Silage Metagenomics

TBD

September 3, 2017

Study: Aer_Comp

```
packageVersion("optparse")
```

```
## [1] '1.4.4'
```

```
packageVersion("phyloseq")
```

```
## [1] '1.19.1'
```

```
packageVersion("ggplot2")
```

```
## [1] '2.2.1'
```

```
packageVersion("plyr")
```

```
## [1] '1.8.4'
```

```
packageVersion("DESeq2")
```

```
## [1] '1.14.1'
```

```
packageVersion("ggthemes")
```

```
## [1] '3.4.0'
```

```
packageVersion("pander")
```

```
## [1] '0.6.1'
```

```
theme_set(theme_bw())
# would set pdf device here if we weren't knitting
```

```
# min # counts/sample filtering relevant for qiime corediv, but not here (at least yet)
# no "sing/doub" filtering
# also, no json conversion!
```

```r
# have I chosen the most appropriate biom file?

# taxon filtering:  min count in min samples
# if the counts for a OTU are <= (max otu count)/max.frac, then discard
# see also the calculation for the minimum number of samples in which this condition must be satisfied
max.frac <- 1000

# rank selected for plotting unless otherwise specified
selected.rank <- "Family"

# taxon filtering:  top N OTUs
# only keep the top N OTUs, which could be of different taxonomic ranks
#   and/or which could all come from a singe higher aggregated rank
#   if OTUs L. acidophilus, L. delbrueckii, L. plantarum, L. brevis,  and L. buchneri all have counts of 2
#   and E. faecalis and Bacteroides succinogenes both have counts of 5,
#   and the top 2 are requested, aggregated genus Lactobacillus won't be reported
the.N <- 10

# at what rank should the results of this filtering be plotted
top.N.plot.rank <- "Genus"

# categorical factor for differential abundance comparison
# add continuous or complex modeling later
exp.factor <- "treatment"

alpha.div.measures <-
  c("Observed", "Chao1", "Shanon", "Simpson", "InvSimpson")
alpha.div.plot.alab.fontsize <- 6
alpha.point.size <- 5
alpha.point.alpha <- 0.7

ordination.dist <- "bray"
ord_meths  <-
  c("DCA", "CCA", "RDA", "DPCoA", "NMDS", "MDS", "PCoA")
# see problems with DPCoA below!
# ord_meths <- setdiff(ord_meths, "DPCoA")

# max significance for reporting and plotting in DESeq2
```

```
DESeq2.alpha <- 0.05

# what two ranks should be used to group the OTUs
higher.rank.name <- "Family"
lower.rank.name <- "Genus"

# what values from DESeq analysis should be plotted
y.val.name <- "log2FoldChange"

volcano.lfc <- 2
volcano.pval <- 0.01
```

```
# experiment <- "Aer_Comp"
# experiment <- "LalStress"
# experiment <- "V-HMC"

experiment <- params$study_name


# THIS WILL DEFINITELY BE DIFFERENT FOR EACH STUDY
# what levels of the exp.factor should be compared?
# may want to do multiple
# numerator... top portion of ratio

# denominator... bottom portion of ratio
if (experiment == "LalStress") {
  constrat.num <- "C"
  constrat.den <- "LB500.LATE"
} else if (experiment == "V-HMC") {
  constrat.num <- "C"
  constrat.den <- "LB"
} else if (experiment == "Aer_Comp") {
  constrat.num <- "C"
  constrat.den <- "S2.NS"
}
```

```
otufile <- paste0('/home/mark/gitstage/uderica/silage/',
                  experiment ,
                  '/Fungi/raw_data_fungi/flashed/extended/flash_trim_cat_pick/otu_table_mc2_w_tax.biom')
```

Warnings from import_biom may have been suppressed!

check the chunk parameters

MAM: I haven't found a good way to summarize this

```
# ow <- options("warn")
# options(warn=1)
# warnlist <- capture.output({otutable <-
#   import_biom(BIOMfilename = otufile,
#               parseFunction = parse_taxonomy_greengenes)})
# table(warnlist)
# options(ow)

otutable <-
  import_biom(BIOMfilename = otufile,
              parseFunction = parse_taxonomy_greengenes)
```

```
mapfile <-
  paste0('/home/mark/gitstage/uderica/silage/',
         experiment ,
         '/Fungi/',
         'map.txt')

mapping <-
  import_qiime_sample_data(mapfilename = mapfile)

phylo <- merge_phyloseq(otutable, mapping)

print(phylo)
```

```
## phyloseq-class experiment-level object
## otu_table()   OTU Table:         [ 525 taxa and 28 samples ]
## sample_data() Sample Data:       [ 28 samples by 8 sample variables ]
## tax_table()   Taxonomy Table:    [ 525 taxa by 14 taxonomic ranks ]
```

```
otu.dat <- otutable@otu_table@.Data
dim(otu.dat)
```

```
## [1] 525  28
```

```
otu.tab.excerpt <- otu.dat[order(rownames(otu.dat)),]
otu.tab.excerpt <- otu.tab.excerpt[,order(as.numeric(colnames(otu.dat)))]
pander(otu.tab.excerpt[1:9,1:9])
```

	657	658	659	660	662	663	664	665	667
AB026015	0	0	0	0	0	0	0	0	0
AB214655	0	0	0	0	0	0	0	0	0
AB237662	1	1	0	2	0	0	0	0	0
AF081468	0	0	0	0	0	0	0	0	0
AF294700	6	5	1	44	0	0	0	2	0
AJ246154	0	0	0	0	0	0	0	0	0
AJ301962	0	0	2	0	0	0	0	0	0
AJ301998	13	0	0	0	0	0	0	0	0
AJ557830	1013	1455	265	2441	167	14	1	5	20

```
unlisted.otu.counts <- unlist(otu.dat)

# histogram, of unaggregated counts, log scale
# would it be OK to filter out taxa with observations that are less than 1/x of the max
# where x is something like 1000?

max.count <- max(unlisted.otu.counts)
min.proposal <- max.count / max.frac

print(max.count)
```

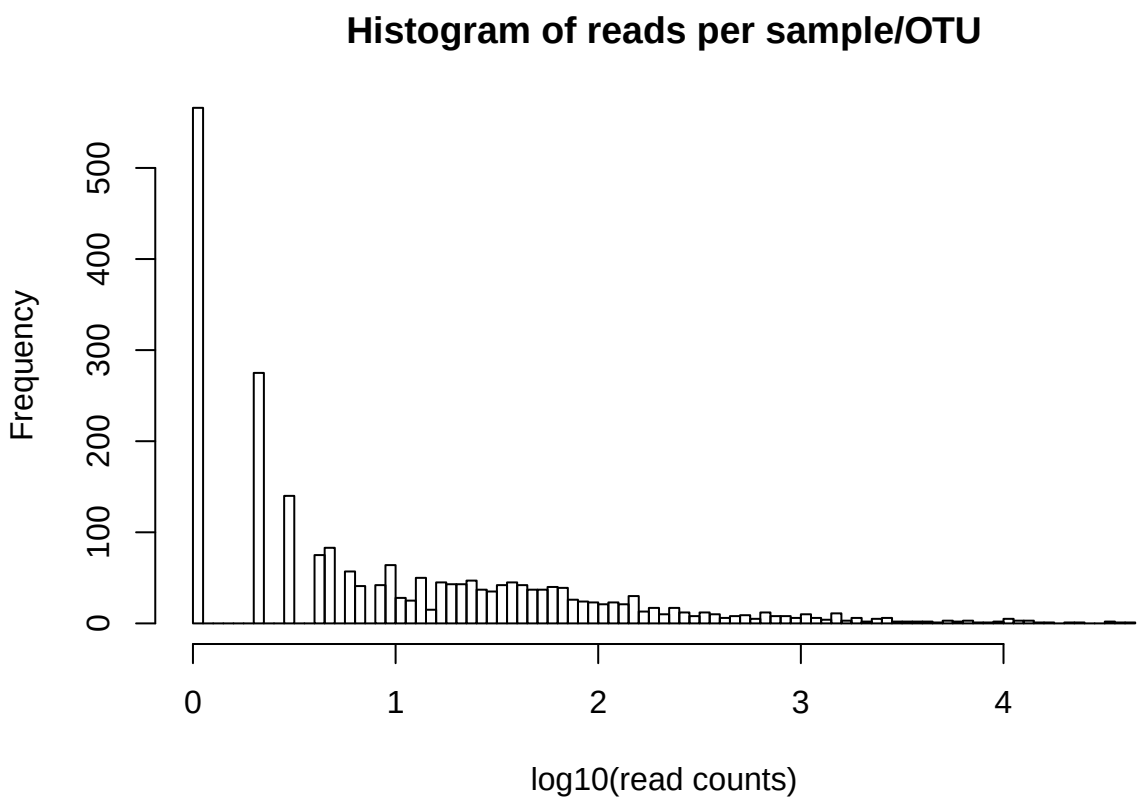
```
## [1] 40260
```

```
print(min.proposal)
```

```
## [1] 40.26
```

```
old.mar <- par("mar")
par("mar" = c(5, 4, 3, 2))
```

```
hist(
  log10(unlisted.otu.counts),
  breaks = 100,
  main = "Histogram of reads per sample/OTU",
  xlab = "log10(read counts)"
)
```




```
par("mar" = old.mar)



otu.rowsums <- rowSums(otu.dat)
otu.rowsums <-
```

```
  cbind.data.frame(names(otu.rowsums), as.numeric(otu.rowsums))
names(otu.rowsums) <- c("OTU", "count.sum")


otu.colsums <- colSums(otu.dat)
otu.colsums <-
  cbind.data.frame(names(otu.colsums), as.numeric(otu.colsums))
names(otu.colsums) <- c("sample", "count.sum")
otu.colsums[] <- lapply(otu.colsums[], as.character)
otu.colsums[] <- lapply(otu.colsums[], as.numeric)

# is "new.cleanup.reference OTU"" a problem?
# might have been mapped to a lineage despite that name
# what about unmapped?
# what has been lost before this point?
#   un-flashed (really un-extended)
#   discarded because of quality by qiime split?
#   do "split" some sequences not make it into the biom file for some reason?
```

```
map.dat <- as.data.frame(mapping@.Data)
names(map.dat) <- mapping@names

exp.design.matrix <- map.dat[, c("X.SampleID", "treatment", "day")]

# for V-HMC
# days of fermentation and hours of air stress
if (sum(grepl(pattern = "h", x = exp.design.matrix$day))) {
  names(exp.design.matrix) <-
    c("SampleID", "treatment", "days.string")

  temp <-
    strsplit(as.character(exp.design.matrix$days.string), "\\+|h")
  temp <- ldply(temp, rbind)
  temp <- as.data.frame(temp)
  temp[] <- lapply(temp[], as.character)
  temp[] <- lapply(temp[], as.numeric)
  names(temp) <- c("days", "hours")
```

```
  temp$hours[is.na(temp$hours)] <- 0
  exp.design.matrix <- cbind.data.frame(exp.design.matrix, temp)

  # exp.design.matrix$days.hours <- temp$days + (temp$hours / 24)

  exp.design.tabulation <-
    table(exp.design.matrix$treatment, exp.design.matrix$days.string)

} else {
  exp.design.tabulation <-
    table(exp.design.matrix$treatment, exp.design.matrix$day)
}


exp.design.tabulation <- as.data.frame.matrix(exp.design.tabulation)
pander(exp.design.tabulation)
```

	0	56
C	4	0
C-3H	0	4
C-3H-LATE	0	4
C-NS	0	4
S2-3H	0	4
S2-3H-LATE	0	4
S2-NS	0	4

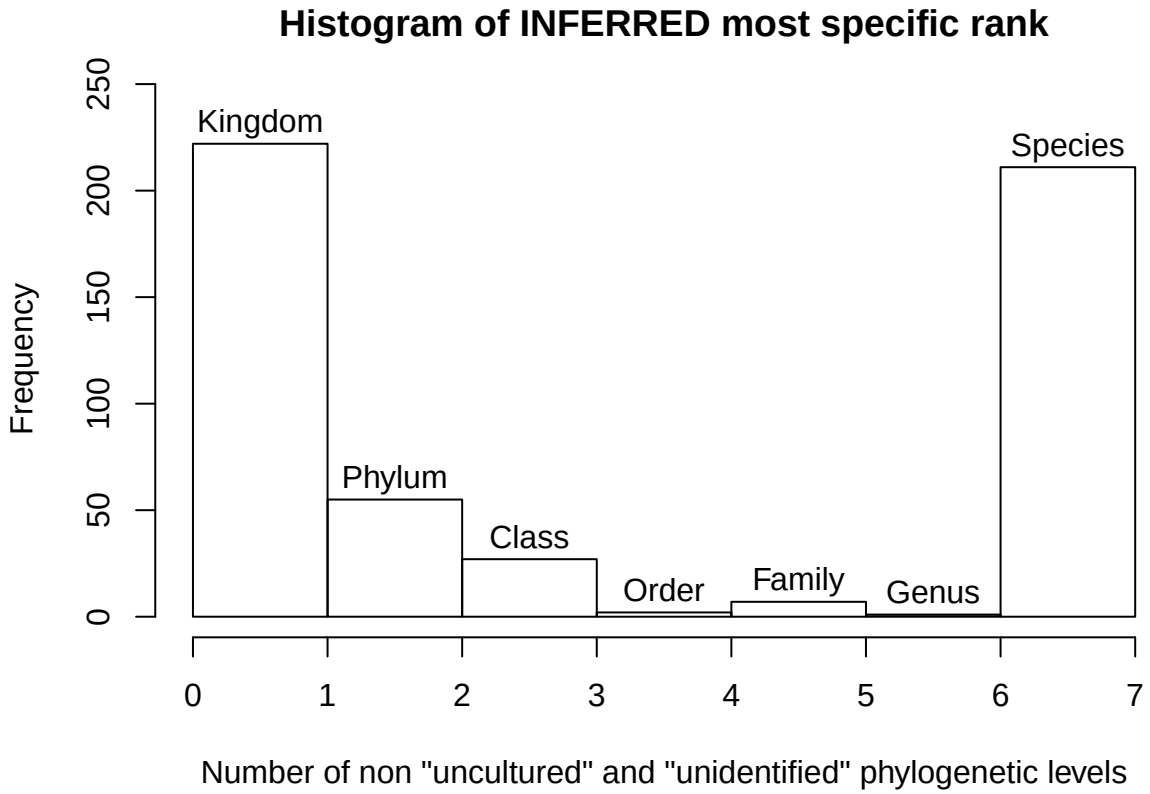
```
# # if col names are all arithmetics and have long decimal components
# names(exp.design.tabulation) <-
#   round(as.numeric(as.character(names(
#     exp.design.tabulation
#   ))), digits = 1)


exp.design.tabulation[exp.design.tabulation == 0] <- NA
temp.values <- unlist(exp.design.tabulation)
temp.flag <- complete.cases(temp.values)
```

```
pander(table(temp.values[temp.flag]))
```

7

```
min.replication <- min(temp.values[temp.flag])
```

```
tax.dat <- as.data.frame(otutable@tax_table)
temp <- as.matrix(tax.dat)
temp[grepl(pattern = "^uncultured", temp)] <- NA
temp[temp == "unidentified"] <- NA
temp <- !is.na(temp)
taxonomic.specificity <- rowSums(temp)
tax.dat$taxonomic.specificity <- taxonomic.specificity

max.specificity <- max(tax.dat$taxonomic.specificity)
temp.frame <-
  cbind.data.frame((1:max.specificity), names(tax.dat)[1:max.specificity])
names(temp.frame) <- c("rank", "label")

max.count <- hist(tax.dat$taxonomic.specificity, plot = FALSE)
max.count <- max.count$counts
max.count <-  max(max.count)

par("mar" = c(5, 4, 3, 2))

hist(
  tax.dat$taxonomic.specificity,
  labels = as.character(temp.frame$label),
  breaks = 0:max.specificity,
  main = "Histogram of INFERRED most specific rank",
  xlab = 'Number of non "uncultured" and "unidentified" phylogenetic levels',
  ylim = c(0, (max.count + 20))
)
```



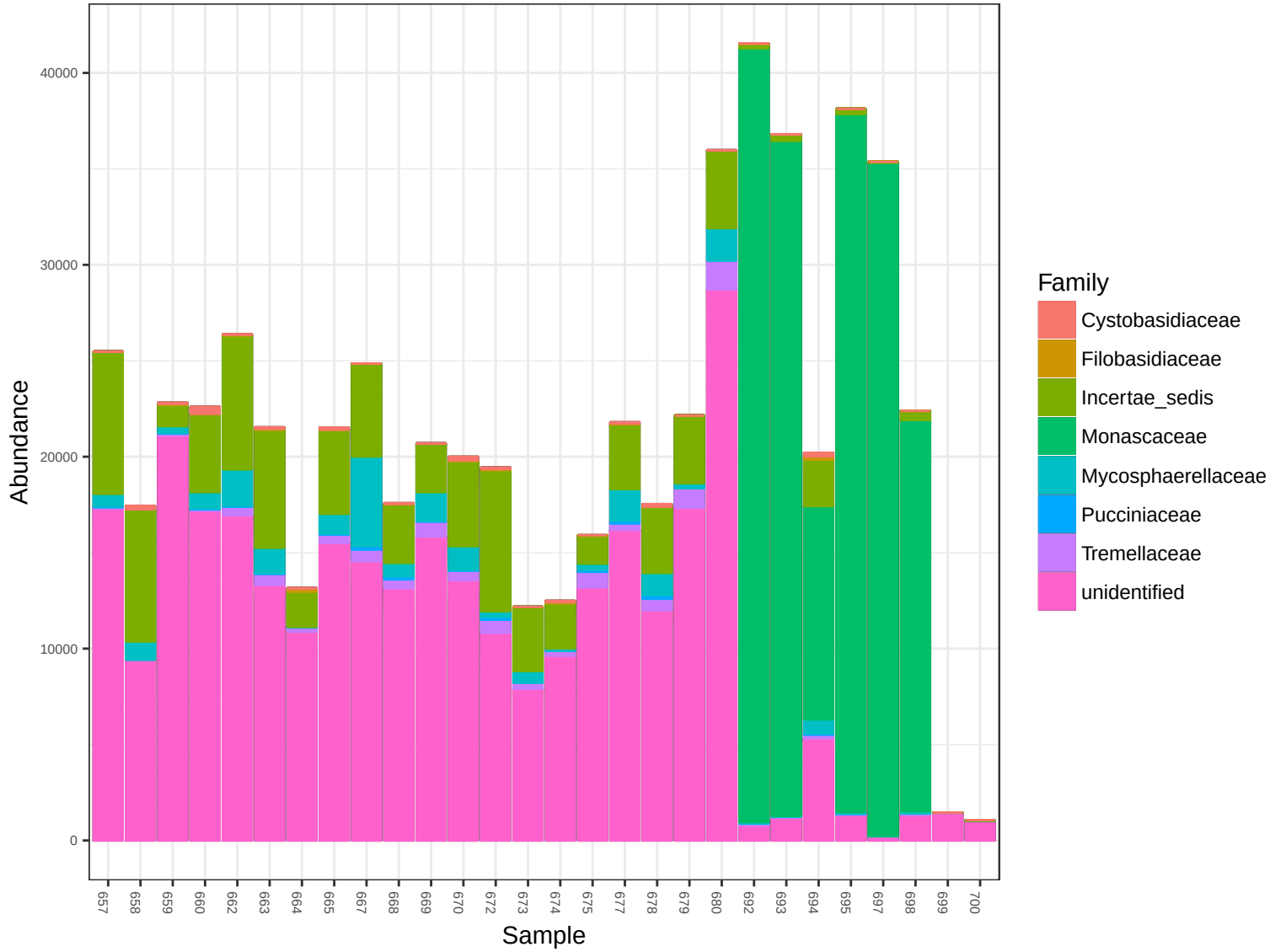

```
par("mar" =  old.mar)
```

```
re.min.filtered <-
  genefilter_sample(phylo,
                    filterfun_sample(function(x)
                      x > min.proposal),
                    A = min.replication)

phylo.k.of.A <- prune_taxa(re.min.filtered, phylo)

print(phylo.k.of.A)
```

```
## phyloseq-class experiment-level object
## otu_table()   OTU Table:         [ 48 taxa and 28 samples ]
```

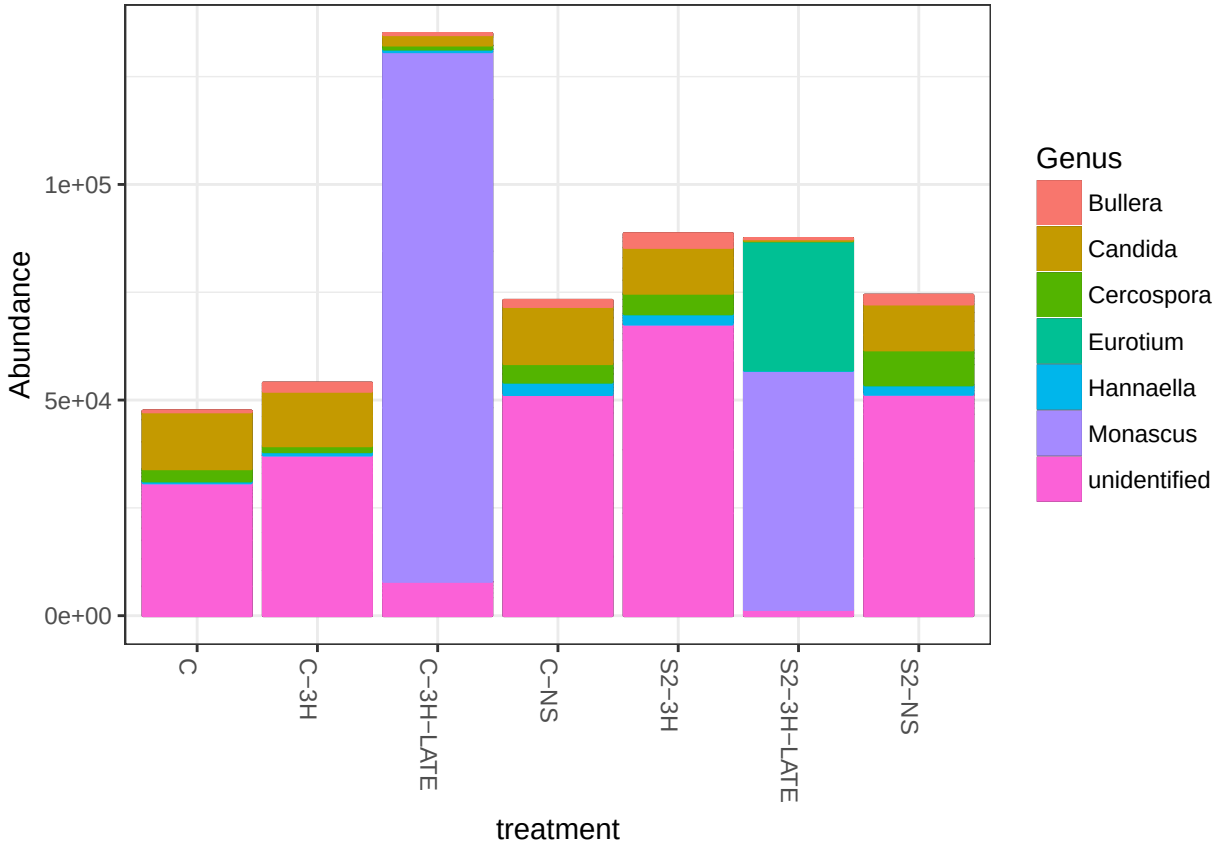
```
## sample_data() Sample Data:       [ 28 samples by 8 sample variables ]
## tax_table()   Taxonomy Table:    [ 48 taxa by 14 taxonomic ranks ]
```

```
# absolute or relative?
# plot bar is absolute reads

# could use + scale_colour_brewer(palette = "Set1"), but that adds in ORANGE separators !?
# set3 is paler but has 4 more levels
# paired also has more levels but is "paired"!
# + scale_colour_manual(values = colorRamps::primary.colors(n = 10))

# ungrouped... one way to make sure that all samples made it to this phase
p <- plot_bar(phylo.k.of.A, fill = "Family")
# to get rid of black dividers
p + geom_bar(aes(color = Family, fill = Family),
             stat = "identity") + theme(axis.text=element_text(size=6))
```



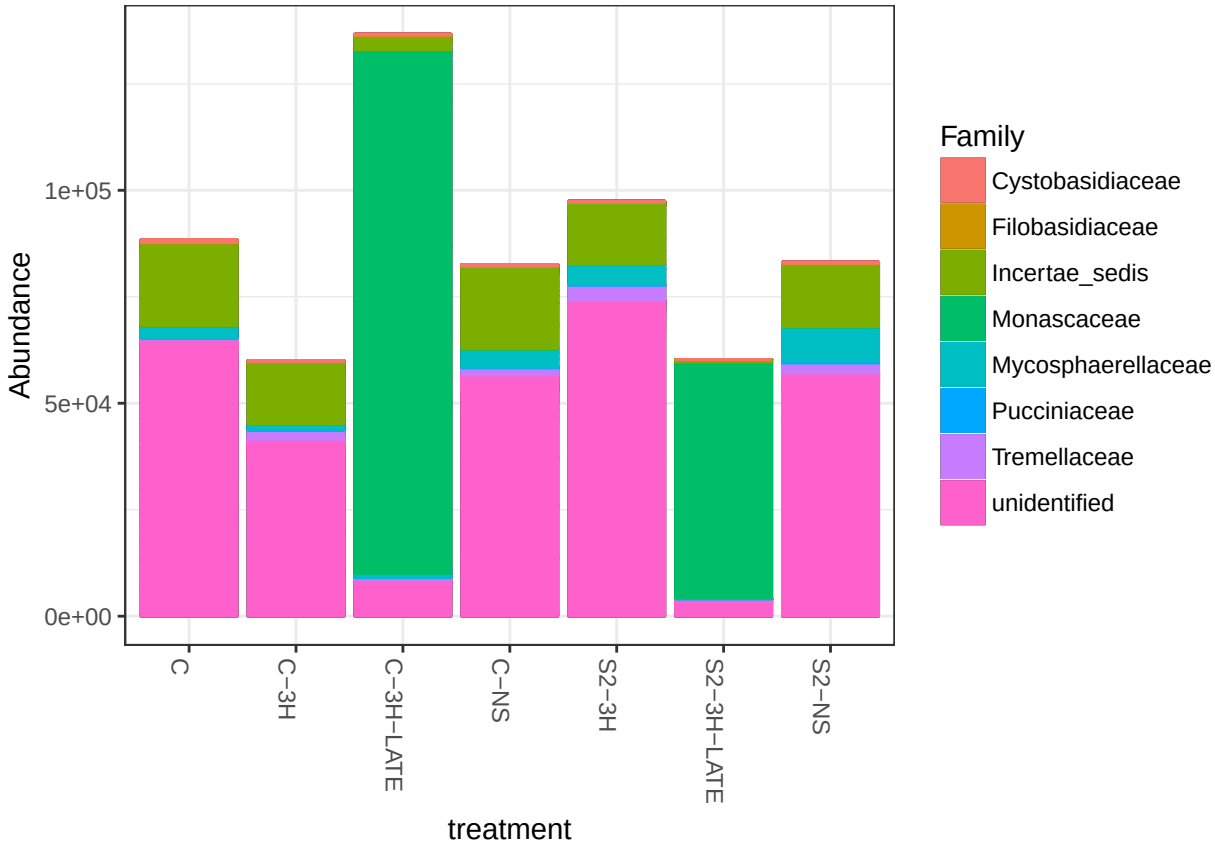

```
taxa.named.vector <- sort(taxa_sums(phylo), TRUE)
taxa.ids <- names(taxa.named.vector)
top.taxa <- taxa.ids[1:the.N]
```

```
phylo.top.N <- prune_taxa(top.taxa, phylo)

p <- plot_bar(phylo.top.N, exp.factor, fill = top.N.plot.rank)
p + geom_bar(aes_(
  color = as.name(top.N.plot.rank),
  fill = as.name(top.N.plot.rank)
), stat = "identity")
```




```
p <- plot_bar(phylo.k.of.A, exp.factor, fill = "Family")

# this gets rid of the black otu separators
```

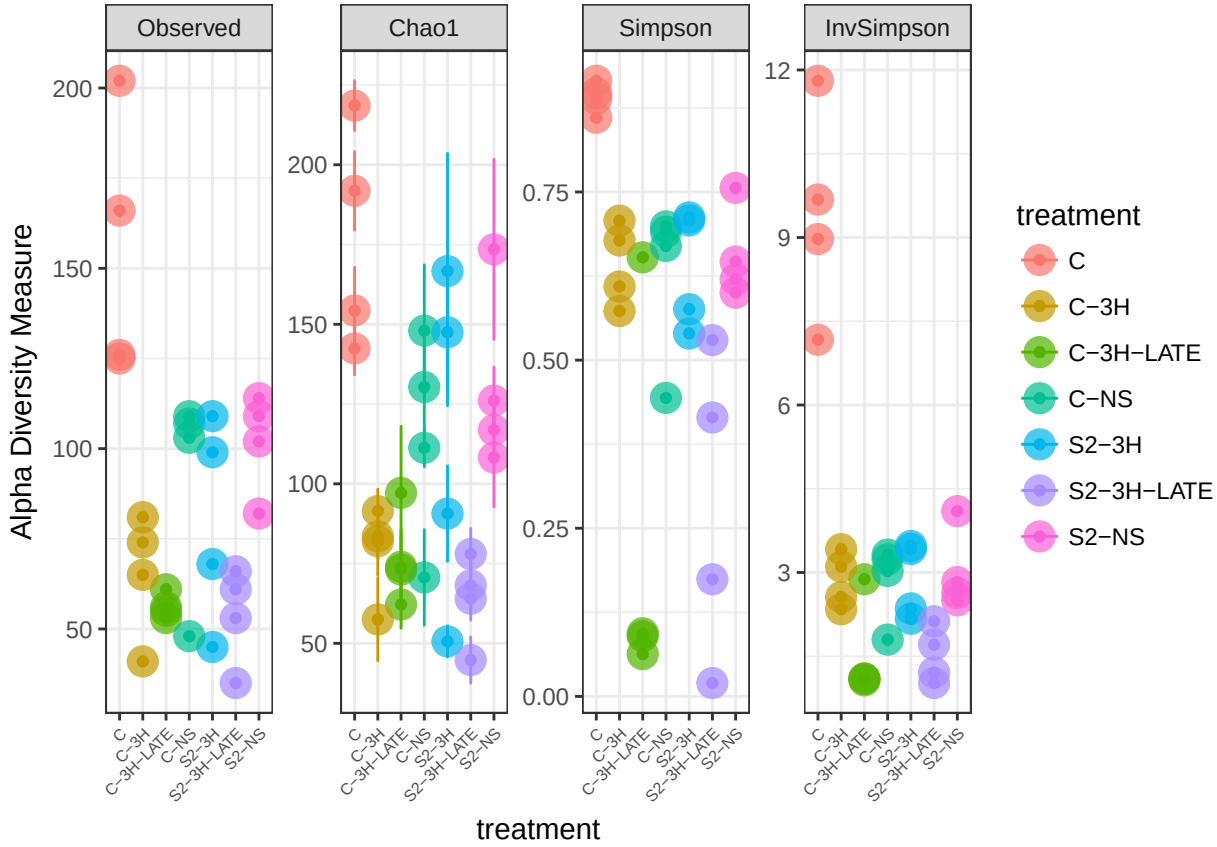
```
# add faceting?
p + geom_bar(aes(color = Family, fill = Family),
             stat = "identity")
```




```
# plot richness without filtering?
p <-
  plot_richness(phylo,
                x = exp.factor,
                color = exp.factor,
                measures = alpha.div.measures)

# parameterize
```

```
p + geom_point(size = alpha.point.size, alpha = alpha.point.alpha) + theme(axis.text.x = element_text(
  angle = 45,
  hjust = 1,
  vjust = 1,
  size = alpha.div.plot.alab.fontsize
))
```

```
## Warning: Removed 84 rows containing missing values (geom_errorbar).
```




```
alpha.diversity <-
  estimate_richness(phylo,
                    measures = alpha.div.measures)
```

```r
alpha.diversity$sample.id.string <-
  as.numeric(sub("^X", "", rownames(alpha.diversity)))
rownames(alpha.diversity) <- NULL
non.sample.cols <-
  sort(setdiff(names(alpha.diversity), "sample.id.string"))
alpha.diversity <-
  alpha.diversity[order(alpha.diversity$sample), c("sample.id.string", non.sample.cols)]
# write.table(alpha.diversity, "safeearly.txt")
# dump to xtable if knitting
```

```r
# messages when including DPCoA
# 1) Species coordinates not found directly in ordination object. Attempting weighted average (`vegan::wascores`)
# 2) non-unique values when setting 'row.names':

# DPCoA = Double Principle Coordinate Analysis using a (corrected, if necessary) phylogenetic/patristic distance between species. The calc

ord_meths <- setdiff(ord_meths, "DPCoA")

plist <- llply(as.list(ord_meths), function(i, phylo.k.of.A, dist) {
  message(i)
  ordi <- ordinate(phylo.k.of.A, method = i, distance = dist)
  plot_ordination(phylo.k.of.A, ordi, "samples", color = exp.factor)
}, phylo.k.of.A, ordination.dist)
```

```
## DCA
```

```
## CCA
```

```
## RDA
```

```
## NMDS
```

```
## Square root transformation
## Wisconsin double standardization
## Run 0 stress 0.07620143
## Run 1 stress 0.07874939
## Run 2 stress 0.07875225
## Run 3 stress 0.07874979
## Run 4 stress 0.09031496
## Run 5 stress 0.07874638
## Run 6 stress 0.07875006
```

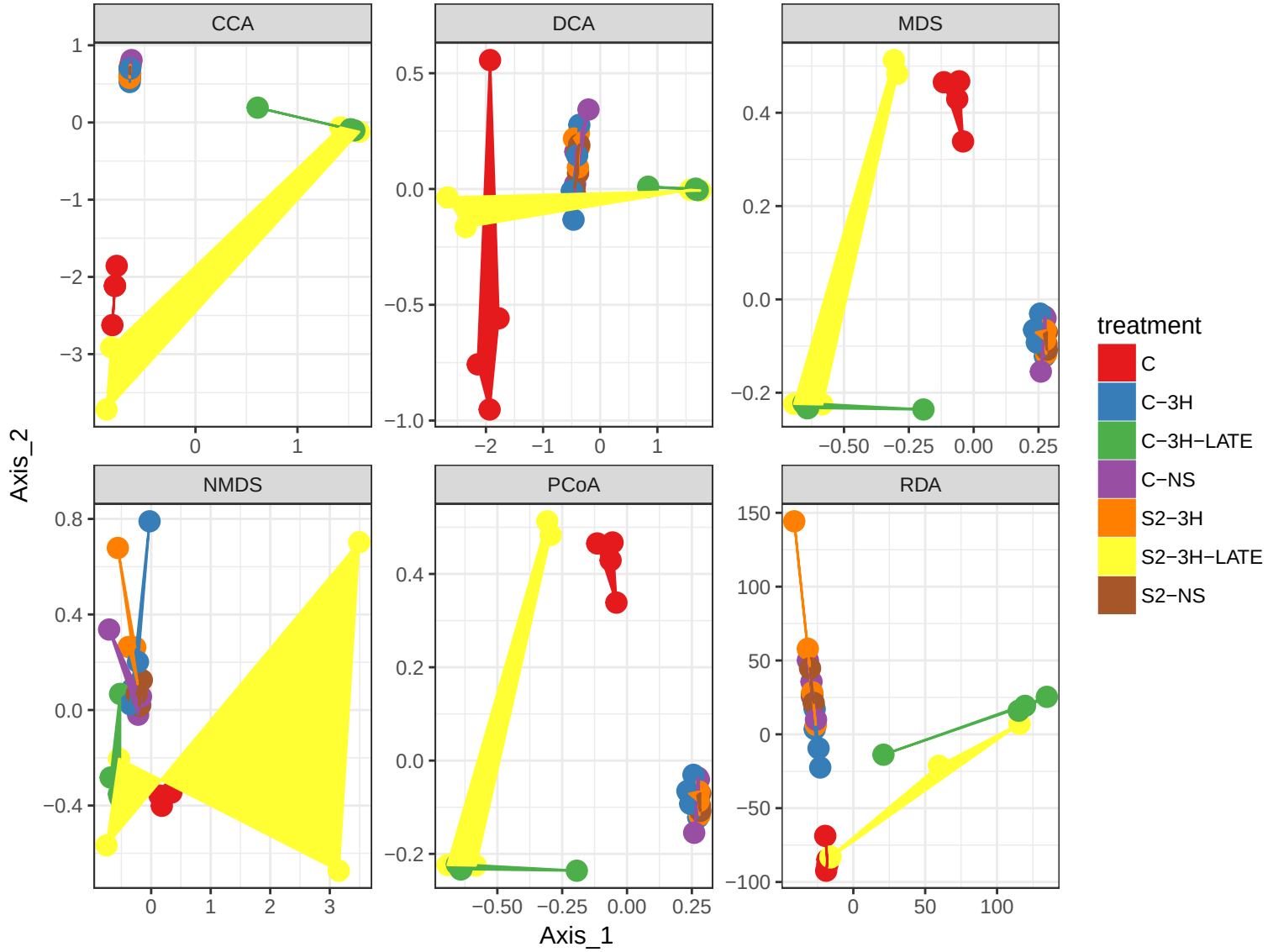
```
## Run 7 stress 0.093516
## Run 8 stress 0.07874922
## Run 9 stress 0.07874997
## Run 10 stress 0.07620188
## ... Procrustes: rmse 0.0008130542  max resid 0.002273205
## ... Similar to previous best
## Run 11 stress 0.07620147
## ... Procrustes: rmse 0.0001158666  max resid 0.0003498789
## ... Similar to previous best
## Run 12 stress 0.07874773
## Run 13 stress 0.07874618
## Run 14 stress 0.101779
## Run 15 stress 0.09453982
## Run 16 stress 0.0787487
## Run 17 stress 0.07874842
## Run 18 stress 0.07874938
## Run 19 stress 0.08957518
## Run 20 stress 0.07620104
## ... New best solution
## ... Procrustes: rmse 0.0006391394  max resid 0.001608052
## ... Similar to previous best
## *** Solution reached
```

```
## MDS
```

```
## PCoA
```

```
names(plist) <- ord_meths


pdataframe <- ldply(plist, function(x) {
  df <- x$data[, 1:2]
  colnames(df) <- c("Axis_1", "Axis_2")
  return(cbind(df, x$data))
})

names(pdataframe)[1] <- "method"


p <- ggplot(pdataframe,
```

```
            aes(Axis_1, Axis_2, color = treatment, fill = treatment))
p <- p + geom_point(size = 4) + geom_polygon()
p <- p + facet_wrap( ~ method, scales = "free")
p <- p + scale_fill_brewer(type = "qual", palette = "Set1")
p <- p + scale_colour_brewer(type = "qual", palette = "Set1")
p
```




```
# user may desire an individual plot on a page by itself
# so print all on a page by themselves?
# how did Erica determine that p = plist[[6]] (MDS) was the best ordination?
```

```
# these analyses WERE being run at different levels of filtering
# initial barplot and alpha diversity at 2 (from Bash script)
# ordination refiltered to 3
# diff abundance back to 2
#
```

```
message(" \n start differential analysis \n ")
```

```
##
##  start differential analysis
##
```

```
# DESeq2.data <- phylo
# DESeq2.data <- subset_samples(DESeq2.data, mapping != exp.factor)
# DESeq2.data
# head(sample_data(DESeq2.data)$treatment, 8)
# # http://joey711.github.io/phyloseq-extensions/DESeq2.html warns of "none" diagnoses
# # I don't think that kind of filetering is relevant here
# sample_data(DESeq2.data)$treatment

# DESeq2.data <- subset_samples(DESeq2.data, mapping != exp.factor)

# diagdds <- phyloseq_to_deseq2(DESeq2.data, ~ treatment)
#   V-HMC, without modificatiosn below:
#      estimating size factors Error in estimateSizeFactorsForMatrix(counts(object), locfunc = locfunc,  :
#      every gene contains at least one zero, cannot compute log geometric means
# DESeq2.rowSums <- rowSums(counts(diagdds))
# DESeq2.rowSums <- cbind.data.frame(names(DESeq2.rowSums),as.numeric(DESeq2.rowSums))
#
# DESeq2.colSums <- colSums(counts(diagdds))
# DESeq2.colSums <- cbind.data.frame(names(DESeq2.colSums),as.numeric(DESeq2.colSums))

# dezeroed  = diagdds[ rowSums(counts(diagdds)) > 1000 , ]
# dezeroed

# x <- estimateSizeFactors(diagdds, type="iterate")
# idx <- rowSums( counts(x, normalized=TRUE) >= 5 ) >= 3
# x <- x[idx,]
# x <- DESeq(x)
```

```
# DESeq2.data <- phylo.k.of.A
# make a DESEQ2 object


# review the treatments (see legacy fitlering above)
# also shows the order, whcih can be relevled
# sample_data(phylo.k.of.A)$treatment <- relevel(sample_data(phylo.k.of.A)$treatment, "LB-AS")
# the first is the reference
sample_data(phylo.k.of.A)$treatment
```

```
##  [1] S2-3H      C          C-NS       C-3H       C-NS       S2-NS
##  [7] C          S2-NS      C          C-3H-LATE  S2-3H-LATE C-3H-LATE
## [13] C          C-NS       C-3H       S2-3H      S2-3H      S2-3H-LATE
## [19] S2-NS      C-3H       S2-NS      S2-3H      S2-3H-LATE C-3H-LATE
## [25] C-NS       C-3H       C-3H-LATE  S2-3H-LATE
## Levels: C C-3H C-3H-LATE C-NS S2-3H S2-3H-LATE S2-NS
```

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

diagdds <- phyloseq_to_deseq2(phylo.k.of.A, ~ treatment)
```

```
## converting counts to integer mode
```

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

class(diagdds)
```

```
## [1] "DESeqDataSet"
## attr(,"package")
## [1] "DESeq2"
```

```
# create a matrix with direct access to the counts (whicha are a slot of an S4 object)
cts <- counts(diagdds)
class(cts)
```

```
## [1] "matrix"
```

```
# do rowwise and colwise sums... was handy when figurting out DESeq2 calulation failure
# would also be applciible to intiial filtering
DESeq2.rowSums <- rowSums(cts)
DESeq2.rowSums <-
```

```
  cbind.data.frame(names(DESeq2.rowSums), as.numeric(DESeq2.rowSums))

DESeq2.colSums <- colSums(cts)
DESeq2.colSums <-
  cbind.data.frame(names(DESeq2.colSums), as.numeric(DESeq2.colSums))

# this was necessary for the DESeq calulation when non-zero values were extremely sparse
geoMeans <-
  apply(cts, 1, function(row)
    if (all(row == 0))
      0
    else
      exp(mean(log(row[row != 0]))))
class(geoMeans)
```

```
## [1] "numeric"
```

```
diagdds <- estimateSizeFactors(diagdds, geoMeans = geoMeans)
class(diagdds)
```

```
## [1] "DESeqDataSet"
## attr(,"package")
## [1] "DESeq2"
```

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

# should I make the model explicit here
diagdds <- DESeq(diagdds, test = "Wald", fitType = "parametric")
```

```
## using pre-existing size factors

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

## -- note: fitType='parametric', but the dispersion trend was not well captured by the
##    function: y = a/x + b, and a local regression fit was automatically substituted.
##    specify fitType='local' or 'mean' to avoid this message next time.

## final dispersion estimates
```

```
## fitting model and testing
class(diagdds)
```

```
## [1] "DESeqDataSet"
## attr(,"package")
## [1] "DESeq2"
```

```
rnms <- resultsNames(diagdds)
rnms
```

```
## [1] "Intercept"           "treatmentC"          "treatmentC.3H"
## [4] "treatmentC.3H.LATE"  "treatmentC.NS"       "treatmentS2.3H"
## [7] "treatmentS2.3H.LATE" "treatmentS2.NS"
```

```
res <-
  results(
    diagdds,
    contrast = c("treatment", constrat.num, constrat.den),
    cooksCutoff = FALSE
  )
class(res)
```

```
## [1] "DESeqResults"
## attr(,"package")
## [1] "DESeq2"
```

```
# contrast
# this argument specifies what comparison to extract from the object to build a results table. one of either:
#   a character vector with exactly three elements:
#       the name of a factor in the design formula, the name of the numerator level for the fold change,
#       and the name of the denominator level for the fold change (simplest case)
# a list of 2 character vectors:
#       the names of the fold changes for the numerator, and the names of the fold changes for the denominator.
#       these names should be elements of resultsNames(object).
#       if the list is length 1, a second element is added which is the empty character vector, character().
#       (more general case, can be to combine interaction terms and main effects)
# a numeric contrast vector with one element for each element in resultsNames(object) (most general case)
# If specified, the name argument is ignored.
#
# name
#     the name of the individual effect (coefficient) for building a results table.
```

```
#     Use this argument rather than contrast for continuous variables,
#     individual effects or for individual interaction terms.
#     The value provided to name must be an element of resultsNames(object).

###   ###   ###


# get table with adjsuted pvalues below a user-speciifed cutoff
sigtab <- res[which(res$padj < DESeq2.alpha), ]
class(sigtab)
```

```
## [1] "DESeqResults"
## attr(,"package")
## [1] "DESeq2"
```

```
sigtab <-
  cbind(as(sigtab, "data.frame"), as(tax_table(phylo.k.of.A)[rownames(sigtab),], "matrix"))

# this shows a single significatnce and abundance fold change by taxon
# what condition is being compared to what? (there are more than two treatmenrts in LalStress)
# see the phyloseq to deseq2 converter which has a parameter for the model

head(sigtab)
```

```
##            baseMean log2FoldChange    lfcSE      stat       pvalue
## GQ512074 5215.99496      -3.519690 1.076375 -3.269949 0.0010756671
## JN905772  219.95395       5.209394 1.757970  2.963301 0.0030435921
## JQ666400  238.01308       5.485724 1.620960  3.384243 0.0007137488
## HQ631046  242.74545      -3.638815 1.208974 -3.009838 0.0026138717
## AJ557830   75.13288       4.773924 1.797403  2.656013 0.0079070655
## GU721432  132.38950       5.092401 1.761780  2.890486 0.0038464620
##                padj Kingdom        Phylum           Class        Order
## GQ512074 0.01290801   Fungi  unidentified    unidentified unidentified
## JN905772 0.02087035   Fungi    Ascomycota Dothideomycetes unidentified
## JQ666400 0.01141998   Fungi  unidentified    unidentified unidentified
## HQ631046 0.02087035   Fungi Basidiomycota Tremellomycetes  Tremellales
## AJ557830 0.03871215   Fungi    Ascomycota Sordariomycetes  Hypocreales
## GU721432 0.02307877   Fungi    Ascomycota Dothideomycetes unidentified
##                  Family        Genus                  Species Rank1 Rank2
```

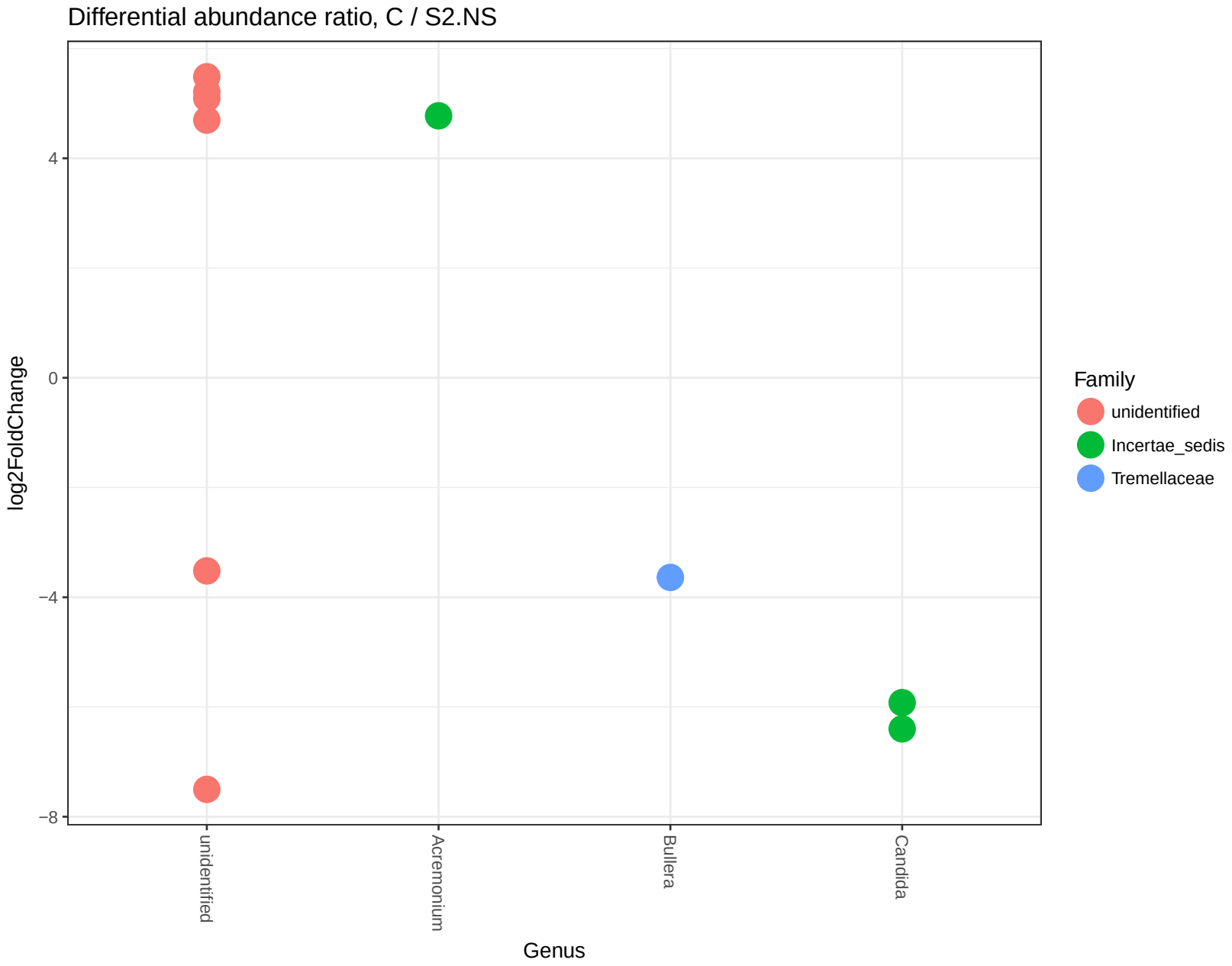
```
## GQ512074   unidentified unidentified          unculturedfungus  <NA>  <NA>
## JN905772   unidentified unidentified unculturedDothideomycetes  <NA>  <NA>
## JQ666400   unidentified unidentified          unculturedfungus  <NA>  <NA>
## HQ631046   Tremellaceae      Bullera       Bullera_sp_TMS_2011  <NA>  <NA>
## AJ557830 Incertae_sedis   Acremonium   Acremonium_sp_JJP_2009a  <NA>  <NA>
## GU721432   unidentified unidentified unculturedDothideomycetes  <NA>  <NA>
##          Rank3 Rank4 Rank5 Rank6 Rank7
## GQ512074  <NA>  <NA>  <NA>  <NA>  <NA>
## JN905772  <NA>  <NA>  <NA>  <NA>  <NA>
## JQ666400  <NA>  <NA>  <NA>  <NA>  <NA>
## HQ631046  <NA>  <NA>  <NA>  <NA>  <NA>
## AJ557830  <NA>  <NA>  <NA>  <NA>  <NA>
## GU721432  <NA>  <NA>  <NA>  <NA>  <NA>
```

```
dim(sigtab)
```

```
## [1] 10 20
```

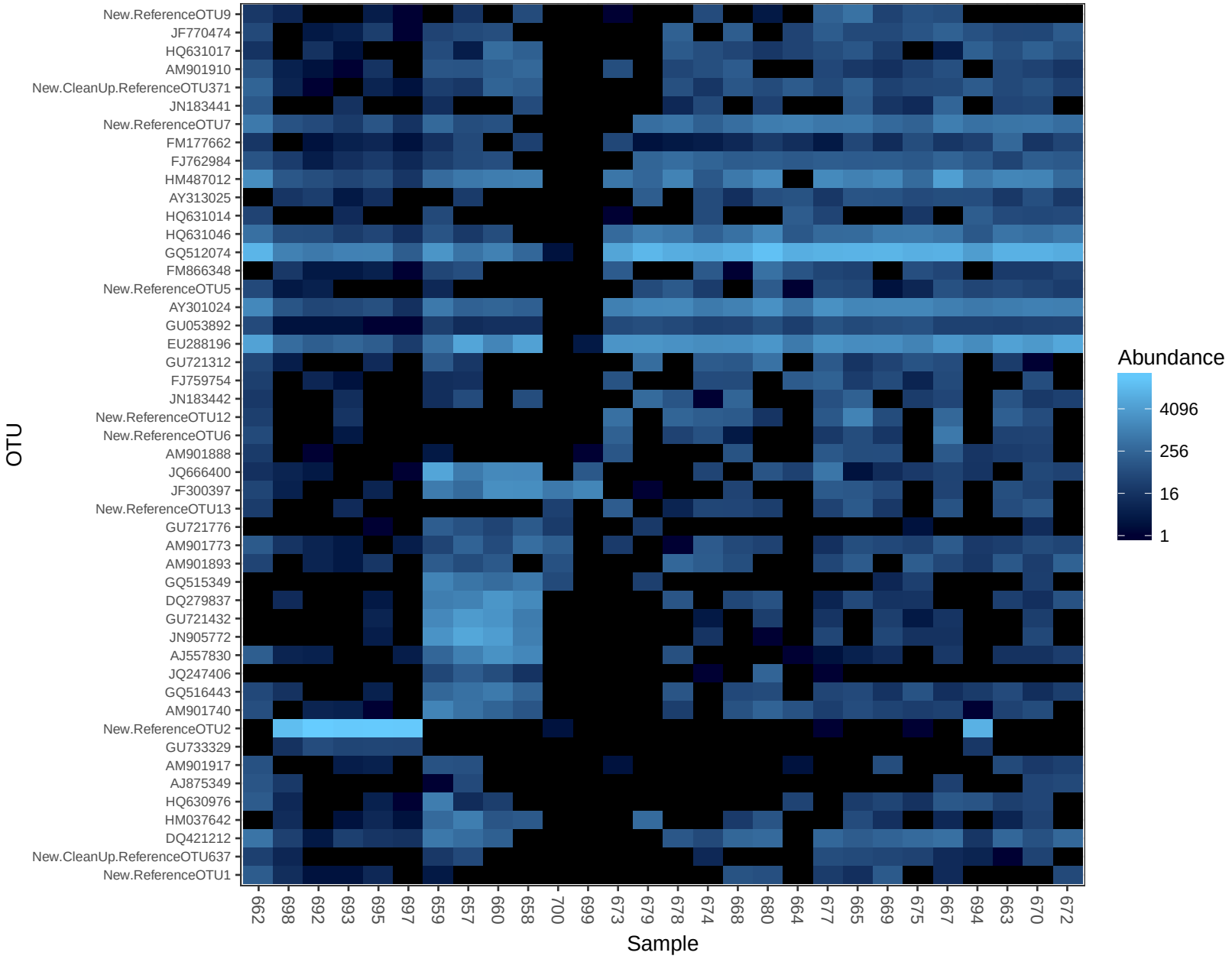
```
# volcano?

# scale_fill_discrete <- function(palname = "Set1", ...) {
#   scale_fill_brewer(palette = palname, ...)
# }


# use color an x position to render two different ranks
# get the rank labels in x
# make sure they're ordered factors



higher.rank.pos <- which(names(sigtab) == higher.rank.name)
lower.rank.pos <- which(names(sigtab) == lower.rank.name)


# higher order
x <-
  tapply(sigtab$log2FoldChange, sigtab[, higher.rank.pos], function(x)
    max(x))
x <- sort(x, TRUE)
```

```r
sigtab[, higher.rank.pos] <-
  factor(as.character(sigtab[, higher.rank.pos]), levels = names(x))


# lower order
x <-
  tapply(sigtab$log2FoldChange, sigtab[, lower.rank.pos], function(x)
    max(x))
x <- sort(x, TRUE)
sigtab[, lower.rank.pos] <-
  factor(as.character(sigtab[, lower.rank.pos]), levels = names(x))

# create plot
p <-
  ggplot(sigtab,
         aes_(
           x = as.name(lower.rank.name),
           y = as.name(y.val.name),
           color = as.name(higher.rank.name)
         )) +
  geom_point(size = 6) +
  theme(axis.text.x = element_text(
    angle = -90,
    hjust = 0,
    vjust = 0.5
  )) + labs(title = paste0(
    "Differential abundance ratio, ",
    constrat.num ,
    " / ",
    constrat.den
  ))
p
```


Differential abundance ratio, C / S2.NS
log2FoldChange
4
0
−4
−8
unidentified
Acremonium
Bullera
Candida
Genus
Family
unidentified
Incertae_sedis
Tremellaceae

```
plot_heatmap(phylo.k.of.A)
```

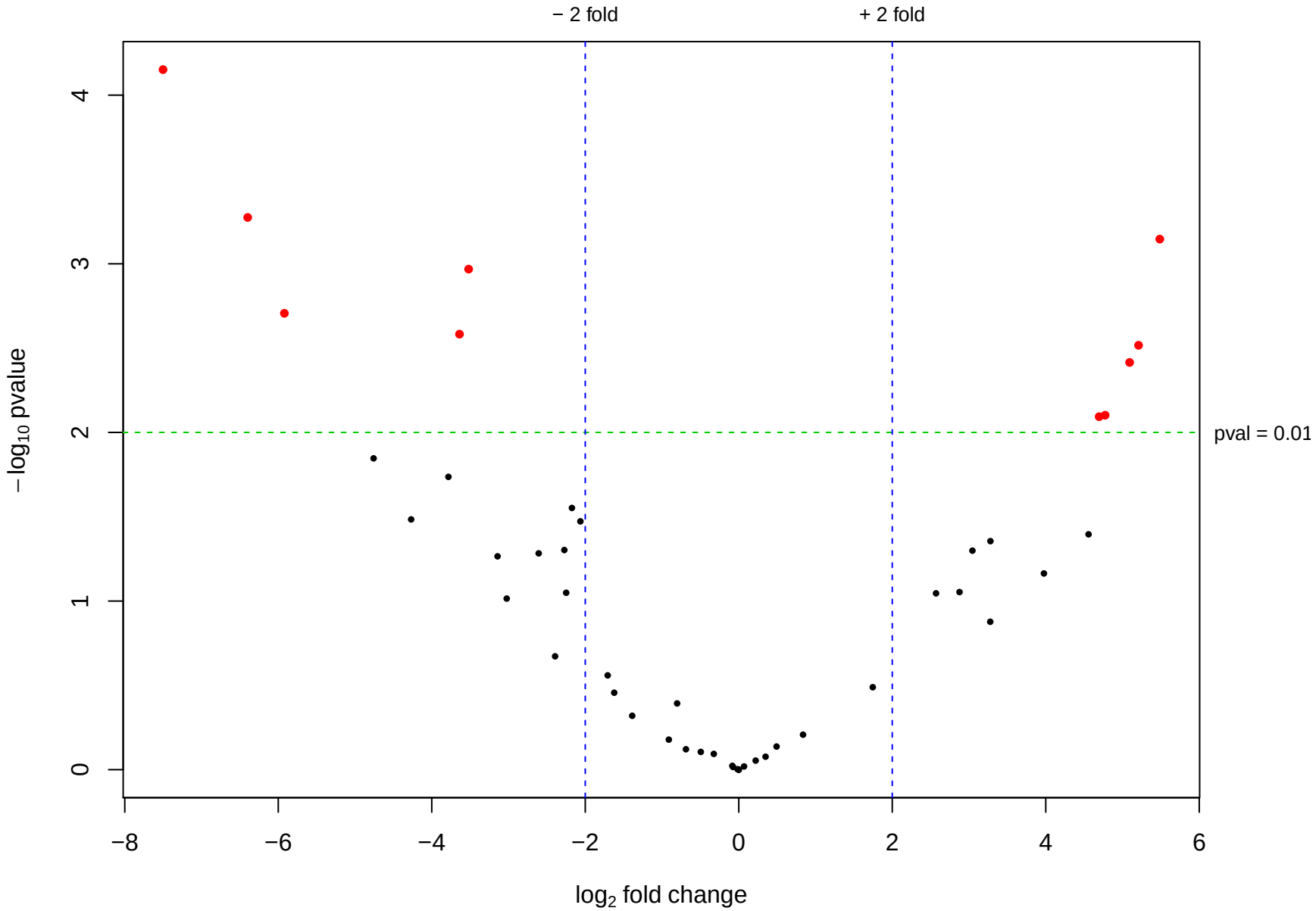
```
## Warning: Transformation introduced infinite values in discrete y-axis
```


New.ReferenceOTU9
JF770474
HQ631017
AM901910
New.CleanUp.ReferenceOTU371
JN183441
New.ReferenceOTU7
FM177662
FJ762984
HM487012
AY313025
HQ631014
HQ631046
GQ512074
FM866348
New.ReferenceOTU5
AY301024
GU053892
EU288196
GU721312
FJ759754
JN183442
New.ReferenceOTU12
New.ReferenceOTU6
AM901888
JQ666400
JF300397
New.ReferenceOTU13
GU721776
AM901773
AM901893
GQ515349
DQ279837
GU721432
JN905772
AJ557830
JQ247406
GQ516443
AM901740
New.ReferenceOTU2
GU733329
AM901917
AJ875349
HQ630976
HM037642
DQ421212
New.CleanUp.ReferenceOTU637
New.ReferenceOTU1
OTU
662
698
692
693
695
697
659
657
660
658
700
699
673
679
678
674
668
680
664
677
665
669
675
667
694
663
670
672
Sample
Abundance
4096
256
16
1

```
# dump sigtab to a file
```

```
tab <- data.frame(logFC = res$log2FoldChange,
                  negLogPval = -log10(res$pvalue))
# head(tab)
par(mar = c(5, 4, 4, 4))
plot(
  tab,
  pch = 16,
  cex = 0.6,
  xlab = expression(log[2] ~ fold ~ change),
  ylab = expression(-log[10] ~ pvalue)
)

sigOTUs <- (abs(tab$logFC) > volcano.lfc & tab$negLogPval > -log10(volcano.pval))
points(tab[sigOTUs,],
       pch = 16,
       cex = 0.8,
       col = "red")
abline(h = -log10(volcano.pval),
       col = "green3",
       lty = 2)
abline(v = c(-volcano.lfc, volcano.lfc),
       col = "blue",
       lty = 2)
mtext(
  paste("pval =", volcano.pval),
  side = 4,
  at = -log10(volcano.pval),
  cex = 0.8,
  line = 0.5,
  las = 1
)
mtext(
  c(paste("-", volcano.lfc, "fold"), paste("+", volcano.lfc, "fold")),
  side = 3,
  at = c(-volcano.lfc, volcano.lfc),
  cex = 0.8,
  line = 0.5
```

```
)

# running interactively?
identify(tab)
```


− 2 fold
+ 2 fold
pval = 0.01
− log10 pvalue
log2 fold change
0
1
2
3
4
−8
−6
−4
−2
0
2
4
6

```
## integer(0)
```

```
# results will be numerical indicies of OTUs... get ID from tab, then lookup in tax table
```