90 = TP5_cnts[apply(TP5_cnts>0,1,sum)>=round(ncol(TP5_cnts)*0.1),]
TP5_colData = data.frame(TP5_map)
TP5_dds = DESeqDataSetFromMatrix(TP5_cnts_90,TP5_colData,formula(~ Group_3))
TP5_geoMeans = apply(counts(TP5_dds), 1, gm_mean)
TP5_ddsSF = estimateSizeFactors(TP5_dds, geoMeans=TP5_geoMeans)
TP5_ddsDE = DESeq(TP5_ddsSF,fitType="local")
TP5_res = results(TP5_ddsDE,contrast=c("Group_3","PT_5","Pre_treatment_2"))
TP5_res_noNa = TP5_res[complete.cases(TP5_res),]
TP5_sig = TP5_res_noNa[TP5_res_noNa$padj<0.05,]
TP5_upinPT5_sigPreT2 = TP5_upinPT5[rownames(TP5_upinPT5) %in%
rownames(TP5_sig),]

preT2 = c(rownames(TP5_upinPT5_sigPreT2),rownames(TP5_upinCT5_sigPreT2))

TP5_meta = meta_data[meta_data$Group_4 %in%
c("Pre_treatment_1","Pre_treatment_2","Post_AB","CT_5","PT_5"),]
TP5_tp = data.frame(t(cnts_filt_prop[,rownames(TP5_meta)]),TP5_meta)

TP5_tw2 = TP5_tw[preT2,]

for (seqid in rownames(TP5_tw2)){
  print(ggplot(data = TP5_tp, aes(x = Group_4, y = get(seqid), fill =
Group_4))+geom_boxplot()+ylab(seqid)+theme_classic())

}


## CT6 vs PT6 ##
TP6_map = meta_data[which(meta_data$Group_4 %in% c("CT_6","PT_6")),]
TP6_cnts = raw_cnts[,rownames(TP6_map)]
TP6_cnts_90 = TP6_cnts[apply(TP6_cnts>0,1,sum)>=round(ncol(TP6_cnts)*0.1),]

TP6_colData = data.frame(TP6_map)
TP6_dds = DESeqDataSetFromMatrix(TP6_cnts_90,TP6_colData,formula(~ Group_4))
TP6_geoMeans = apply(counts(TP6_dds), 1, gm_mean)
TP6_ddsSF = estimateSizeFactors(TP6_dds, geoMeans=TP6_geoMeans)
TP6_ddsDE = DESeq(TP6_ddsSF,fitType="local")
TP6_res = results(TP6_ddsDE,contrast=c("Group_4","CT_6","PT_6"))
TP6_res_noNa = TP6_res[complete.cases(TP6_res),]
nrow(TP6_res_noNa[TP6_res_noNa$padj<0.1,])
TP6_sig = TP6_res_noNa[TP6_res_noNa$padj<0.05,]
TP6_tw = data.frame(TP6_res_noNa[TP6_res_noNa$padj<0.05,])
TP6_tw = data.frame(taxa[rownames(TP6_tw),],TP6_tw)
TP6_toplot = data.frame(TP6_res_noNa)

TP6_s = norm_means[rownames(TP6_toplot)]
```

```
TP6_toplot$size = TP6_s
TP6_toplot = TP6_toplot[order(TP6_toplot$size,decreasing=FALSE),]
TP6_sizes = all_size[rownames(TP6_toplot)]


TP6_f = as.character(droplevels(taxa[rownames(TP6_toplot),"Family"]))
TP6_fs = as.character(droplevels(taxa[rownames(TP6_sig),"Family"]))
TP6_f = gsub("Candidatus_Saccharibacteria_unclassified","Other",TP6_f)
TP6_f = gsub("Deferribacteraceae","Other",TP6_f)
TP6_f = gsub("Erysipelotrichaceae","Other",TP6_f)

table(levels(as.factor(TP6_f)) %in% family_colors$family)
table(levels(as.factor(TP6_fs)) %in% family_colors$family)

TP6_c = as.character(family_colors[levels(as.factor(TP6_f)),"colors"])

TP6_volcano_plot = ggplot(data = TP6_toplot,aes(x =
TP6_toplot$log2FoldChange,y = -
log10(TP6_toplot$padj),size=as.factor(TP6_sizes),color=TP6_f))+

geom_point()+geom_hline(yintercept=1.30103)+scale_color_manual("Family",value
s=TP6_c)+theme_classic()+scale_size_manual(values=c(7,10))
TP6_volcano_plot


TP6_meta = meta_data[meta_data$Group_4 %in%
c("Pre_treatment_1","Pre_treatment_2","Post_AB","CT_6","PT_6"),]
TP6_tp = data.frame(t(cnts_filt_prop[,rownames(TP6_meta)]),TP6_meta)

for (seqid in rownames(TP6_tw)){
  print(ggplot(data = TP6_tp, aes(x = Group_4, y = get(seqid), fill =
Group_4))+geom_boxplot()+ylab(seqid)+theme_classic())

}


TP6_upinCT6 = TP6_tw[TP6_tw$log2FoldChange>0,]
TP6_upinPT6 = TP6_tw[TP6_tw$log2FoldChange<0,]

## CT5 vs Pre_treatment_2
TP6_map = meta_data[which(meta_data$Group_3 %in%
c("CT_6","Pre_treatment_2")),]
TP6_cnts = raw_cnts[,rownames(TP6_map)]
TP6_cnts_90 = TP6_cnts[apply(TP6_cnts>0,1,sum)>=round(ncol(TP6_cnts)*0.1),]
TP6_colData = data.frame(TP6_map)
TP6_dds = DESeqDataSetFromMatrix(TP6_cnts_90,TP6_colData,formula(~ Group_3))
TP6_geoMeans = apply(counts(TP6_dds), 1, gm_mean)
TP6_ddsSF = estimateSizeFactors(TP6_dds, geoMeans=TP6_geoMeans)
TP6_ddsDE = DESeq(TP6_ddsSF,fitType="local")
```

```
TP6_res = results(TP6_ddsDE,contrast=c("Group_3","CT_6","Pre_treatment_2"))
TP6_res_noNa = TP6_res[complete.cases(TP6_res),]
TP6_sig = TP6_res_noNa[TP6_res_noNa$padj<0.05,]
TP6_upinCT6_sigPreT2 = TP6_upinCT6[rownames(TP6_upinCT6) %in%
rownames(TP6_sig),]

## PT5 vs Pre_treatment_2
TP6_map = meta_data[which(meta_data$Group_3 %in%
c("PT_6","Pre_treatment_2")),]
TP6_cnts = raw_cnts[,rownames(TP6_map)]
TP6_cnts_90 = TP6_cnts[apply(TP6_cnts>0,1,sum)>=round(ncol(TP6_cnts)*0.1),]
TP6_colData = data.frame(TP6_map)
TP6_dds = DESeqDataSetFromMatrix(TP6_cnts_90,TP6_colData,formula(~ Group_3))
TP6_geoMeans = apply(counts(TP6_dds), 1, gm_mean)
TP6_ddsSF = estimateSizeFactors(TP6_dds, geoMeans=TP6_geoMeans)
TP6_ddsDE = DESeq(TP6_ddsSF,fitType="local")
TP6_res = results(TP6_ddsDE,contrast=c("Group_3","PT_6","Pre_treatment_2"))
TP6_res_noNa = TP6_res[complete.cases(TP6_res),]
TP6_sig = TP6_res_noNa[TP6_res_noNa$padj<0.05,]
TP6_upinPT6_sigPreT2 = TP6_upinPT6[rownames(TP6_upinPT6) %in%
rownames(TP6_sig),]

preT2 = c(rownames(TP6_upinPT6_sigPreT2),rownames(TP6_upinCT6_sigPreT2))

TP6_meta = meta_data[meta_data$Group_4 %in%
c("Pre_treatment_1","Pre_treatment_2","Post_AB","CT_6","PT_6"),]
TP6_tp = data.frame(t(cnts_filt_prop[,rownames(TP6_meta)]),TP6_meta)

TP6_tw2 = TP6_tw[preT2,]

for (seqid in rownames(TP6_tw2)){
  print(ggplot(data = TP6_tp, aes(x = Group_4, y = get(seqid), fill =
Group_4))+geom_boxplot()+ylab(seqid)+theme_classic())

}

## CT7 vs PT7 ##
TP7_map = meta_data[which(meta_data$Group_4 %in% c("CT_7","PT_7")),]
TP7_cnts = raw_cnts[,rownames(TP7_map)]
TP7_cnts_90 = TP7_cnts[apply(TP7_cnts>0,1,sum)>=round(ncol(TP7_cnts)*0.1),]

TP7_colData = data.frame(TP7_map)
TP7_dds = DESeqDataSetFromMatrix(TP7_cnts_90,TP7_colData,formula(~ Group_4))
TP7_geoMeans = apply(counts(TP7_dds), 1, gm_mean)
TP7_ddsSF = estimateSizeFactors(TP7_dds, geoMeans=TP7_geoMeans)
TP7_ddsDE = DESeq(TP7_ddsSF,fitType="local")
TP7_res = results(TP7_ddsDE,contrast=c("Group_4","CT_7","PT_7"))
TP7_res_noNa = TP7_res[complete.cases(TP7_res),]
nrow(TP7_res_noNa[TP7_res_noNa$padj<0.1,])
```

```
TP7_sig = TP7_res_noNa[TP7_res_noNa$padj<0.05,]
TP7_tw = data.frame(TP7_res_noNa[TP7_res_noNa$padj<0.05,])
TP7_tw = data.frame(taxa[rownames(TP7_tw),],TP7_tw)
TP7_toplot = data.frame(TP7_res_noNa)

TP7_s = norm_means[rownames(TP7_toplot)]
TP7_toplot$size = TP7_s
TP7_toplot = TP7_toplot[order(TP7_toplot$size,decreasing=FALSE),]
TP7_sizes = all_size[rownames(TP7_toplot)]


TP7_f = as.character(droplevels(taxa[rownames(TP7_toplot),"Family"]))
TP7_fs = as.character(droplevels(taxa[rownames(TP7_sig),"Family"]))
TP7_f = gsub("Candidatus_Saccharibacteria_unclassified","Other",TP7_f)
TP7_f = gsub("Erysipelotrichaceae","Other",TP7_f)

table(levels(as.factor(TP7_f)) %in% family_colors$family)
table(levels(as.factor(TP7_fs)) %in% family_colors$family)

TP7_c = as.character(family_colors[levels(as.factor(TP7_f)),"colors"])

TP7_volcano_plot = ggplot(data = TP7_toplot,aes(x =
TP7_toplot$log2FoldChange,y = -
log10(TP7_toplot$padj),size=as.factor(TP7_sizes),color=TP7_f))+

geom_point()+geom_hline(yintercept=1.30103)+scale_color_manual("Family",value
s=TP7_c)+theme_classic()+scale_size_manual(values=c(7,10))
TP7_volcano_plot
```

Virome Analysis

```
library(ggplot2)
library(plyr)
library(reshape)
library(RColorBrewer)
library(xlsx)
library(gridExtra)
library(gtools)

cnts = read.table("virome_raw_counts")
family_con_prop = prop.table(as.matrix(cnts),2)
family_con_prop = family_con_prop * 100

# Establish which low aboundant taxa should be summed into Other
tobesummed = rownames(family_con_prop)[apply(family_con_prop,1,mean)<1]
tobesummed = tobesummed[!tobesummed %in% "NODE_1_length_282958_cov_17.4263"]
tobesummed = tobesummed[!tobesummed %in% "NODE_1852_length_4960_cov_624.798"]
tobesummed = tobesummed[!tobesummed %in% "NODE_67_length_52460_cov_25.1863"]
tobesummed = tobesummed[!tobesummed %in% "NODE_29_length_67264_cov_59.806"]
```

```
tobesummed = tobesummed[!tobesummed %in% "NODE_147_length_37977_cov_94.2795"]
tobesummed = tobesummed[!tobesummed %in% "NODE_882_length_9207_cov_556.169"]



family_prop_tree =
data.frame(as.character(rownames(family_con_prop)),family_con_prop)
colnames(family_prop_tree)[1] = "ID"
family_prop_tree$ID = as.character(family_prop_tree$ID)
family_prop_tree[tobesummed,"ID"] = "Other"
family_con_prop_tree_summed = ddply(family_prop_tree,"ID",numcolwise(sum))
nrow(family_con_prop_tree_summed)

rownames(family_con_prop_tree_summed) =family_con_prop_tree_summed$ID
new_ids = c()
key = c()
for (number in 1:(nrow(family_con_prop_tree_summed)-1)){
  id = paste("APC_pVirus",number,sep="_")
  new_ids = c(new_ids,id)
  key[id] = rownames(family_con_prop_tree_summed)[number]
}
new_ids = c(new_ids,"Other")
key = data.frame(key)
family_con_prop_tree_summed$ID = new_ids
rownames(family_con_prop_tree_summed) = new_ids

# Prepare ggplot2 object and relevel
data.df = melt(family_con_prop_tree_summed,id.vars = "ID")
data.df$ID = factor(data.df$ID,levels=mixedsort(data.df$ID))
M1 = data.df[grep("m1",data.df$variable),]
M2 = data.df[grep("m2",data.df$variable),]
M1$variable = droplevels(M1$variable)
M2$variable = droplevels(M2$variable)

M1$variable = factor(M1$variable,levels =
c("V1.m1","V2.m1","V3.m1","V4.m1","V5.m1","V6.m1","V7.m1","V8.m1","V9.m1","V1
0.m1","V11.m1","V12.m1","V13.m1","V14.m1","V15.m1"))
M2$variable = factor(M2$variable,levels =
c("V1.m2","V2.m2","V3.m2","V4C.m2","V4P.m2","V5C.m2","V5P.m2","V6C.m2","V6P.m
2","V7C.m2","V7P.m2"))

getPalette = colorRampPalette(brewer.pal(9, "Set1"))

cols = getPalette(nrow(family_con_prop_tree_summed))
cols = cols[1:length(cols)-1]
cols = sample(cols)
cols = c(cols,"#999999")



#Mouse 1
```

```
M1c = data.df[data.df$variable %in%
c("V1.m1","V2.m1","V3.m1","V5.m1","V6.m1","V8.m1","V10.m1","V12.m1","V14.m1")
,]
M1p = data.df[data.df$variable %in%
c("V1.m1","V2.m1","V3.m1","V4.m1","V7.m1","V9.m1","V11.m1","V13.m1","V15.m1")
,]
M1c$variable = droplevels(M1c$variable)
M1p$variable = droplevels(M1p$variable)
M1c$variable = factor(M1c$variable,levels =
c("V1.m1","V2.m1","V3.m1","V5.m1","V6.m1","V8.m1","V10.m1","V12.m1","V14.m1")
)
M1p$variable = factor(M1p$variable,levels =
c("V1.m1","V2.m1","V3.m1","V4.m1","V7.m1","V9.m1","V11.m1","V13.m1","V15.m1")
)

m1c_plot = ggplot(M1c, aes(x = variable, y = value, fill = ID)) +
geom_bar(position = "fill",stat = "identity",width=0.9)+
  theme_classic()+ggtitle("Mouse 1 control assembly based
virome")+theme(axis.text.x = element_text(angle = 45, hjust = 1))+
  scale_fill_manual(values=cols)
m1p_plot = ggplot(M1p, aes(x = variable, y = value, fill = ID)) +
geom_bar(position = "fill",stat = "identity",width=0.9)+
  theme_classic()+ggtitle("Mouse 1 phage assembly based
virome")+theme(axis.text.x = element_text(angle = 45, hjust = 1))+
  scale_fill_manual(values=cols)
grid.arrange(m1c_plot,m1p_plot)

## Mouse 2
m2c = data.df[data.df$variable %in%
c("V1.m2","V2.m2","V3.m2","V4C.m2","V5C.m2","V6C.m2","V7C.m2"),]
m2p = data.df[data.df$variable %in%
c("V1.m2","V2.m2","V3.m2","V4P.m2","V5P.m2","V6P.m2","V7P.m2"),]
m2c$variable = droplevels(m2c$variable)
m2p$variable = droplevels(m2p$variable)
m2c$variable = factor(m2c$variable,levels =
c("V1.m2","V2.m2","V3.m2","V4C.m2","V5C.m2","V6C.m2","V7C.m2"))
m2p$variable = factor(m2p$variable,levels =
c("V1.m2","V2.m2","V3.m2","V4P.m2","V5P.m2","V6P.m2","V7P.m2"))

m2c_plot = ggplot(m2c, aes(x = variable, y = value, fill = ID)) +
geom_bar(position = "fill",stat = "identity",width=0.9)+
  theme_classic()+ggtitle("Mouse 2 control assembly based
virome")+theme(axis.text.x = element_text(angle = 45, hjust = 1))+
  scale_fill_manual(values=cols)
m2p_plot = ggplot(m2p, aes(x = variable, y = value, fill = ID)) +
geom_bar(position = "fill",stat = "identity",width=0.9)+
  theme_classic()+ggtitle("Mouse 2 phage assembly based
virome")+theme(axis.text.x = element_text(angle = 45, hjust = 1))+
  scale_fill_manual(values=cols)
grid.arrange(m2c_plot,m2p_plot)
```