Day 10: Statistical Analysis (Global Gut)

Back to Table of Contents

All of the code in this page is meant to be run in `R` unless otherwise specified.

Loading a genus table and the metadata into R

Before loading data into R, this QIIME command must be run on the command line to collapse OTU counts into genus (L6) and phylum (L2) count tables:

```
# (run on command line)
summarize_taxa.py -i otu_table.biom -L 6

# convert to JSON BIOM format to load into R using R biom package:
biom convert -i otu_table_L6.biom -o otu_table_L6_json.biom --to-json
```

Inside `R`, Install biom package and vegan package if not installed.

```
install.packages(c('biom','vegan'),repo='http://cran.wustl.edu')
```

Load biom package, vegan package; load data

```
# increase display width for better viewing
options(width=110)

library('biom')
library('vegan')

# load biom file
genus.biom <- read_biom('otu_table_L6_json.biom')

# Extract data matrix (genus counts) from biom table
genus <- as.matrix(biom_data(genus.biom))

# transpose so that rows are samples and columns are genera
genus <- t(genus)

# load mapping file
map <- read.table('map.txt', sep='\t', comment='', head=T, row.names=1)
```

It is extremely important to ensure that your genus table and metadata table sample IDs are lined up correctly.

```
# find the overlapping samples
common.ids <- intersect(rownames(map), rownames(genus))

# get just the overlapping samples
genus <- genus[common.ids,]
map <- map[common.ids,]
```

See dimensions of genus table. Then drop genera present in < 10% of samples.

```
dim(genus)
```

```
## [1]  66 205
```

```
genus <- genus[,colMeans(genus > 0) >= .1]
dim(genus)
```

```
## [1] 66 97
```

Show only the first ten genera in genus table

```
colnames(genus)[1:10]
```

```
##  [1] "k__Archaea;p__Euryarchaeota;c__Methanobacteria;o__Methanobacteriales;f__Methanobacteriaceae;g_
##  [2] "k__Archaea;p__Euryarchaeota;c__Methanobacteria;o__Methanobacteriales;f__Methanobacteriaceae;g_
##  [3] "k__Bacteria;p__Actinobacteria;c__Actinobacteria;o__Actinomycetales;f__Actinomycetaceae;g__Acti
##  [4] "k__Bacteria;p__Actinobacteria;c__Actinobacteria;o__Bifidobacteriales;f__Bifidobacteriaceae;g__
##  [5] "k__Bacteria;p__Actinobacteria;c__Coriobacteriia;o__Coriobacteriales;f__Coriobacteriaceae;g__"
##  [6] "k__Bacteria;p__Actinobacteria;c__Coriobacteriia;o__Coriobacteriales;f__Coriobacteriaceae;g__Ad
##  [7] "k__Bacteria;p__Actinobacteria;c__Coriobacteriia;o__Coriobacteriales;f__Coriobacteriaceae;g__Co
##  [8] "k__Bacteria;p__Actinobacteria;c__Coriobacteriia;o__Coriobacteriales;f__Coriobacteriaceae;g__Sl
##  [9] "k__Bacteria;p__Bacteroidetes;c__Bacteroidia;o__Bacteroidales;f__;g__"
## [10] "k__Bacteria;p__Bacteroidetes;c__Bacteroidia;o__Bacteroidales;f__Bacteroidaceae;g__Bacteroides"
```

Abbreviate the taxonomic names to make them easier to display.

```
colnames(genus) <- sapply(strsplit(colnames(genus),';'),function(xx) paste(paste(substr(xx[-c(1,length(
```

Show the first 10 rows and first 2 columns of the genus table

```
genus[1:10,1:2]
```

```
##                Eury;Meth;Meth;Meth;Methanobrevibacter Eury;Meth;Meth;Meth;Methanosphaera
## h122M.1.418534                           0.0015625000                       0.0015625000
## h85M.1.418596                            0.0010368066                       0.0005184033
## h95M.1.418831                            0.0004625347                       0.0009250694
## h9M.1.418588                             0.0000000000                       0.0000000000
## k278A.2.418424                           0.0000000000                       0.0000000000
## h146M.1.418838                           0.0000000000                       0.0007648184
## h147M.1.418531                           0.0000000000                       0.0031890661
## h101M.1.418586                           0.0004692633                       0.0000000000
## h165M.1.418394                           0.0000000000                       0.0000000000
## h186M.1.418788                           0.0003984064                       0.0015936255
```

See available columns in the metadata

```
colnames(map)
```

```
##  [1] "BarcodeSequence"               "LinkerPrimerSequence"          "AGE"
##  [4] "AGE_GROUP"                     "BODY_SITE"                     "COUNTRY"
##  [7] "ELEVATION"                     "EXPERIMENT_DESIGN_DESCRIPTION" "FAMILY_RELATIONSHIP"
## [10] "HOST_COMMON_NAME"              "HOST_SUBJECT_ID"               "LATITUDE"
## [13] "LONGITUDE"                     "PCR_PRIMERS"                   "REGION"
## [16] "RUN"                           "RUN_CENTER"                    "RUN_DATE"
## [19] "RUN_LANE"                      "SEX"                           "STUDY_ABSTRACT"
## [22] "Description"
```

Show how many samples are from each Country

```
table(map$COUNTRY)
```

```
##
##                   GAZ:Malawi GAZ:United States of America                GAZ:Venezuela
##                           22                           22                           22
```

Basic association testing

Let's run some tests on Prevotella. First extract the Prevotella column and save it to a variable `prevotella` for convenience.

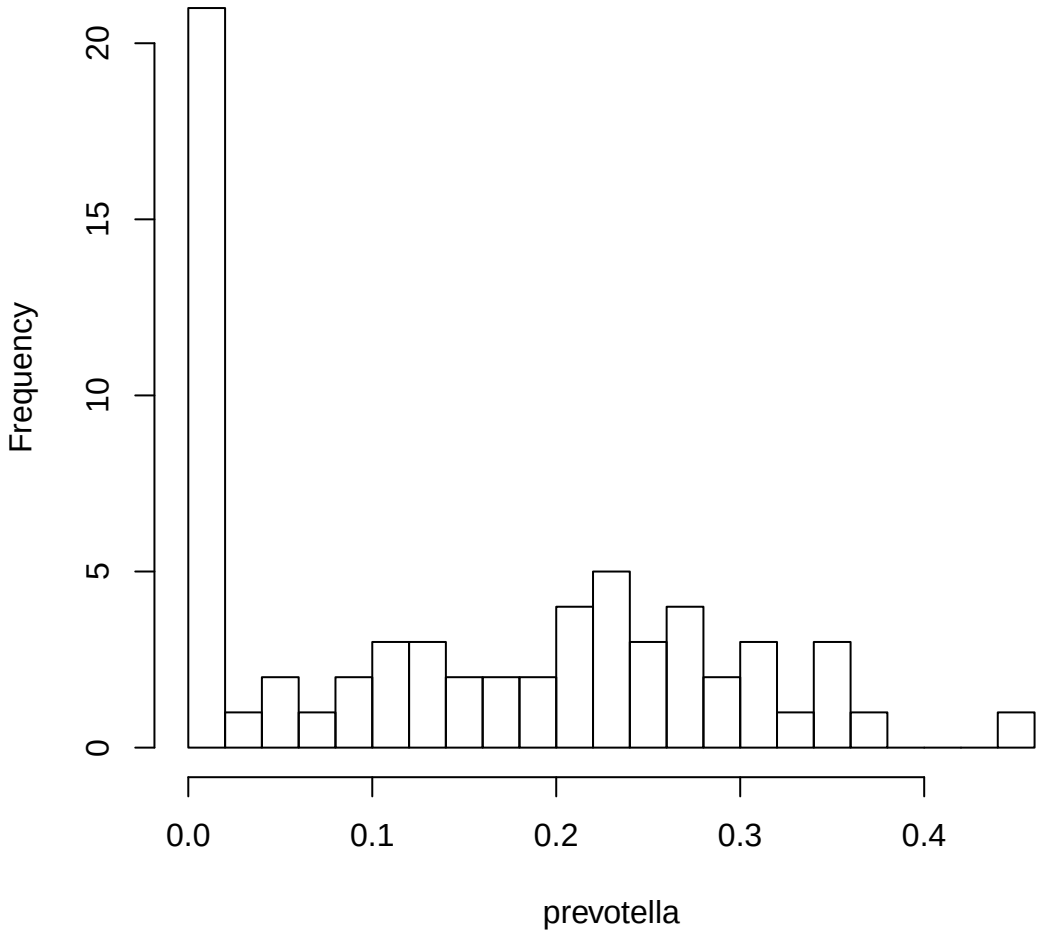
```
# find out what column Prevotella is in
# the "$" tells grep to find only column names that end with "Prevotella".
grep('Prevotella$',colnames(genus))
```

```
## [1] 13
```

```
# save that column in a variable
prevotella <- genus[,grep('Prevotella$',colnames(genus))]
```

Visualize the distribution of Prevotella

```
# find out what column Prevotella is in
hist(prevotella, br=30)
```



Run a test of Pearson's correlation of Prevotella and age. Note that the result is not quite significant (p=0.0531).

```
cor.test(prevotella, map$AGE)
```

```
##
##  Pearson's product-moment correlation
##
## data:  prevotella and map$AGE
## t = -1.9704, df = 64, p-value = 0.05312
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.454858210  0.003055594
## sample estimates:
##       cor
## -0.239154
```

Now run a linear regression of prevotella against age. Notice that statistically this is equivalent to running the Pearson's correlation. The p-value in row 2 column 4 of the "Coefficients" table is the same as the p-value from the correlation test.

```
# fit a linear model. The "~" means "as a function of"
fit <- lm(prevotella ~ map$AGE)

# print a summary of the results
summary(fit)
```

```
##
## Call:
## lm(formula = prevotella ~ map$AGE)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.18946 -0.12186 -0.01805  0.10726  0.30337
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.2009925  0.0319288   6.295 3.15e-08 ***
## map$AGE     -0.0019218  0.0009753  -1.970   0.0531 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.124 on 64 degrees of freedom
## Multiple R-squared:  0.05719,    Adjusted R-squared:  0.04246
## F-statistic: 3.883 on 1 and 64 DF,  p-value: 0.05312
```

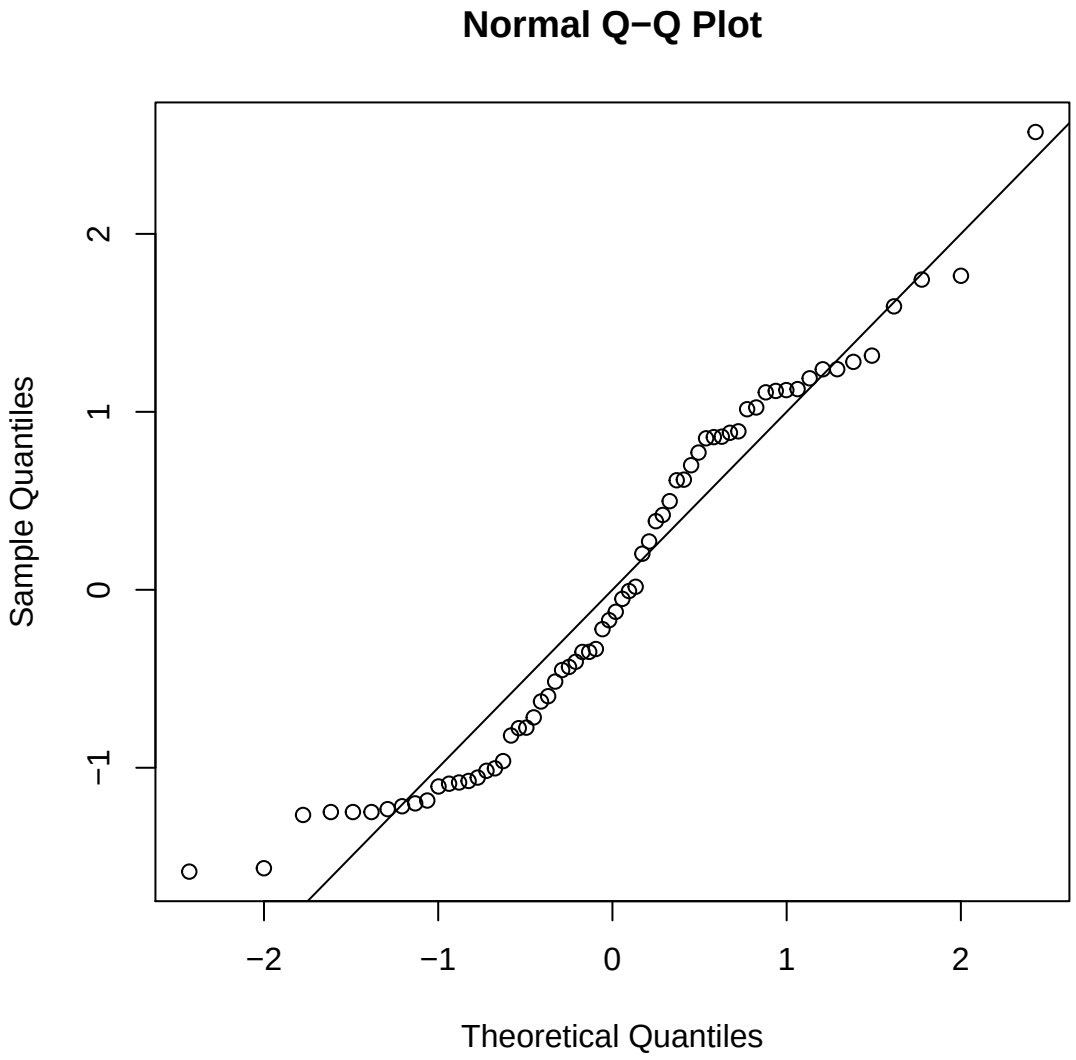
```
# A nice way to get the exact p-value for the age regression coefficient using the anova function
pval <- anova(fit)['map$AGE','Pr(>F)']
pval
```

```
## [1] 0.0531206
```

Testing for normally distributed data

We can test whether the residuals are normally distributed Kolmogorov-Smirnov test. If $p < 0.05$, we can reject the null hypothesis that the data came from a normal distribution, meaning that the linear test is not appropriate.

```
# Make a quantile-quantile plot of the (studentized) residuals vs. a normal distribution
qqnorm(rstudent(fit)); abline(0,1)
```




```
# Kolmogorov-Smirnov test
ks.test(rstudent(fit), pnorm, mean=mean(rstudent(fit)), sd=sd(rstudent(fit)))
```

```
## Warning in ks.test(rstudent(fit), pnorm, mean = mean(rstudent(fit)), sd = sd(rstudent(fit))): ties sh
## be present for the Kolmogorov-Smirnov test

##
##  One-sample Kolmogorov-Smirnov test
##
## data:  rstudent(fit)
## D = 0.10218, p-value = 0.496
## alternative hypothesis: two-sided
```

Controling for confounders

Perhaps country of origin is a confounder that is obscuring the association of Prevotella and Age. Using lm() we can add confounders to the regression. Now after removing the effects of country, there is a strong association of Prevotella and age.

```
# fit a linear model. The "~" means "as a function of"
fit <- lm(prevotella ~ map$AGE + map$COUNTRY)

# print a summary of the results
summary(fit)
```

```
##
## Call:
## lm(formula = prevotella ~ map$AGE + map$COUNTRY)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.150373 -0.040611 -0.008627  0.033487  0.199252
##
## Coefficients:
##                                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)                             0.2982044  0.0238895  12.483  < 2e-16 ***
## map$AGE                                -0.0016456  0.0006193  -2.657 0.010010 *
## map$COUNTRYGAZ:United States of America -0.2311571  0.0237310  -9.741 4.08e-14 ***
## map$COUNTRYGAZ:Venezuela                -0.0843013  0.0237296  -3.553 0.000736 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.07861 on 62 degrees of freedom
## Multiple R-squared:  0.6329, Adjusted R-squared:  0.6152
## F-statistic: 35.63 on 3 and 62 DF,  p-value: 1.641e-13
```

Testing multiple hypotheses

We have so far only tested one genus. Let's test them all using a loop.

```
# pvals is a vector initialized with zeroes
# with enough slots for the different genera
pvals <- numeric(ncol(genus))

# "name" the pvalues after the genera
names(pvals) <- colnames(genus)

# Loop through the columns of the genus table, testing each one
for(i in 1:ncol(genus)) {
    fit <- lm(genus[,i] ~ map$AGE + map$COUNTRY)
    pvals[i] <-  anova(fit)['map$AGE','Pr(>F)']
}

# note, you could put this all on one line with:
# for(i in 1:ncol(genus)) pvals[i] <-  anova(lm(genus[,i] ~ map$AGE + map$COUNTRY))['map$AGE','Pr(>F)']
```

```
# print the 10 smallest p-values:
sort(pvals)[1:10]
```

```
##                                WPS-;;;;         Bact;Bact;Bact;Prev;Prevotella
##                                0.002762885                         0.002841431
##                     Verr;Verr;WCHB;RFP1;                  Firm;Clos;Clos;Chri;
##                                0.004747721                         0.010042840
##         Firm;Baci;Lact;Stre;Lactococcus                      Prot;Alph;RF32;;
##                                0.011344588                         0.019484449
## Eury;Meth;Meth;Meth;Methanobrevibacter                  Bact;Bact;Bact;S24-;
##                                0.028070760                         0.030610821
##    Firm;Clos;Clos;Rumi;Faecalibacterium                        Prot;Alph;;;
##                                0.035887736                         0.045431657
```

Note, you could put this all on one line with this:

```
# "apply" with genus, 2 means do something to every column of genus
# ("apply" with genus, 1 would mean do something every row)
# the last part defines a new function to do to each column, which
# will be passed in the temporary variable named "xx"
pvals <- apply(genus, 2, function(xx) anova(lm(xx ~ map$AGE + map$COUNTRY))['map$AGE','Pr(>F)'])

# print the 10 smallest p-values:
sort(pvals)[1:10]
```

```
##                                WPS-;;;;         Bact;Bact;Bact;Prev;Prevotella
##                                0.002762885                         0.002841431
##                     Verr;Verr;WCHB;RFP1;                  Firm;Clos;Clos;Chri;
##                                0.004747721                         0.010042840
##         Firm;Baci;Lact;Stre;Lactococcus                      Prot;Alph;RF32;;
##                                0.011344588                         0.019484449
## Eury;Meth;Meth;Meth;Methanobrevibacter                  Bact;Bact;Bact;S24-;
##                                0.028070760                         0.030610821
##    Firm;Clos;Clos;Rumi;Faecalibacterium                        Prot;Alph;;;
##                                0.035887736                         0.045431657
```

Looks like there are some significant p-values. But did they happen just by chance? There are 97 columns in the genus table, so that means we did 97 tests, and about 5% of them should be $p < 0.05$ just by chance. To correct for this, we can adjust the p-values using the `p.adjust` function. Here we are correcting for multiple hypothesis testing use False Discovery Rate (FDR). The adjusted p-values are often called "q-values."

```
qvals <- p.adjust(pvals,'fdr')

# print the lowest 10 q-values
sort(qvals)[1:10]
```

```
##         Bact;Bact;Bact;Prev;Prevotella                                WPS-;;;;
##                                0.1378094                           0.1378094
##                     Verr;Verr;WCHB;RFP1;         Firm;Baci;Lact;Stre;Lactococcus
##                                0.1535096                           0.2200850
##                     Firm;Clos;Clos;Chri;                      Prot;Alph;RF32;;
```

```
##                                0.2200850                                0.3149986
## Eury;Meth;Meth;Meth;Methanobrevibacter                Bact;Bact;Bact;S24-;
##                                0.3711562                                0.3711562
##   Firm;Clos;Clos;Rumi;Faecalibacterium                        Prot;Alph;;;
##                                0.3867900                                0.4406871
```

Testing with generalized linear regression.

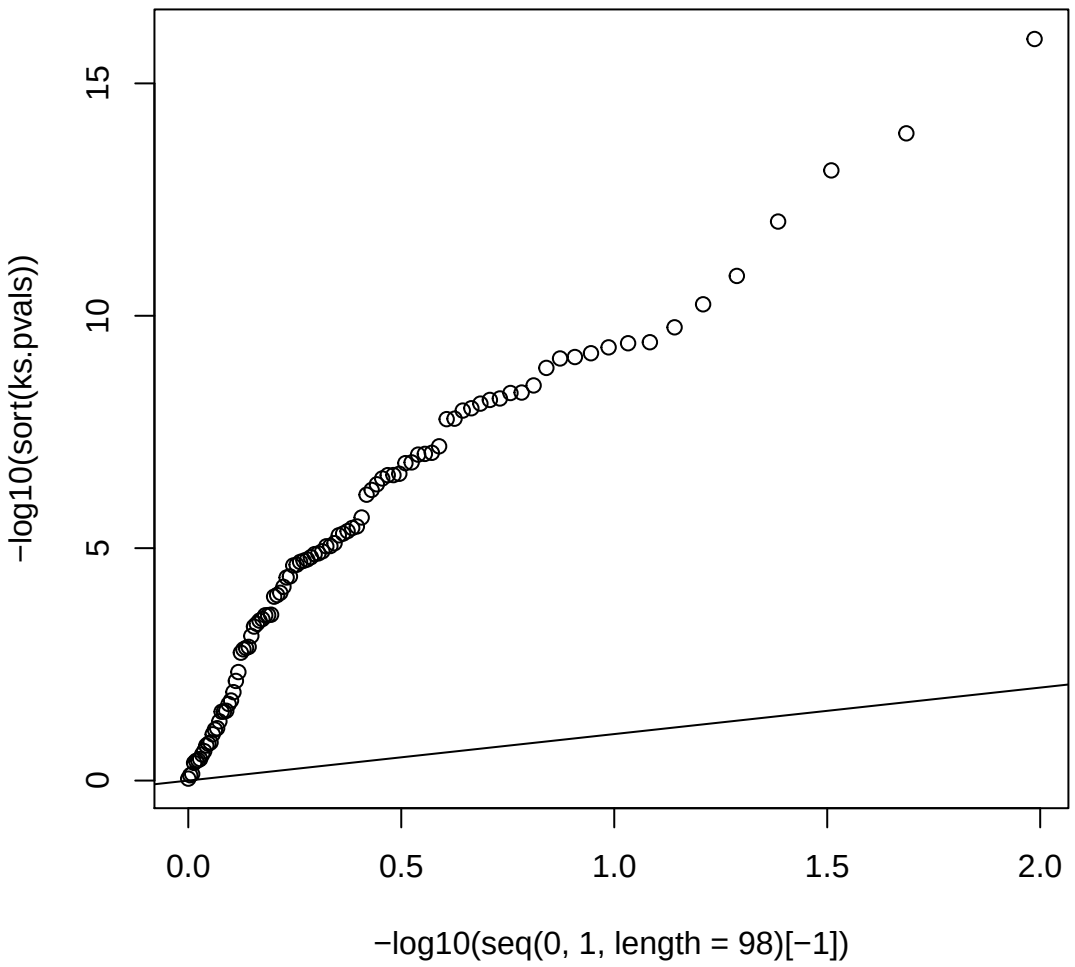
Let's test whether the residuals from linear regression were normally distributed for all of the taxa. To follow along, you can put the following code in a separate text file called `stats.r` (or whatever you want to call it, and then call it using `source('stats.r')`.

```
ks.pvals <- numeric(ncol(genus))

# "name" the pvalues after the genera
names(ks.pvals) <- colnames(genus)

# turn annoying warnings off
options(warn=-1)

# Loop through the columns of the genus table, testing each one
for(i in 1:ncol(genus)) {
    fit <- lm(genus[,i] ~ map$AGE + map$COUNTRY)
    ks.pvals[i] <-  ks.test(rstudent(fit), pnorm, mean=mean(rstudent(fit)), sd=sd(rstudent(fit)),exact=
}

# turn warnings back on
options(warn=0)


# Now since we ran 97 tests we should correct for multiple hypothesis testing.
ks.qvals <- p.adjust(ks.pvals,'fdr')

# print the lowest 10 q-values
sort(ks.qvals)[1:10]
```

```
##              Bact;Bact;Bact;RF16;              Firm;Baci;Lact;Leuc;   Prot;Gamm;Ente;Ente;Citrobacte
##                      1.076916e-14                      5.761502e-13                      2.408703e-1
##   Lent;[Len;Vict;Vict;Victivallis             Prot;Beta;Burk;Coma;     Bact;Bact;Bact;Rike;Alistipes
##                      2.278486e-11                      2.695500e-10                      9.155081e-1
##  Tene;Moll;Anae;Anae;Anaeroplasma Firm;Baci;Lact;Stre;Streptococcus   Prot;Gamm;Ente;Ente;Escherichi
##                      2.454626e-09                      4.203562e-09                      4.203562e-0
## Firm;Erys;Erys;Erys;Coprobacillus
##                      4.632104e-09
```

Wow. There are a lot of these whose residuals are highly non-normal. We can confirm this with a q-q plot of the observed p-values vs. the expected p-values in a uniform distribution:

```
plot(-log10(seq(0,1,length=98)[-1]), -log10(sort(ks.pvals))); abline(0,1)
```



Let's use a negative binomial distribution instead. We will use the `edgeR` package. If you don't have it, you can install it with:

```
source("https://bioconductor.org/biocLite.R")

# If the previous command doesn't work, try http://

biocLite("edgeR")
```

Now use a convenient wrapper function that I provided in `repo/src/wrap.edgeR.r`. Load this script using `source`:

```
source('../../src/wrap.edgeR.r')
```

Note: the negative binomial uses raw counts of sequences (rarefied or not), not the relative abundances. Therefore we must re-run `summarize_taxa.py` with the `-a` flag to use absolute abundance. **These commands are run on the command line (not in R)**

```
# (run on command line)
summarize_taxa.py -i otu_table.biom -L 6 -a -o taxa-absolute

# convert to JSON BIOM format to load into R using R biom package:
biom convert -i taxa-absolute/otu_table_L6.biom -o taxa-absolute/otu_table_L6_json.biom --to-json
```

Now we need to load the genus table again.

```
genus.biom <- read_biom('taxa-absolute/otu_table_L6_json.biom')
genus.a <- as.matrix(biom_data(genus.biom))
genus.a <- t(genus.a)
genus.a <- genus.a[common.ids,]
genus.a <- genus.a[,colMeans(genus.a > 0) >= .1]
colnames(genus.a) <- sapply(strsplit(colnames(genus.a),';'),function(xx) paste(paste(substr(xx[-c(1,leng
```

First let us run the regression without covariates. The main function in this script is `glm.edgeR()`. This function will perform multiple hypothesis testing on all columns of a data matrix. It has the following main parameters:
- x: the independent variable. If discrete must be binary; OK to be continuous. - Y: A matrix with samples in rows and dependent variables in columns - covariates: a matrix of additional covariates you want to control for (default NULL)

```
result <- glm.edgeR(x=map$AGE, Y=genus.a)
```

```
## Making DGEList...
## calcNormFactors...
## estimate common dispersion...
## estimate trended dispersion...
## estimate tagwise dispersion...
## fit glm...
## likelihood ratio test...
```

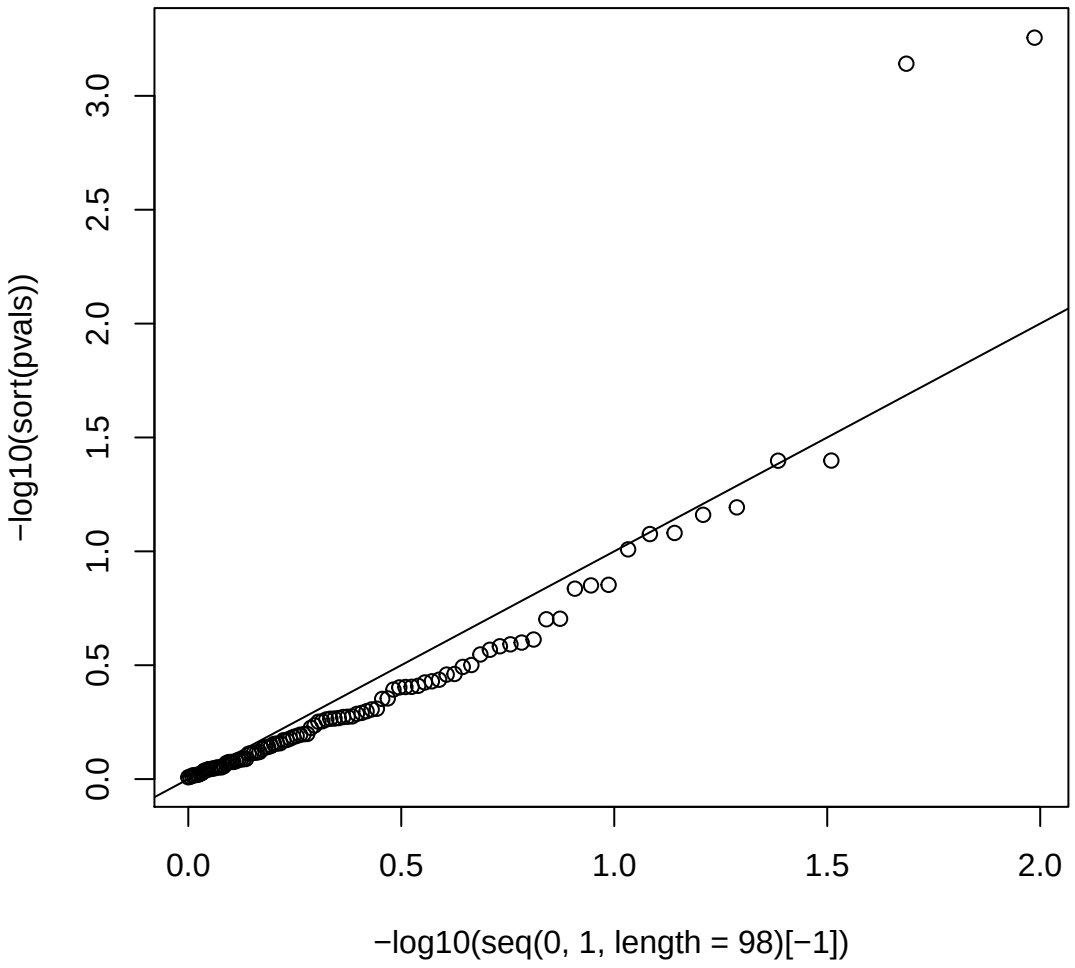
We can print the top "tags" (genera) using the topTags function. Note that two genera are significant even after correction for multiple hypothesis testing.

```
topTags(result)
```

```
## Coefficient:  x
##                                          logFC   logCPM        LR       PValue        FDR
## Firm;Baci;Lact;Stre;Lactococcus       0.10705153 11.04892 11.920014 0.0005553406 0.03503487
## Firm;Baci;Lact;Stre;Streptococcus    -0.06816128 13.37950 11.430793 0.0007223685 0.03503487
## Firm;Clos;Clos;Chri;                  0.04133732 12.95898  4.220124 0.0399472332 0.96984321
## Firm;Clos;Clos;Rumi;Faecalibacterium -0.01574617 15.82647  4.218159 0.0399935344 0.96984321
## Firm;Clos;Clos;Veil;Veillonella      -0.06840230 11.08899  3.427907 0.0641032986 0.98181209
## Acti;Acti;Bifi;Bifi;Bifidobacterium  -0.04492773 14.14257  3.303897 0.0691157195 0.98181209
## Prot;Gamm;Ente;Ente;                  0.03860829 13.37335  3.004676 0.0830245689 0.98181209
## Prot;Gamm;Ente;Ente;Citrobacter       0.13186169 10.92203  2.986154 0.0839794047 0.98181209
## Prot;Alph;RF32;;                      0.05371944 11.56598  2.737262 0.0980326107 0.98181209
## Verr;Verr;WCHB;RFP1;                  0.03726694 10.47449  2.174756 0.1402917541 0.98181209
```

If we plot all p-values coming from edgeR in a quantile-quantile plot, we see that they mostly follow the null (uniform) distribution:

```
pvals <- topTags(result,n=Inf)$table[,'PValue']
plot(-log10(seq(0,1,length=98)[-1]), -log10(sort(pvals))); abline(0,1)
```



However, we have not controlled for COUNTRY, which may be a confounder. We can do this with edgeR.

```
result <- glm.edgeR(x=map$AGE, Y=genus.a, covariates=map$COUNTRY)
```

```
## Making DGEList...
## calcNormFactors...
## estimate common dispersion...
## estimate trended dispersion...
## estimate tagwise dispersion...
## fit glm...
## likelihood ratio test...
```

```
topTags(result)
```

```
## Coefficient:  x
##                                          logFC   logCPM       LR       PValue       FDR
## Firm;Baci;Lact;Stre;Streptococcus    -0.05175765 13.37950 6.890239 0.008666769 0.3498913
## Bact;Bact;Bact;[Par;[Prevotella]      0.04802689 13.87313 6.810955 0.009060028 0.3498913
## Firm;Baci;Lact;Stre;Lactococcus       0.07897558 11.04892 6.494416 0.010821381 0.3498913
## Firm;Clos;Clos;Rumi;Faecalibacterium -0.01581681 15.82647 5.120219 0.023648625 0.4872203
## Prot;Alph;RF32;;                      0.05509428 11.56598 4.547161 0.032973382 0.4872203
## Bact;Bact;Bact;Prev;Prevotella       -0.03662945 17.37559 4.424217 0.035432356 0.4872203
## Acti;Acti;Bifi;Bifi;Bifidobacterium  -0.05065800 14.14257 4.076610 0.043480810 0.4872203
## Tene;Moll;Anae;Anae;Anaeroplasma     -0.02490410 10.47215 3.816345 0.050754830 0.4872203
## Firm;Clos;Clos;Chri;                  0.03788764 12.95898 3.665489 0.055550332 0.4872203
## Firm;Erys;Erys;Erys;Bulleidia         0.02852503 11.68246 3.525563 0.060429297 0.4872203
```

Note that no genera are significantly associated with age after controlling for country, and using the negative binomial as the assumed distribution for the residuals.

We can also pass in a matrix of covariates like this, although note that SEX is not a good variable here because there are only 6 males and there are 6 with unknown gender:

```
result <- glm.edgeR(x=map$AGE, Y=genus.a, covariates=map[ , c('COUNTRY','SEX')])
```

Let us test whether any genera are significantly associated with being from the USA, while controling for age:

```
# make a vector that is TRUE if sample is from USA, FALSE otherwise
# the "==" means "test if equal"
is.USA <- map$COUNTRY == "GAZ:United States of America"

# run with is.USA as the independent variable
result <- glm.edgeR(x=is.USA, Y=genus.a, covariates=map$AGE)
```

```
## Making DGEList...
## calcNormFactors...
## estimate common dispersion...
## estimate trended dispersion...
## estimate tagwise dispersion...
## fit glm...
## likelihood ratio test...
```

```
topTags(result)
```

```
## Coefficient:  xTRUE
##                                         logFC   logCPM        LR       PValue          FDR
## Bact;Bact;Bact;Rike;                 6.348729 13.33516 176.08813 3.464099e-40 3.360176e-38
## Bact;Bact;Bact;Bact;Bacteroides      5.317940 16.17437 141.26282 1.409522e-32 6.836184e-31
## Prot;Gamm;Aero;Succ;Succinivibrio   -9.118860 14.63896  85.44031 2.388020e-20 7.721264e-19
## Firm;Erys;Erys;Erys;Bulleidia       -5.810289 11.68246  72.09081 2.055187e-17 4.983828e-16
## Cyan;4C0d;YS2;;                     -6.150553 12.51266  60.46102 7.505003e-15 1.455971e-13
## Bact;Bact;Bact;Porp;Parabacteroides  3.040652 13.46415  48.91294 2.675800e-12 4.325877e-11
## Bact;Bact;Bact;;                    -5.439312 12.89334  47.72142 4.912944e-12 6.807937e-11
## Bact;Bact;Bact;[Par;[Prevotella]    -5.104367 13.87313  42.17749 8.335380e-11 1.010665e-09
## Spir;Spir;Spir;Spir;Treponema       -7.063003 12.57612  31.76952 1.735961e-08 1.788346e-07
## Firm;Clos;Clos;Clos;Clostridium     -2.807418 12.71195  31.65262 1.843655e-08 1.788346e-07
```

Now we're talking! Many significant results. Print all with:

```
topTags(result, n=Inf)
```

```
## Coefficient:  xTRUE
##                                                 logFC   logCPM           LR       PValue          FDR
## Bact;Bact;Bact;Rike;                         6.34872919 13.33516 176.08813410 3.464099e-40 3.360176e-38
## Bact;Bact;Bact;Bact;Bacteroides              5.31793971 16.17437 141.26282335 1.409522e-32 6.836184e-31
## Prot;Gamm;Aero;Succ;Succinivibrio           -9.11886001 14.63896  85.44031428 2.388020e-20 7.721264e-19
## Firm;Erys;Erys;Erys;Bulleidia               -5.81028945 11.68246  72.09080968 2.055187e-17 4.983828e-16
## Cyan;4C0d;YS2;;                             -6.15055346 12.51266  60.46102012 7.505003e-15 1.455971e-13
## Bact;Bact;Bact;Porp;Parabacteroides          3.04065242 13.46415  48.91293677 2.675800e-12 4.325877e-11
## Bact;Bact;Bact;;                            -5.43931160 12.89334  47.72141975 4.912944e-12 6.807937e-11
## Bact;Bact;Bact;[Par;[Prevotella]            -5.10436692 13.87313  42.17749372 8.335380e-11 1.010665e-09
## Spir;Spir;Spir;Spir;Treponema               -7.06300267 12.57612  31.76951557 1.735961e-08 1.788346e-07
## Firm;Clos;Clos;Clos;Clostridium             -2.80741842 12.71195  31.65262166 1.843655e-08 1.788346e-07
## Bact;Bact;Bact;Prev;Prevotella              -3.90051018 17.37559  29.89298195 4.565624e-08 4.026050e-07
## Firm;Erys;Erys;Erys;p-75-a5                 -5.47452317 11.25422  29.24810688 6.367841e-08 5.147338e-07
## Firm;Clos;Clos;Clos;                        -2.66352289 14.69944  26.15734267 3.146982e-07 2.348132e-06
## Bact;Bact;Bact;[Bar;                         5.05288939 11.65976  25.87576217 3.641125e-07 2.522779e-06
## Prot;Alph;;;                                -4.04836574 10.71031  24.95477187 5.869108e-07 3.795357e-06
## Prot;Delt;Desu;Desu;Bilophila                3.00881353 11.14211  24.77978054 6.426811e-07 3.896254e-06
## Prot;Gamm;Aero;Succ;Ruminobacter            -8.04604202 13.57149  24.24735422 8.472339e-07 4.834217e-06
## Acti;Cori;Cori;Cori;                        -2.19836924 11.95478  21.83643134 2.969088e-06 1.544925e-05
## Bact;Bact;Bact;[Par;                        -4.51583919 11.16206  21.79991062 3.026142e-06 1.544925e-05
## Prot;Gamm;Ente;Ente;                        -4.01465774 13.37335  20.92665565 4.772078e-06 2.314458e-05
## Bact;Bact;Bact;[Odo;Odoribacter              2.66287308 10.68185  20.40499365 6.266606e-06 2.894575e-05
## Elus;Elus;Elus;Elus;                        -4.27200443 11.01060  20.09609965 7.364671e-06 3.247151e-05
## Eury;Meth;Meth;Meth;Methanosphaera          -3.42571689 10.53542  19.17500313 1.192650e-05 5.029871e-05
## Bact;Bact;Bact;[Par;CF231                   -5.13181146 11.37633  18.83159393 1.427823e-05 5.770784e-05
## Firm;Baci;Lact;Lact;Lactobacillus           -4.31172131 11.59423  18.35539116 1.832996e-05 6.825359e-05
## Firm;Clos;Clos;Veil;Veillonella             -3.92053247 11.08899  18.34255746 1.845385e-05 6.825359e-05
## Firm;Clos;Clos;Clos;Sarcina                 -3.45275729 10.74961  18.28715162 1.899842e-05 6.825359e-05
## Verr;Verr;WCHB;RFP1;                        -3.19969497 10.47449  18.05346524 2.147874e-05 7.440848e-05
## Firm;Erys;Erys;Erys;Holdemania               2.30506183 10.23023  17.98543590 2.226016e-05 7.445638e-05
## Bact;Bact;Bact;[Odo;Butyricimonas            2.19789798 10.75335  17.02606034 3.687029e-05 1.192139e-04
## Bact;Bact;Bact;S24-;                        -3.00876173 13.58997  16.85568462 4.033235e-05 1.262012e-04
## Prot;Gamm;Past;Past;Haemophilus             -3.43254646 11.52620  15.22557508 9.540262e-05 2.814560e-04
## Prot;Epsi;Camp;Camp;Campylobacter           -4.75332614 11.12120  15.21865048 9.575307e-05 2.814560e-04
## Firm;Erys;Erys;Erys;Catenibacterium         -3.07552218 12.61100  14.68473898 1.270710e-04 3.625261e-04
## Prot;Gamm;Ente;Ente;Serratia                -3.76667441 10.74841  14.03098412 1.798230e-04 4.876734e-04
## Firm;Clos;Clos;Lach;Coprococcus              0.88322131 14.33786  14.01879736 1.809922e-04 4.876734e-04
## Acti;Cori;Cori;Cori;Adlercreutzia            2.27752686 10.25628  13.86927860 1.959759e-04 5.137747e-04
## Prot;Beta;Burk;Alca;Sutterella               1.54714024 12.89825  13.54013182 2.335160e-04 5.960803e-04
## Bact;Bact;Bact;Prev;                        -2.53317787 10.36637  13.11116664 2.935408e-04 7.300887e-04
## Firm;Baci;Lact;Leuc;                        -4.01340415 10.72999  12.74271263 3.573990e-04 8.528504e-04
## Firm;Clos;Clos;Lach;                         0.68500442 16.40942  12.72664240 3.604831e-04 8.528504e-04
## Firm;Clos;Clos;Lach;Butyrivibrio            -2.74515170 10.45324  12.23235981 4.696776e-04 1.084732e-03
## Verr;Verr;Verr;Verr;Akkermansia              3.38509598 12.04318  12.02375003 5.252691e-04 1.184909e-03
## Firm;Clos;Clos;Veil;Anaerovibrio            -4.27357100 10.81513  11.87006605 5.704339e-04 1.257547e-03
## Firm;Clos;Clos;[Mog;                        -1.29985101 11.24358  11.51046762 6.920537e-04 1.491760e-03
## Prot;Gamm;Ente;Ente;Escherichia             -2.35944545 10.24774  10.45739212 1.221596e-03 2.575975e-03
## Firm;Baci;Lact;;                            -4.95409409 11.54658   9.76657418 1.777133e-03 3.667700e-03
## Bact;Bact;Bact;Rike;Alistipes                2.48192210 10.31506   9.43945799 2.123662e-03 4.291568e-03
```

```
## Firm;Clos;Clos;Lach;[Ruminococcus]          1.05281092 13.43032   9.34377585 2.237441e-03 4.429219e-03
## Tene;Moll;Anae;Anae;                       -2.29372775 10.30114   8.88446062 2.876080e-03 5.579595e-03
## WPS-;;;;                                   -2.97214709 10.39551   8.57740871 3.403594e-03 6.473502e-03
## Firm;Clos;Clos;Lach;Roseburia               1.05101162 13.75258   8.11472152 4.390722e-03 8.142730e-03
## Acti;Cori;Cori;Cori;Slackia                -2.36466673 10.40844   8.09077215 4.449120e-03 8.142730e-03
## Tene;Moll;Anae;Anae;Anaeroplasma           -2.78078958 10.47215   7.96432276 4.770834e-03 8.569831e-03
## Firm;Clos;Clos;Lach;Blautia                 0.72045282 15.17636   7.63357063 5.729177e-03 1.010418e-02
## Firm;Clos;Clos;Rumi;Ruminococcus            0.75058108 15.29530   6.68434842 9.726307e-03 1.684735e-02
## Firm;Clos;;;                               -1.87724472 10.36803   6.61915923 1.008874e-02 1.716857e-02
## Firm;Baci;Lact;Stre;Streptococcus          -1.61800715 13.37950   5.92138965 1.495816e-02 2.501623e-02
## Tene;Moll;RF39;;                           -1.53793813 12.75506   5.47736801 1.926424e-02 3.120959e-02
## Firm;Baci;Lact;Ente;Enterococcus           -3.66744194 10.91133   5.47368332 1.930490e-02 3.120959e-02
## Firm;Clos;Clos;Pept;Peptococcus            -1.82871653 10.29986   4.95078755 2.607879e-02 4.146955e-02
## Bact;Bact;Bact;RF16;                       -3.97575483 10.48303   4.63625196 3.130330e-02 4.897451e-02
## Firm;Clos;Clos;Clos;02d06                  -1.42387915 10.26342   4.16238775 4.133110e-02 6.351720e-02
## Lent;[Len;Vict;Vict;                       -2.15695240 10.78430   4.10901760 4.265513e-02 6.351720e-02
## Firm;Baci;Lact;Stre;Lactococcus            -2.49812772 11.04892   4.10038888 4.287335e-02 6.351720e-02
## Prot;Beta;Burk;Coma;                       -3.30679201 10.44189   4.08685896 4.321789e-02 6.351720e-02
## Acti;Acti;Acti;Acti;Actinomyces            -1.14591502 10.27099   4.05017481 4.416677e-02 6.394294e-02
## Prot;Gamm;Ente;Ente;Citrobacter            -3.05982720 10.92203   3.97858118 4.608236e-02 6.573514e-02
## Firm;Clos;Clos;Veil;Phascolarctobacterium -1.02024278 12.57425   3.80103910 5.122079e-02 7.200603e-02
## Firm;Clos;Clos;Lach;Anaerostipes            1.00288656 10.86038   3.44241559 6.354273e-02 8.805206e-02
## Firm;Clos;Clos;Clos;SMB53                  -1.19187994 11.32824   3.21493465 7.296910e-02 9.969019e-02
## Firm;Clos;Clos;Chri;                        0.88884465 12.95898   2.40541909 1.209157e-01 1.629004e-01
## Acti;Acti;Bifi;Bifi;Bifidobacterium         1.08881193 14.14257   2.33191270 1.267461e-01 1.648830e-01
## Firm;Clos;Clos;Lach;Lachnospira             0.59493199 13.41375   2.32454818 1.273472e-01 1.648830e-01
## Firm;Erys;Erys;Erys;                        0.66507246 13.21480   2.32284257 1.274869e-01 1.648830e-01
## Acti;Cori;Cori;Cori;Collinsella            -0.84476705 11.22045   2.29838241 1.295088e-01 1.652941e-01
## Firm;Clos;Clos;Veil;Dialister              -0.95827624 14.07699   2.24952139 1.336557e-01 1.683715e-01
## Firm;Clos;Clos;Rumi;Faecalibacterium       -0.34459812 15.82647   2.07295693 1.499306e-01 1.864521e-01
## Firm;Clos;Clos;Lach;Dorea                   0.39934297 13.22003   2.05009600 1.521966e-01 1.868743e-01
## Eury;Meth;Meth;Meth;Methanobrevibacter     -1.05680994 10.56789   1.95338640 1.622224e-01 1.966947e-01
## Firm;Clos;Clos;Lach;Epulopiscium           -0.57767719 10.21051   0.86912167 3.511986e-01 4.205711e-01
## Firm;Clos;Clos;Rumi;Oscillospira           -0.27159583 13.82903   0.76027176 3.832435e-01 4.533490e-01
## Prot;Alph;RF32;;                           -0.72972807 11.56598   0.60242662 4.376537e-01 5.114748e-01
## Tene;RF3;ML61;;                             0.86013439 10.69928   0.44995421 5.023567e-01 5.801024e-01
## Firm;Clos;Clos;Veil;Megasphaera             0.97801446 12.49381   0.43405063 5.100080e-01 5.820092e-01
## Lent;[Len;Vict;Vict;Victivallis            -0.60254524 10.23983   0.41260986 5.206478e-01 5.872423e-01
## Prot;Beta;Burk;Oxal;Oxalobacter            -0.32009129 10.21144   0.30861819 5.785294e-01 6.450270e-01
## Firm;Clos;Clos;Deha;Dehalobacterium         0.30530808 10.18989   0.27646094 5.990303e-01 6.602947e-01
## Firm;Baci;Turi;Turi;Turicibacter            0.37710760 11.20289   0.23891530 6.249906e-01 6.811696e-01
## Firm;Clos;Clos;Rumi;                       -0.08491411 17.26294   0.20725753 6.489255e-01 6.993974e-01
## Firm;Clos;Clos;;                           -0.08808477 15.79517   0.19199673 6.612599e-01 7.018220e-01
## Firm;Erys;Erys;Erys;[Eubacterium]           0.22285152 13.55286   0.18059500 6.708624e-01 7.018220e-01
## Firm;Clos;Clos;Lach;Lachnobacterium         0.25065694 10.90059   0.17825073 6.728809e-01 7.018220e-01
## Firm;Erys;Erys;Erys;Coprobacillus           0.18971570 10.29425   0.07214812 7.882344e-01 8.133908e-01
## Firm;Clos;Clos;Veil;                        0.12247904 11.87426   0.01701154 8.962276e-01 9.034640e-01
## Firm;Clos;Clos;Pept;                        0.07277174 12.09818   0.01616665 8.988232e-01 9.034640e-01
## Prot;Delt;Desu;Desu;Desulfovibrio           0.09451716 11.49044   0.01471045 9.034640e-01 9.034640e-01
```

Write the results to a tab-delimited file (to open in Excel) with:

```
write.table(topTags(result, n=Inf)$table, file='edgeR_results.txt',sep='\t',quote=FALSE, col.names=NA)
```

How many genera were significantly associated with the USA?

```
sum(topTags(result,n=Inf)$table$FDR <= 0.05)
```

```
## [1] 62
```

Non-parametric tests

Whenever we do not need to control for confounding variables, we can use non-parametric tests. These are the safest because they don't rely on any null distribution (e.g. normal). They typically consider only the **ranks** of values (order of values), not the actual values themselves.

For testing differences between two categories, we can use the Mann-Whitney U test, sometimes called the Wilcoxon Signed Rank test or Wilcoxon Rank Sum test. There are slight differences between these tests. Here we will test the difference in **Prevotella** abundance between USA and non-USA. Make sure to use the relative abundances, not the absolute abundances.

```
wilcox.test(prevotella ~ is.USA, exact=FALSE)
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  prevotella by is.USA
## W = 934, p-value = 7.732e-10
## alternative hypothesis: true location shift is not equal to 0
```

```
# get the exact p-value
wilcox.test(prevotella ~ is.USA, exact=FALSE)$p.value
```

```
## [1] 7.732064e-10
```

We can do a test for differentiation across multiple categories, analogous to ANOVA, using the Kruskal-Wallis test.

```
kruskal.test(prevotella ~ map$COUNTRY)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  prevotella by map$COUNTRY
## Kruskal-Wallis chi-squared = 42.001, df = 2, p-value = 7.581e-10
```

For continuous variables, we can use Spearman correlation instead of Pearson correlation.

```
cor.test(prevotella, map$AGE, method='spearman', exact=FALSE)
```

```
##
##  Spearman's rank correlation rho
##
## data:  prevotella and map$AGE
## S = 60496, p-value = 0.033
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##        rho
## -0.2628328
```