R and Bioconductor Assignment

Stephen Kanyerezi

6/2/2021

Contents

1 Question 1

Import the data described above into R, provide descriptive summaries of the subject data (using appropriate graphics and statistical summary measures) given in the diabimmune_16s_t1d_metadata.csv file. In addition, use appropriate test(s) to check for association/independency between disease status and other variables (delivery mode, gender and age). Note that age is given in days.

```
## Load the required packages
library(tidyverse)
library(phyloseq)
library(DESeq2)
```

```
sample_metadata <- read.csv("diabimmune_16s_t1d_metadata.csv", sep = ",", header = T) # import sample d
# Explore the data and have a glimpse of it
head(sample_metadata)
```

```
##   Sample_ID Subject_ID Case_Control Gender Delivery_Route Age_at_Collection
## 1    G36449    E001463      control   male        vaginal                62
## 2    G36034    E001463      control   male        vaginal                82
## 3    G36993    E001463      control   male        vaginal               124
## 4    G35523    E001463      control   male        vaginal               153
## 5    G36450    E001463      control   male        vaginal               187
## 6    G36028    E001463      control   male        vaginal               213
```

```
tail(sample_metadata)
```

```
##     Sample_ID Subject_ID Case_Control Gender Delivery_Route Age_at_Collection
## 772    G36938    T026177      control female        vaginal               570
## 773    G36936    T026177      control female        vaginal               592
## 774    G36937    T026177      control female        vaginal               646
## 775    G35535    T026177      control female        vaginal               677
## 776    G35536    T026177      control female        vaginal               703
## 777    G35537    T026177      control female        vaginal               729
```

```
dim(sample_metadata)
```

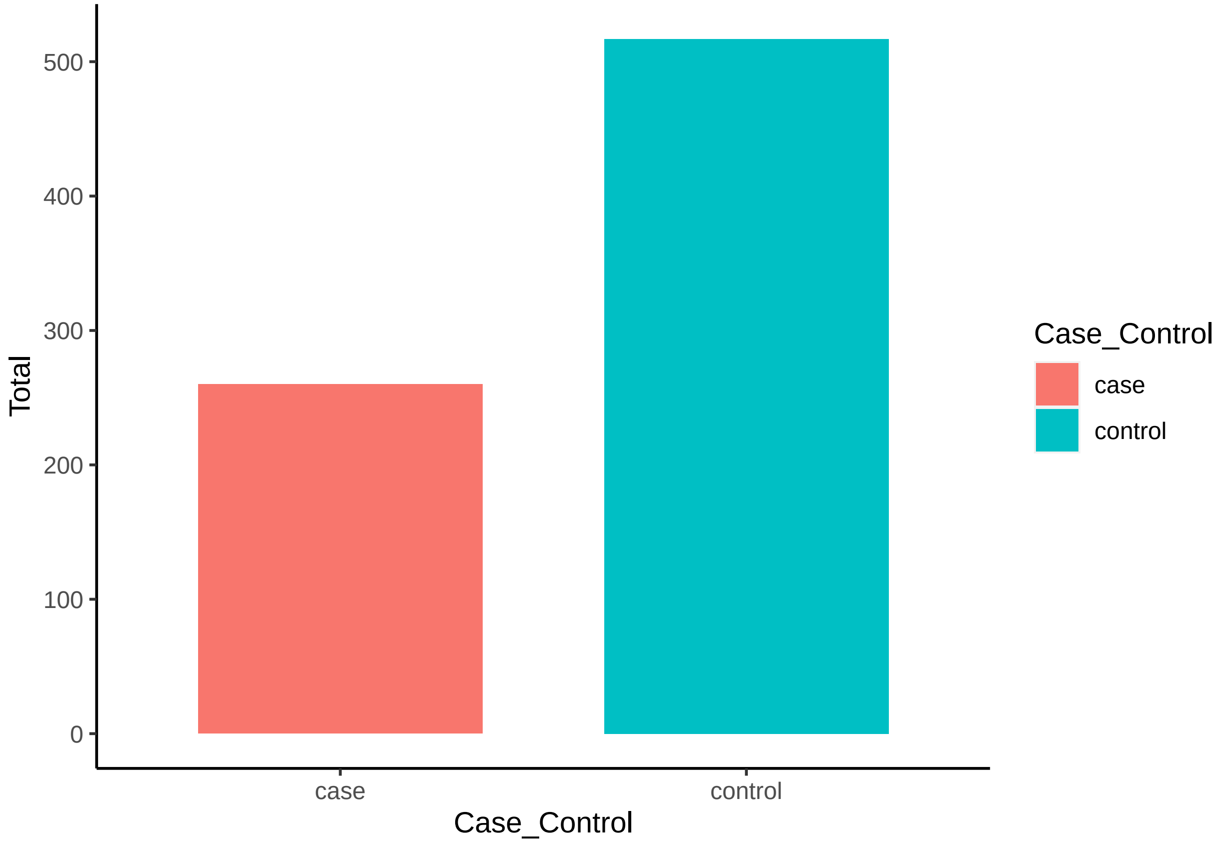
```
## [1] 777   6
```

1.1 How many were cases and controls

```
case_control_count <- sample_metadata %>%
  group_by(Case_Control) %>%
  dplyr::summarise(Total = n(), Percentage = (Total / nrow(sample_metadata)) * 100)
case_control_count
```

```
## # A tibble: 2 x 3
##   Case_Control Total Percentage
##   <chr>        <int>      <dbl>
## 1 case           260       33.5
## 2 control        517       66.5
```

```
ggplot(case_control_count, aes(x = Case_Control, y = Total, fill = Case_Control)) +
  geom_bar(stat = "identity", width = 0.7) +
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
        panel.background = element_blank(), axis.line = element_line(colour = "black"))
```

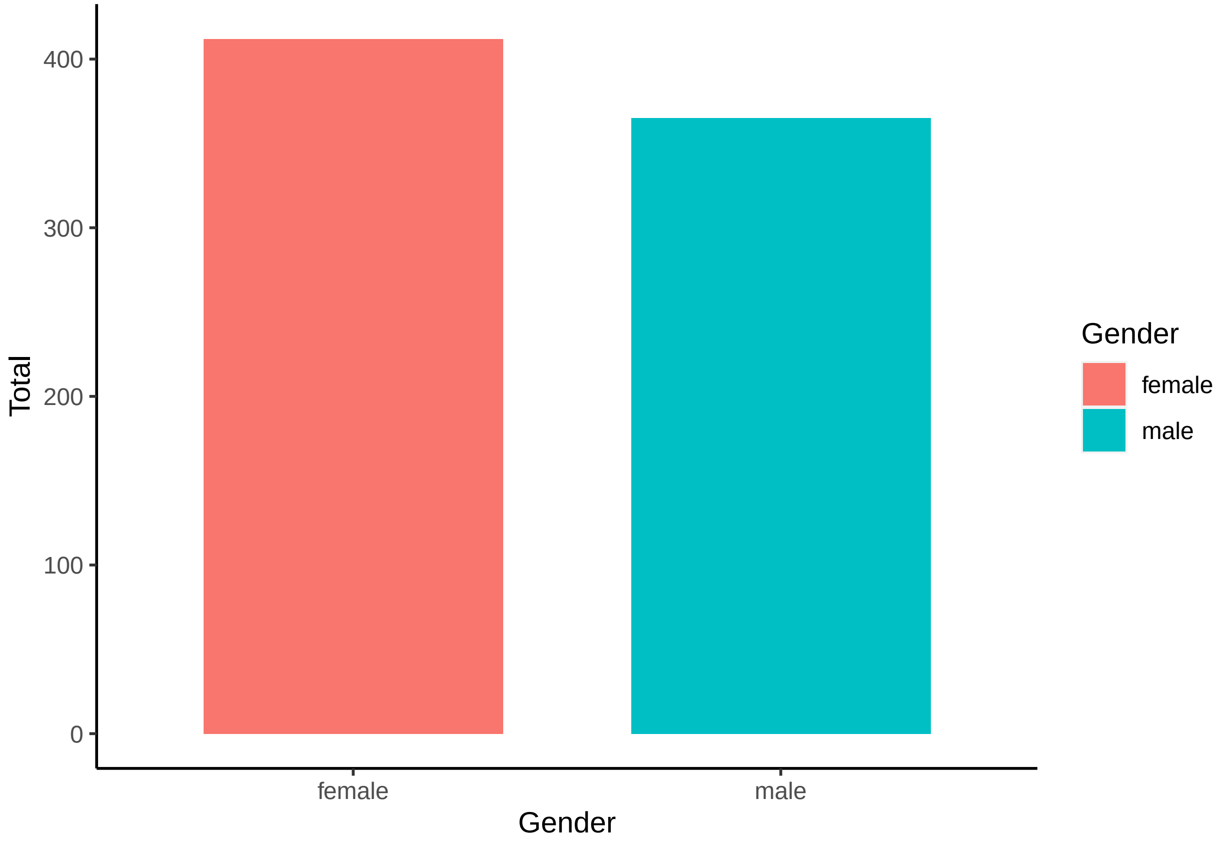


1.2 How many were males and females

```r
gender_counts <- sample_metadata %>%
  group_by(Gender) %>%
  summarise(Total = n(), Percentage = (Total / nrow(sample_metadata)) * 100)
gender_counts
```

```
## # A tibble: 2 x 3
##   Gender Total Percentage
##   <chr>  <int>      <dbl>
## 1 female   412       53.0
## 2 male     365       47.0
```

```r
ggplot(gender_counts, aes(x = Gender, y = Total, fill = Gender)) +
  geom_bar(stat = "identity", width = 0.7) +
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
        panel.background = element_blank(), axis.line = element_line(colour = "black"))
```

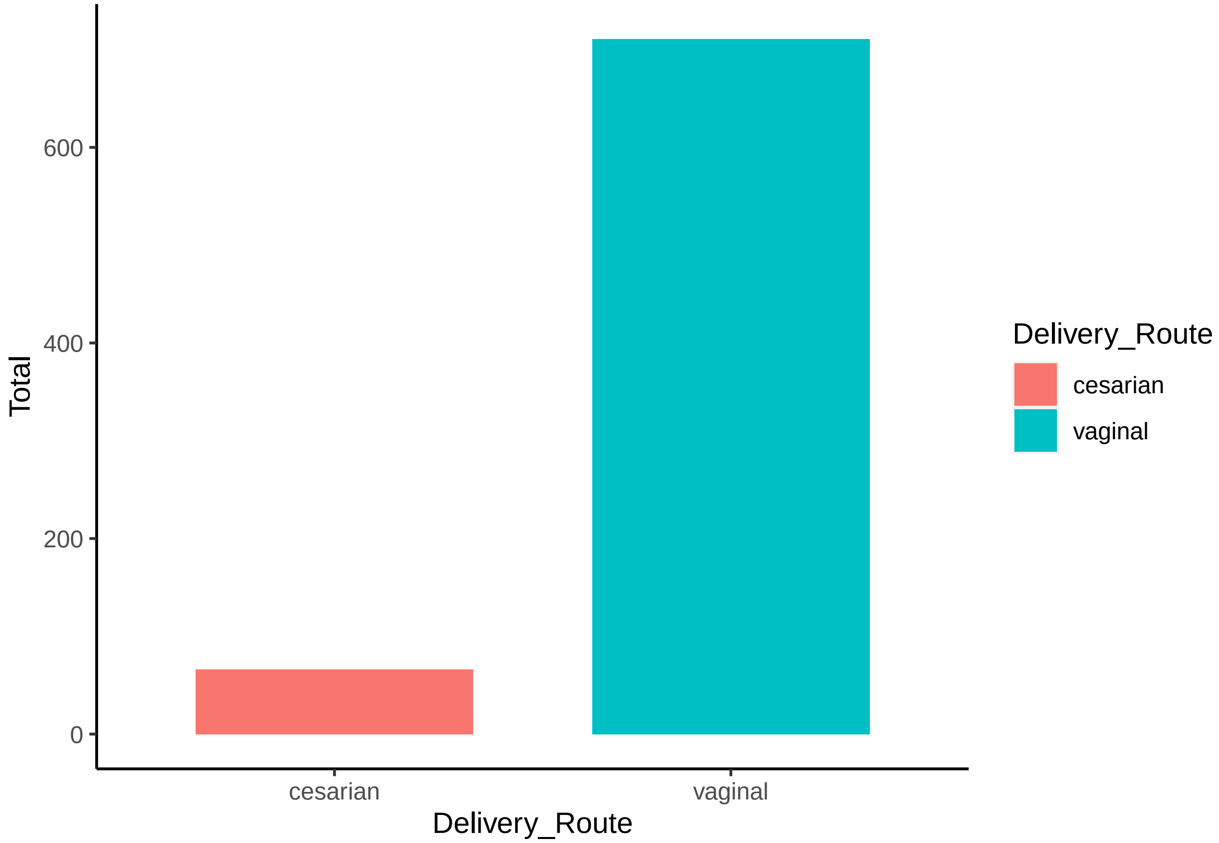


1.3 How many fall into each delivery route

```
delivery_route_count <- sample_metadata %>%
  group_by(Delivery_Route) %>%
  summarise(Total = n(), Percentage = (Total / nrow(sample_metadata)) * 100)
delivery_route_count
```

```
## # A tibble: 2 x 3
##   Delivery_Route Total Percentage
##   <chr>          <int>      <dbl>
## 1 cesarian          66       8.49
## 2 vaginal          711      91.5
```

```
ggplot(delivery_route_count, aes(x = Delivery_Route, y = Total, fill = Delivery_Route)) +
  geom_bar(stat = "identity", width = 0.7) +
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
        panel.background = element_blank(), axis.line = element_line(colour = "black"))
```



1.4 Cases and controls versus gender

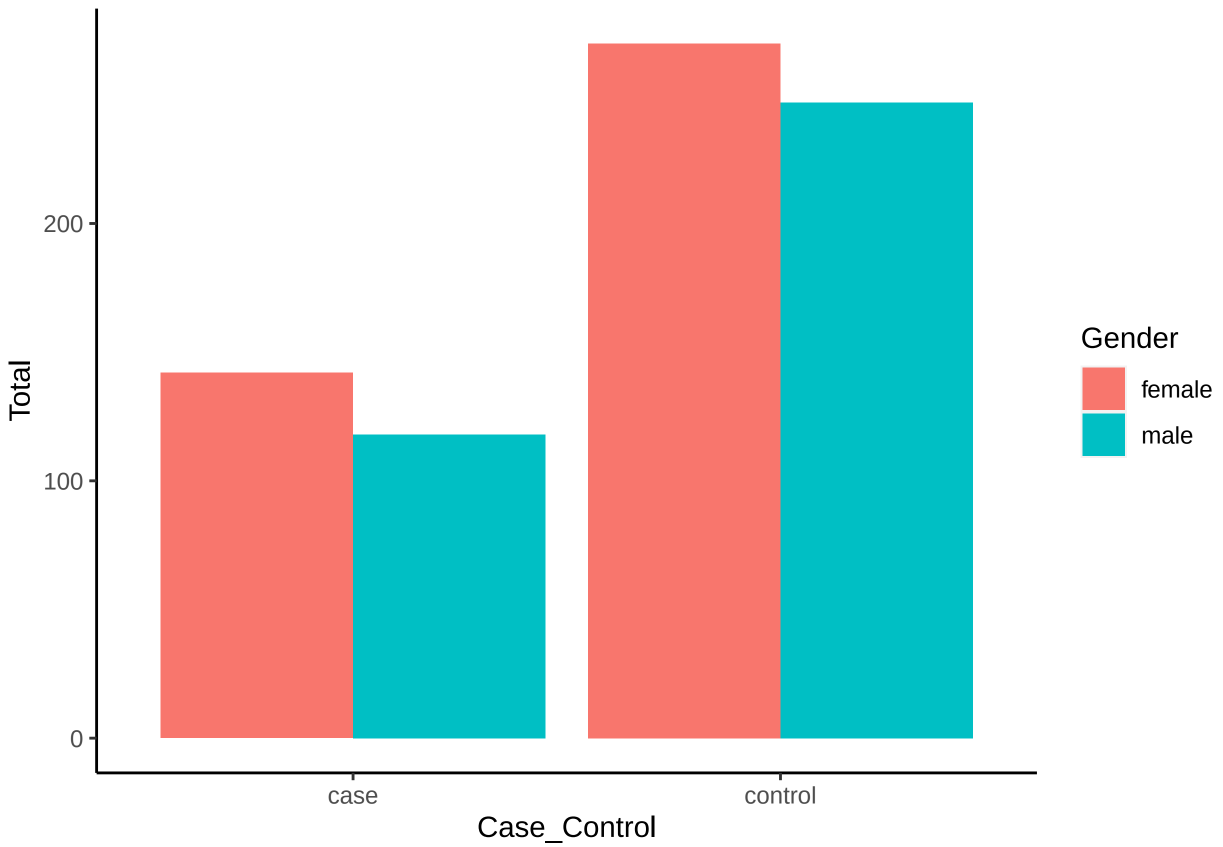
```
status_gender <- sample_metadata %>%
  group_by(Gender, Case_Control) %>%
  summarise(Total = n(), Percentage = (Total / nrow(sample_metadata)) * 100)
```

```
## `summarise()` has grouped output by 'Gender'. You can override using the `.groups` argument.
```

```
status_gender
```

```
## # A tibble: 4 x 4
## # Groups:   Gender [2]
##   Gender Case_Control Total Percentage
##   <chr>  <chr>        <int>      <dbl>
## 1 female case           142       18.3
## 2 female control        270       34.7
## 3 male   case           118       15.2
## 4 male   control        247       31.8
```

```
ggplot(status_gender, aes(fill=Gender, y=Total, x=Case_Control)) +
  geom_bar(position="dodge", stat="identity") +
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
        panel.background = element_blank(), axis.line = element_line(colour = "black"))
```



1.4.1 Relationship between status and gender

Here we use the chi-square test to determine whether the status and gender are independent or dependent of each other.

Null hypothesis: Status and gender are independent

Alternate hypothesis: Status and gender relate to each other Note: We set our significance level at 0.05. So, if we get a p-value less than 0.05, we shall reject the null hypothesis, otherwise, we fail to reject it.

```
gen_mat <- table(sample_metadata$Case_Control, sample_metadata$Gender) # create a contigency table
gen_mat
```

```
##
##           female male
##   case       142  118
##   control    270  247
```

```
chisq.test(gen_mat)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  gen_mat
## X-squared = 0.30687, df = 1, p-value = 0.5796
```

Since P-value is greater than 0.05 - our significance level, we fail to reject the null hypothesis and conclude that status and gender are related

1.5 Cases and controls versus delivery route

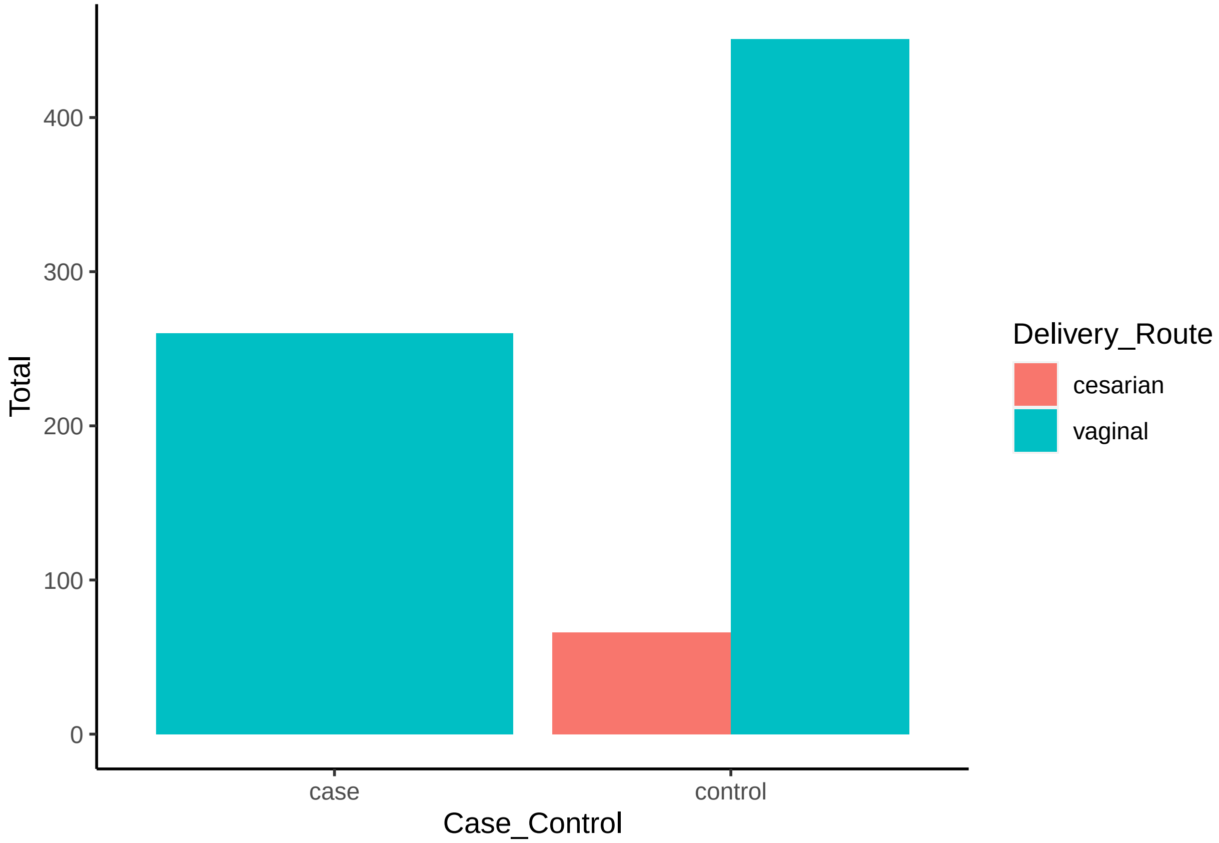
```
status_route <- sample_metadata %>%
  group_by(Case_Control, Delivery_Route) %>%
  summarise(Total = n(), Percentage = (Total / nrow(sample_metadata)) * 100)
```

```
## `summarise()` has grouped output by 'Case_Control'. You can override using the `.groups` argument.
```

```
status_route
```

```
## # A tibble: 3 x 4
## # Groups:   Case_Control [2]
##   Case_Control Delivery_Route Total Percentage
##   <chr>        <chr>          <int>      <dbl>
## 1 case         vaginal          260      33.5
## 2 control      cesarian          66       8.49
## 3 control      vaginal          451      58.0
```

```
ggplot(status_route, aes(fill=Delivery_Route, y=Total, x=Case_Control)) +
  geom_bar(position="dodge", stat="identity") +
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
        panel.background = element_blank(), axis.line = element_line(colour = "black"))
```



1.5.1 Relationship between status and delivery route

Here we use the chi-square test to determine whether the status and delivery routes are independent or dependent of each other.

Null hypothesis: Status and delivery route are independent

Alternate hypothesis: Status and delivery route relate to each other Note: We set our significance level at 0.05. So, if we get a p-value less than 0.05, we shall reject the null hypothesis, otherwise, we fail to reject it.

```
del_mat <- table(sample_metadata$Case_Control, sample_metadata$Delivery_Route)
del_mat
```

```
##
##           cesarian vaginal
##   case           0     260
##   control       66     451
```

```
chisq.test(del_mat)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  del_mat
## X-squared = 34.649, df = 1, p-value = 3.949e-09
```

Since P-value is less than 0.05 - our significance level, we reject the null hypothesis and conclude that status and delivery route are independent of each other

1.6 Cases and controls, gender plus delivery route

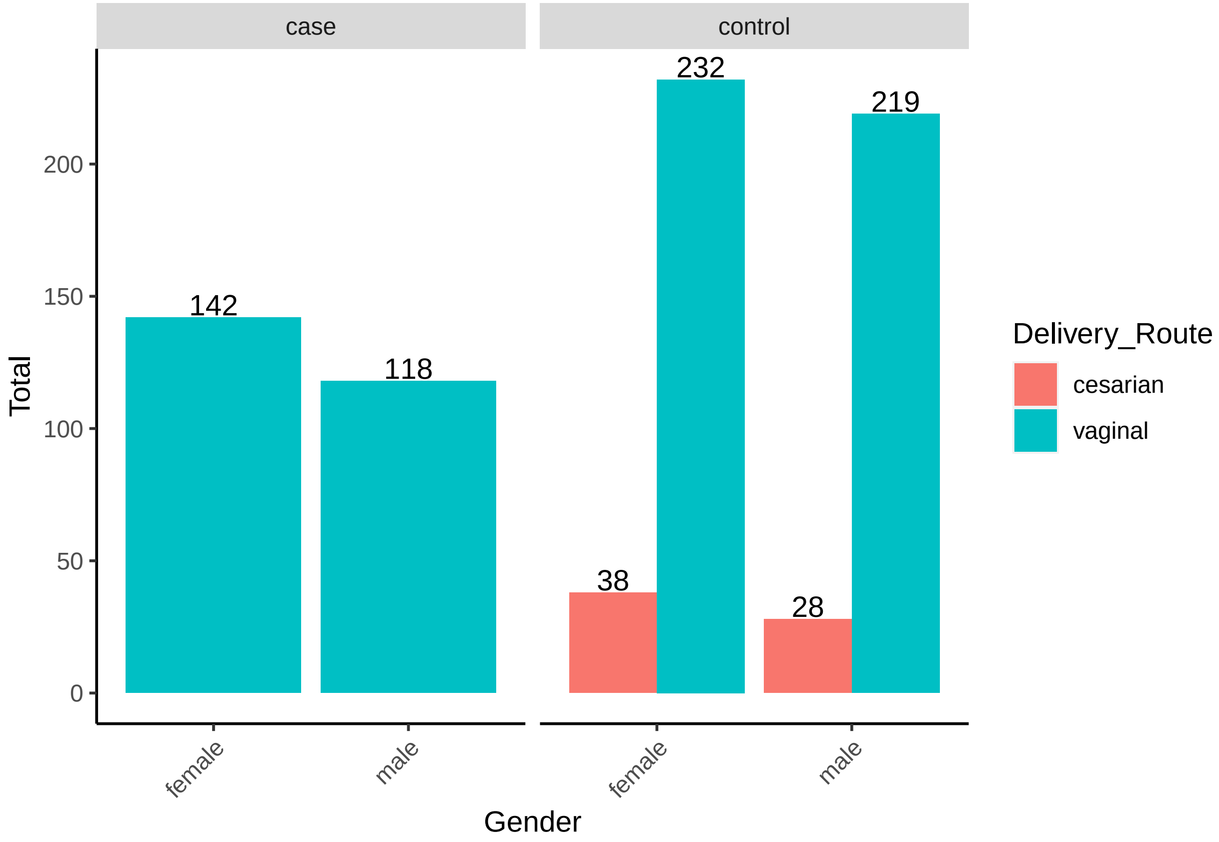
```
status_gender_delivery <- sample_metadata %>%
  group_by(Gender, Case_Control, Delivery_Route) %>%
  summarise(Total = n(), Percentage = (Total / nrow(sample_metadata)) * 100)
```

```
## `summarise()` has grouped output by 'Gender', 'Case_Control'. You can override using the `.groups` a
```

```
status_gender_delivery
```

```
## # A tibble: 6 x 5
## # Groups:   Gender, Case_Control [4]
##   Gender Case_Control Delivery_Route Total Percentage
##   <chr>  <chr>        <chr>          <int>      <dbl>
## 1 female case         vaginal          142      18.3
## 2 female control      cesarian          38       4.89
## 3 female control      vaginal          232      29.9
## 4 male   case         vaginal          118      15.2
## 5 male   control      cesarian          28       3.60
## 6 male   control      vaginal          219      28.2
```

```
p <- ggplot(status_gender_delivery, aes(x = Gender, y = Total, fill = Delivery_Route)) +
  geom_bar(position="dodge", stat="identity") +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(), panel.background = element_blank(), axis.line =
          element_line(colour = "black"), axis.text.x = element_text(angle = 45, hjust = 1))
p + facet_grid(. ~ Case_Control) + geom_text(aes(label = Total), vjust = -0.1, position =
   position_dodge(width = 0.9)) +
    labs(x = "Gender")
```



1.7 Age

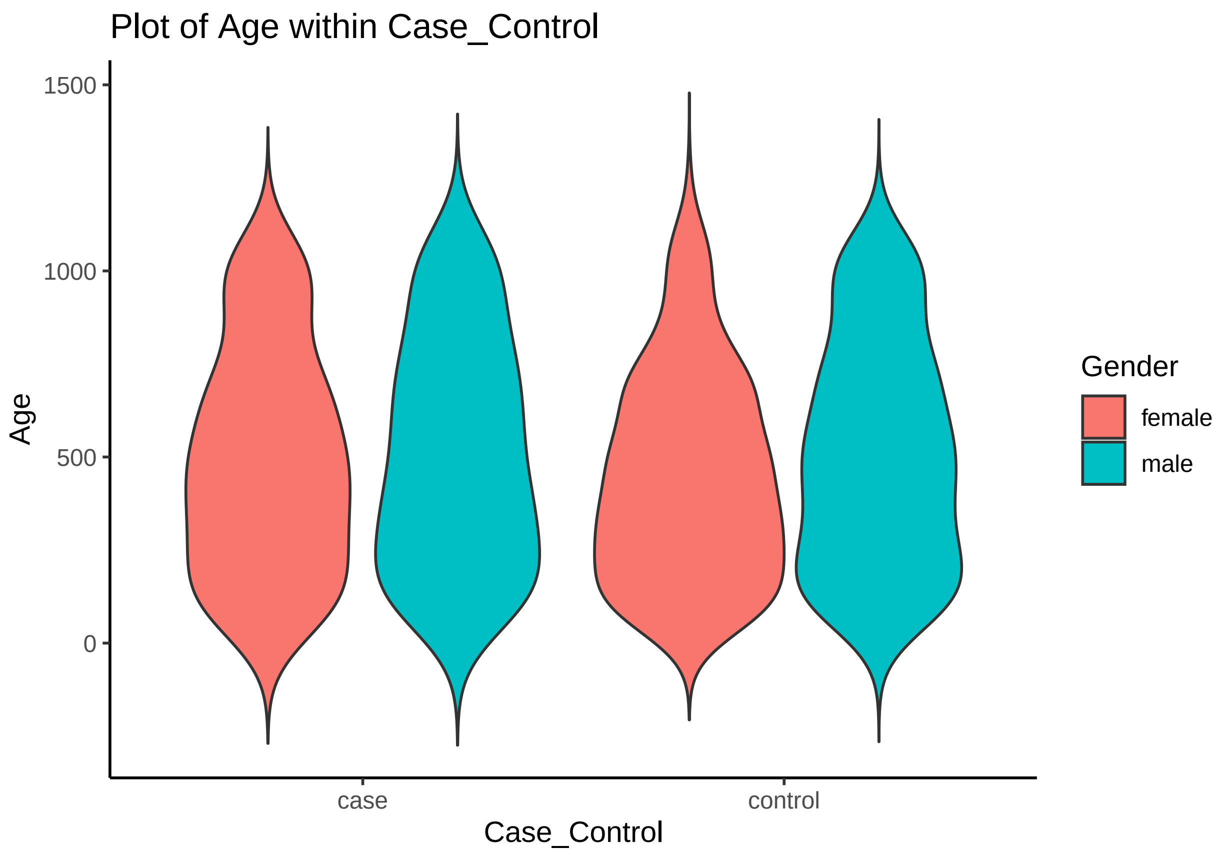
```
# calculate the decriptive statistics of age
summary(sample_metadata$Age_at_Collection)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##     6.0   229.0   452.0   482.9   702.0  1233.0
```

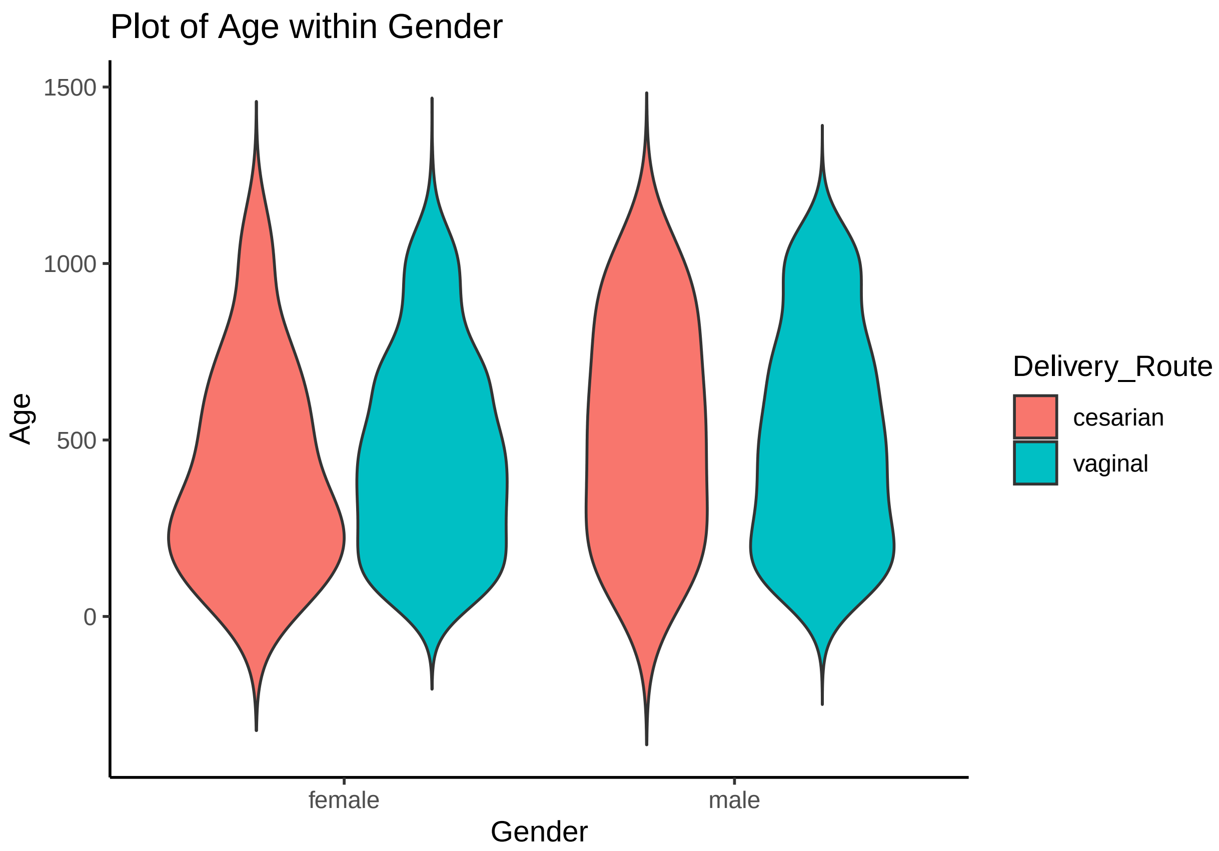
```
# find the standard deviation within age
sd(sample_metadata$Age_at_Collection)
```

```
## [1] 294.7245
```

```
# do a violin plot to show age vs case_control and fill with gender
ggplot(sample_metadata, aes(y = Age_at_Collection, x = Case_Control, fill = Gender)) +
  geom_violin(trim = FALSE) +
  theme_classic() +
  labs(title="Plot of Age within Case_Control",x="Case_Control", y = "Age")
```



```r
# do a violin plot to show age vs case_control and fill with delivery_route
ggplot(sample_metadata, aes(y = Age_at_Collection, x = Gender, fill = Delivery_Route)) +
  geom_violin(trim = FALSE) +
  theme_classic() +
  labs(title="Plot of Age within Gender",x="Gender", y = "Age")
```



2 Question 2

Using phyloseq, create a phyloseq object. This will comprise the OTU abundance, taxonomy (provided in the .txt file) and sample data (provided in the .csv file)

```
# since the provided txt file couldn't be read in by the import_biom function, I decided to divided it
otut <- read.table("otu_table") # import otu table
#head(otut, n = 1)

taxat <- read.table("new_taxa_table") # import taxa table
#head(taxat)
dim(taxat)
```

```
## [1] 2240    7
```

```
dim(otut)
```

```
## [1] 2240  777
```

```
class(taxat)
```

```
## [1] "data.frame"
```

```
class(otut)
```

```
## [1] "data.frame"
```

```r
otut_mat <- as.matrix(otut) # convert the otu table into a matrix
class(otut_mat)
```

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

```r
taxat_mat <- as.matrix(taxat) # convert the taxa table into a matrix
class(taxat_mat)
```

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

```r
#head(taxat_mat)
taxat_mat_sub <- gsub(";", "", taxat_mat) # remove ; from the columns of taxa table
#head(taxat_mat_sub)

# Our row names are ending with ; and this makes them different from the naming in the otu table
# so we need to remove the ;
mat_names <- row.names(taxat_mat_sub) # extract the rownames and store them in the mat_names object
new_naam <- gsub(";", "", mat_names) # remove ; from the mat_names and store them to new_naam
#new_naam
row.names(taxat_mat_sub) <- new_naam # now change the row names of taxat_mat_sub to the names in "new_n


OTU <- otu_table(otut_mat, taxa_are_rows = TRUE)
TAX <- tax_table(taxat_mat_sub)
#OTU
#TAX
samp_data <- column_to_rownames(sample_metadata, var="Sample_ID")
samp_data <- sample_data(samp_data)
physeq <- phyloseq(OTU, TAX, samp_data)
physeq
```

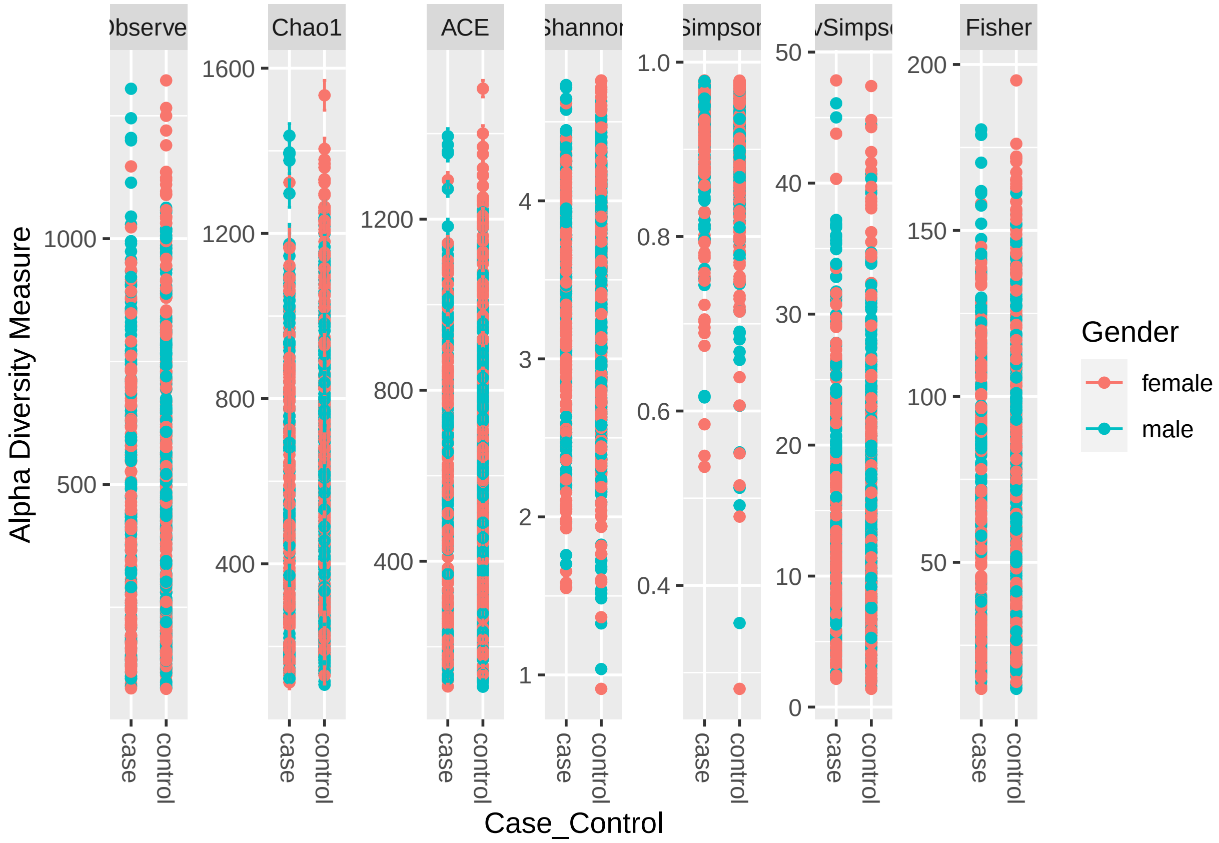
```
## phyloseq-class experiment-level object
## otu_table()   OTU Table:         [ 2240 taxa and 777 samples ]
## sample_data() Sample Data:       [ 777 samples by 5 sample variables ]
## tax_table()   Taxonomy Table:    [ 2240 taxa by 7 taxonomic ranks ]
```

3 Question 3

Generate Alpha diversity plots and ordination plots. Examine any observed patterns by delivery mode, gender and disease status.

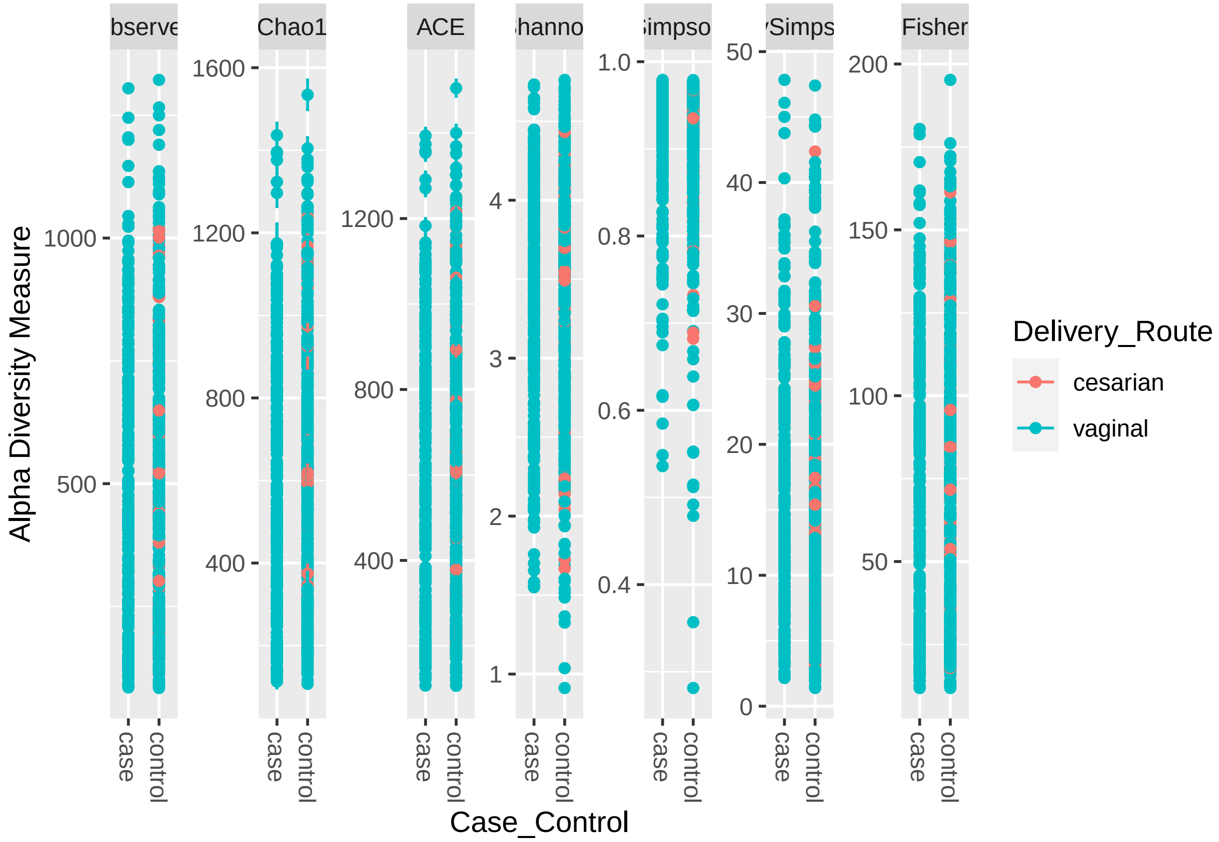
3.1 Alpha diversity comparing the gender in cases and controls

```r
plot_richness(physeq = physeq, x = "Case_Control", color = "Gender")
```



3.2 Alpha diversity comparing the delivery routes in cases and controls

```r
plot_richness(physeq = physeq, x = "Case_Control", color = "Delivery_Route")
```



3.3 Ordination plots

3.3.1 Case_Control and Delivery_Route

```
library(plyr)
```

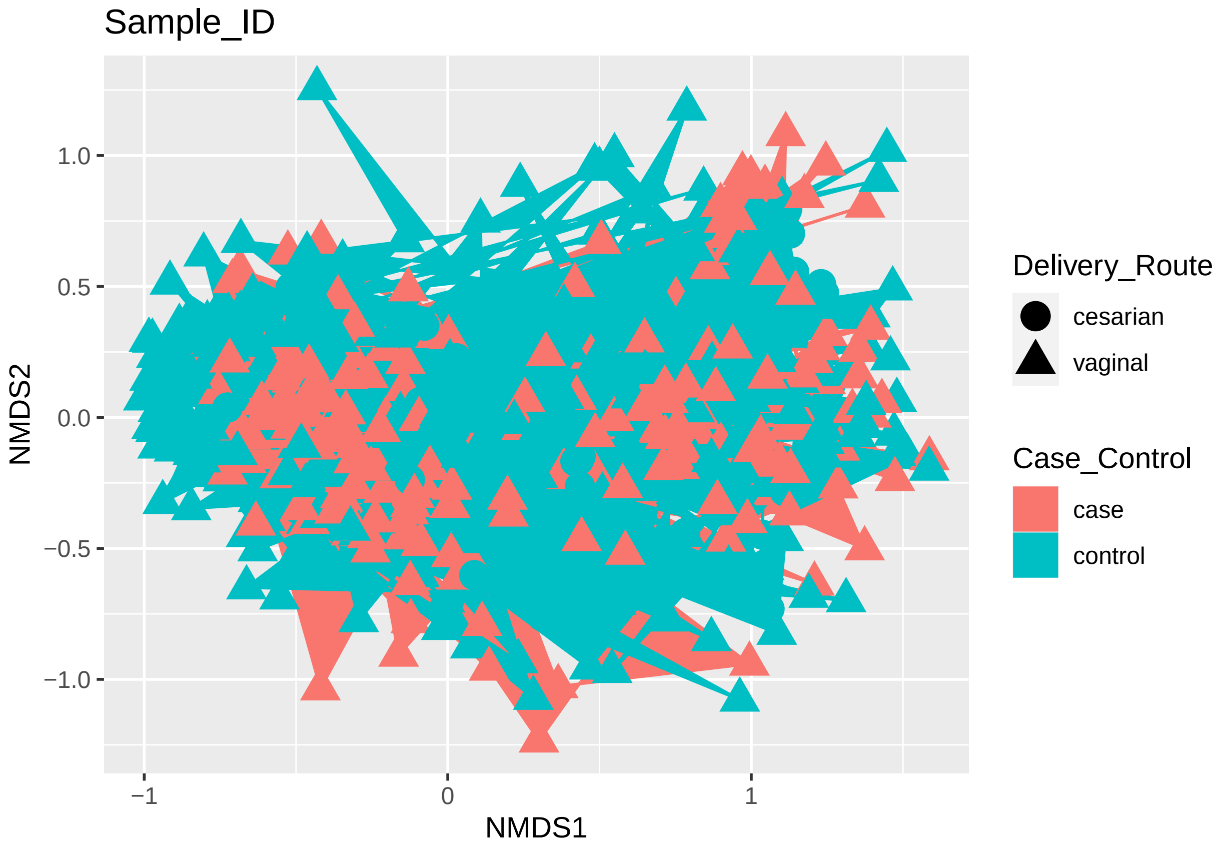
```
## ------------------------------------------------------------------------------

## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)

## ------------------------------------------------------------------------------

##
## Attaching package: 'plyr'

## The following object is masked from 'package:matrixStats':
##
##     count

## The following object is masked from 'package:IRanges':
##
##     desc

## The following object is masked from 'package:S4Vectors':
##
```

```
##     rename

## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize

## The following object is masked from 'package:purrr':
##
##     compact
```

```
GP.ord <- ordinate(physeq, "NMDS", "bray") # calculate the pairwise matrix
```

```
## Square root transformation
## Wisconsin double standardization
## Run 0 stress 0.181114
## Run 1 stress 0.1895064
## Run 2 stress 0.1849785
## Run 3 stress 0.1886943
## Run 4 stress 0.188767
## Run 5 stress 0.1813633
## ... Procrustes: rmse 0.01358951  max resid 0.1101631
## Run 6 stress 0.1882243
## Run 7 stress 0.1925373
## Run 8 stress 0.1865819
## Run 9 stress 0.1828949
## Run 10 stress 0.1901574
## Run 11 stress 0.1926256
## Run 12 stress 0.1823703
## Run 13 stress 0.1830457
## Run 14 stress 0.1906077
## Run 15 stress 0.1843107
## Run 16 stress 0.1854504
## Run 17 stress 0.1858675
## Run 18 stress 0.1897204
## Run 19 stress 0.183785
## Run 20 stress 0.1842714
## *** No convergence -- monoMDS stopping criteria:
##      6: no. of iterations >= maxit
##      7: stress ratio > sratmax
##      7: scale factor of the gradient < sfgrmin
```

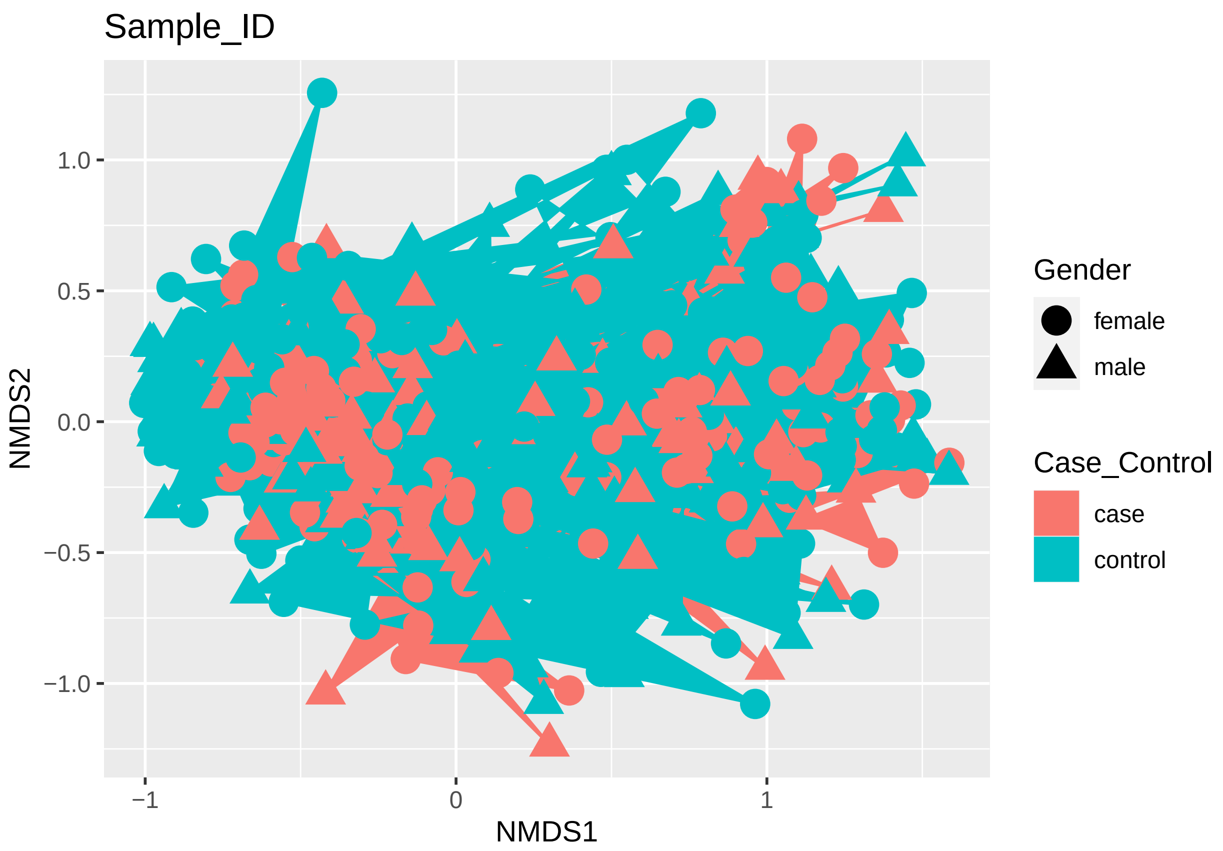
```
p2 = plot_ordination(physeq, GP.ord, type="Sample_ID", color="Case_Control", shape="Delivery_Route")
p2 + geom_polygon(aes(fill=Case_Control)) + geom_point(size=5) + ggtitle("Sample_ID")
```



According to this plot, there is no clear difference among cases and controls while basing on the Delivery route

3.3.2 Case_Control and Gender

```
p2 = plot_ordination(physeq, GP.ord, type="Sample_ID", color="Case_Control", shape="Gender")
p2 + geom_polygon(aes(fill=Case_Control)) + geom_point(size=5) + ggtitle("Sample_ID")
```



According to this plot, there is no clear difference between gender

4 Question 4

Perform differential abundance using DEseq2

4.1 Differential abundance using Case_Control + Delivery_Route as the design

```
library(DESeq2)
diagdds = phyloseq_to_deseq2(physeq, ~ Case_Control + Delivery_Route)
```

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

```
## Warning in DESeqDataSet(se, design = design, ignoreRank): some variables in
## design formula are characters, converting to factors
```

```
# calculate geometric means prior to estimate size factors
gm_mean = function(x, na.rm=TRUE){
  exp(sum(log(x[x > 0]), na.rm=na.rm) / length(x))
}
geoMeans = apply(counts(diagdds), 1, gm_mean)
diagdds = estimateSizeFactors(diagdds, geoMeans = geoMeans)
diagdds = DESeq(diagdds, fitType="local")
```

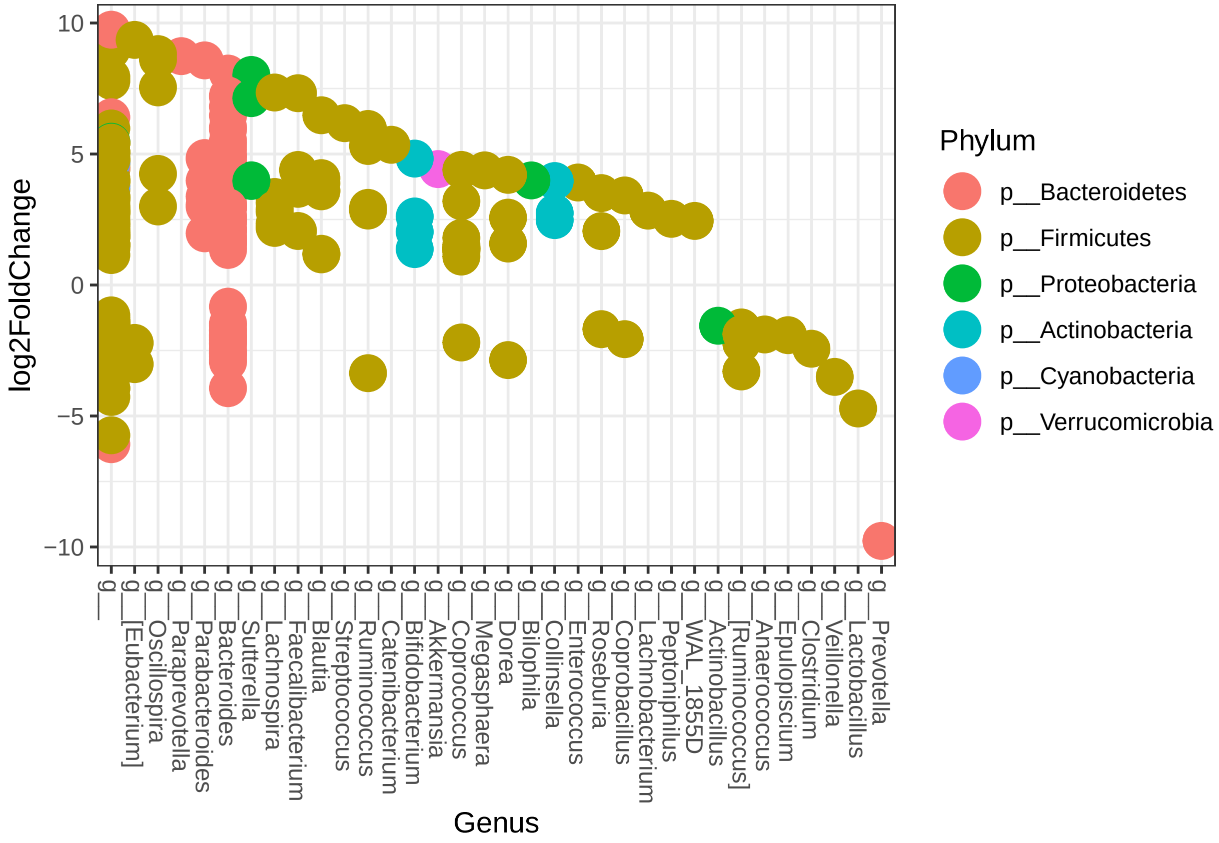
```
## using pre-existing size factors

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing

## -- replacing outliers and refitting for 1083 genes
## -- DESeq argument 'minReplicatesForReplace' = 7
## -- original counts are preserved in counts(dds)

## estimating dispersions

## fitting model and testing
```

```
res = results(diagdds, cooksCutoff = FALSE)
alpha = 0.01
sigtab = res[which(res$padj < alpha), ]
sigtab = cbind(as(sigtab, "data.frame"), as(tax_table(physeq)[rownames(sigtab), ], "matrix"))
head(sigtab)
```

```
##          baseMean log2FoldChange     lfcSE      stat       pvalue         padj
## 3211875  20.27438       1.820295 0.5452404  3.338518 8.422642e-04 5.536848e-03
## 172777   23.72436      -1.941592 0.3537056 -5.489288 4.035569e-08 8.432215e-07
## 189920   66.23363       3.064838 0.3672559  8.345239 7.107029e-17 5.374268e-15
## 3275562 138.16705       4.405241 0.5717719  7.704543 1.313119e-14 7.723088e-13
## 184990    8.68079       1.942984 0.3571946  5.439566 5.341058e-08 1.073620e-06
## 306299    8.60579       1.992251 0.4794025  4.155695 3.243001e-05 3.198686e-04
##             Domain.          Phylum.          Class.           Order.
## 3211875 k__Bacteria p__Bacteroidetes c__Bacteroidia o__Bacteroidales
## 172777  k__Bacteria p__Bacteroidetes c__Bacteroidia o__Bacteroidales
## 189920  k__Bacteria p__Bacteroidetes c__Bacteroidia o__Bacteroidales
## 3275562 k__Bacteria    p__Firmicutes  c__Clostridia o__Clostridiales
## 184990  k__Bacteria    p__Firmicutes  c__Clostridia o__Clostridiales
## 306299  k__Bacteria    p__Firmicutes  c__Clostridia o__Clostridiales
##                    Family.         Genus. Species
## 3211875  f__Bacteroidaceae g__Bacteroides     s__
## 172777   f__Bacteroidaceae g__Bacteroides     s__
## 189920   f__Bacteroidaceae g__Bacteroides     s__
## 3275562 f__Lachnospiraceae            g__     s__
## 184990  f__Ruminococcaceae            g__     s__
## 306299  f__Lachnospiraceae            g__     s__
```

```
## Exploring OTUs that were significant
# library("ggplot2")
theme_set(theme_bw())
scale_fill_discrete <- function(palname = "Set1", ...) {
    scale_fill_brewer(palette = palname, ...)
}
# Phylum order
x = tapply(sigtab$log2FoldChange, sigtab$Phylum, function(x) max(x))
x = sort(x, TRUE)
sigtab$Phylum = factor(as.character(sigtab$Phylum), levels=names(x))
# Genus order
```

```
x = tapply(sigtab$log2FoldChange, sigtab$Genus, function(x) max(x))
x = sort(x, TRUE)
sigtab$Genus = factor(as.character(sigtab$Genus), levels=names(x))
ggplot(sigtab, aes(x=Genus, y=log2FoldChange, color=Phylum)) + geom_point(size=6) +
  theme(axis.text.x = element_text(angle = -90, hjust = 0, vjust=0.5))
```



4.2 Differential abundance using Case_control + Gender

```
diagdds = phyloseq_to_deseq2(physeq, ~ Case_Control + Gender)
```

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

```
## Warning in DESeqDataSet(se, design = design, ignoreRank): some variables in
## design formula are characters, converting to factors
```

```
# calculate geometric means prior to estimate size factors
gm_mean = function(x, na.rm=TRUE){
  exp(sum(log(x[x > 0]), na.rm=na.rm) / length(x))
}
geoMeans = apply(counts(diagdds), 1, gm_mean)
diagdds = estimateSizeFactors(diagdds, geoMeans = geoMeans)
diagdds = DESeq(diagdds, fitType="local")
```

```
## using pre-existing size factors
```

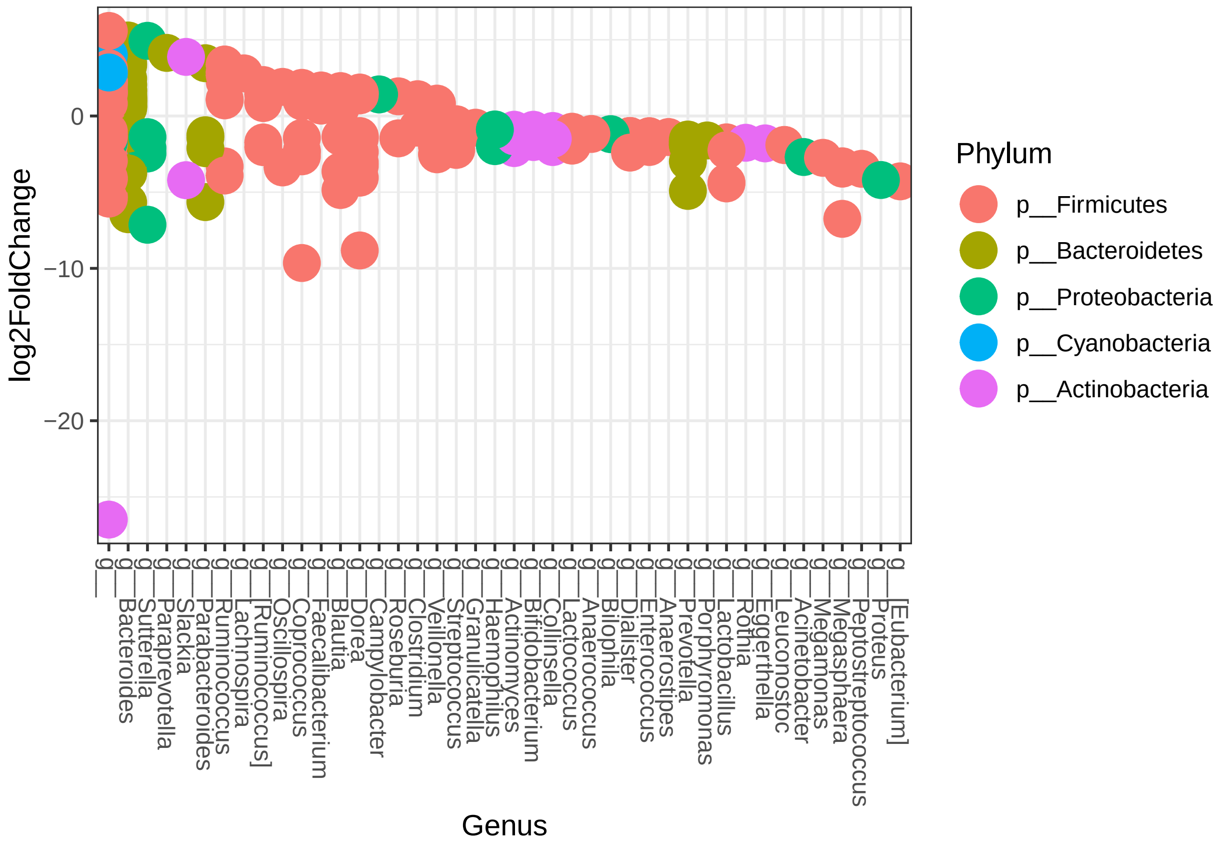
```
## estimating dispersions
## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing

## -- replacing outliers and refitting for 1077 genes
## -- DESeq argument 'minReplicatesForReplace' = 7
## -- original counts are preserved in counts(dds)

## estimating dispersions

## fitting model and testing
```

```r
res = results(diagdds, cooksCutoff = FALSE)
alpha = 0.01
sigtab = res[which(res$padj < alpha), ]
sigtab = cbind(as(sigtab, "data.frame"), as(tax_table(physeq)[rownames(sigtab), ], "matrix"))
head(sigtab)
```

```
##           baseMean log2FoldChange     lfcSE      stat       pvalue         padj
## 190162    8.913805      0.6581823 0.1654124  3.979038 6.919478e-05 7.611426e-04
## 134726    5.372019     -1.9538615 0.5285288 -3.696793 2.183406e-04 1.930376e-03
## 679245   11.846562     -2.0665739 0.4312619 -4.791923 1.651904e-06 3.064120e-05
## 4390365 446.543715      0.7178517 0.1900582  3.777011 1.587219e-04 1.494039e-03
## 189920   66.233626      0.6842066 0.2015487  3.394746 6.869222e-04 4.726024e-03
## 3275562 148.487742      1.5846228 0.3128092  5.065781 4.067296e-07 9.053323e-06
##             Domain.          Phylum.              Class.                 Order.
## 190162  k__Bacteria    p__Firmicutes      c__Clostridia      o__Clostridiales
## 134726  k__Bacteria    p__Firmicutes         c__Bacilli    o__Lactobacillales
## 679245  k__Bacteria    p__Firmicutes         c__Bacilli    o__Lactobacillales
## 4390365 k__Bacteria    p__Firmicutes c__Erysipelotrichi o__Erysipelotrichales
## 189920  k__Bacteria p__Bacteroidetes     c__Bacteroidia      o__Bacteroidales
## 3275562 k__Bacteria    p__Firmicutes      c__Clostridia      o__Clostridiales
##                         Family.            Genus. Species
## 190162      f__Lachnospiraceae       g__Blautia     s__
## 134726     f__Lactobacillaceae g__Lactobacillus     s__
## 679245     f__Lactobacillaceae g__Lactobacillus     s__
## 4390365 f__Erysipelotrichaceae              g__     s__
## 189920       f__Bacteroidaceae   g__Bacteroides     s__
## 3275562     f__Lachnospiraceae              g__     s__
```

```r
## Exploring OTUs that were significant
# library("ggplot2")
theme_set(theme_bw())
scale_fill_discrete <- function(palname = "Set1", ...) {
    scale_fill_brewer(palette = palname, ...)
}
# Phylum order
x = tapply(sigtab$log2FoldChange, sigtab$Phylum, function(x) max(x))
x = sort(x, TRUE)
sigtab$Phylum = factor(as.character(sigtab$Phylum), levels=names(x))
# Genus order
x = tapply(sigtab$log2FoldChange, sigtab$Genus, function(x) max(x))
x = sort(x, TRUE)
sigtab$Genus = factor(as.character(sigtab$Genus), levels=names(x))
```

```r
ggplot(sigtab, aes(x=Genus, y=log2FoldChange, color=Phylum)) + geom_point(size=6) +
  theme(axis.text.x = element_text(angle = -90, hjust = 0, vjust=0.5))
```



4.3 Differential abundance using Case_Control + Gender + Delivery_Route as the design

```r
diagdds = phyloseq_to_deseq2(physeq, ~ Case_Control + Delivery_Route + Gender)
```

```
## converting counts to integer mode

## Warning in DESeqDataSet(se, design = design, ignoreRank): some variables in
## design formula are characters, converting to factors
```

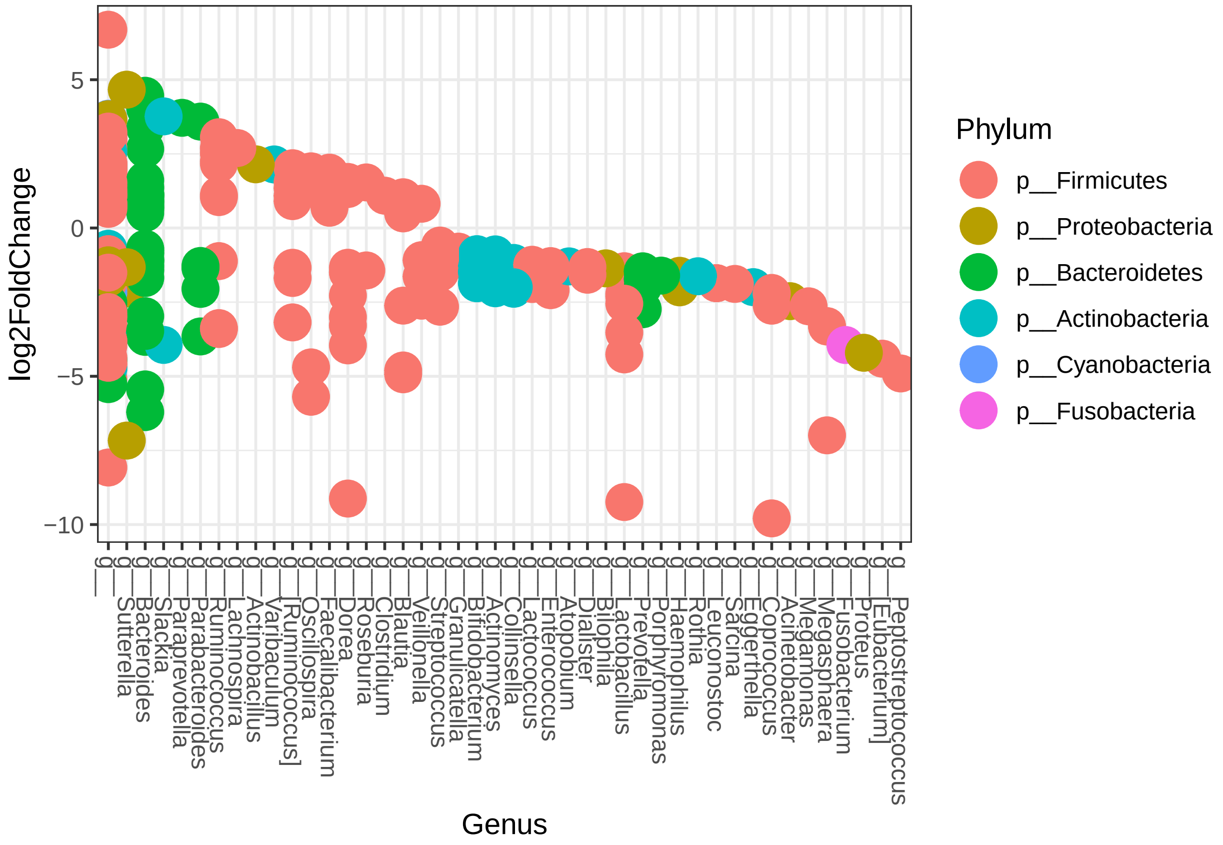
```r
# calculate geometric means prior to estimate size factors
gm_mean = function(x, na.rm=TRUE){
  exp(sum(log(x[x > 0]), na.rm=na.rm) / length(x))
}
geoMeans = apply(counts(diagdds), 1, gm_mean)
diagdds = estimateSizeFactors(diagdds, geoMeans = geoMeans)
diagdds = DESeq(diagdds, fitType="local")
```

```
## using pre-existing size factors

## estimating dispersions

## gene-wise dispersion estimates
## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing

## -- replacing outliers and refitting for 833 genes
## -- DESeq argument 'minReplicatesForReplace' = 7
## -- original counts are preserved in counts(dds)

## estimating dispersions

## fitting model and testing
```

```
res = results(diagdds, cooksCutoff = FALSE)
alpha = 0.01
sigtab = res[which(res$padj < alpha), ]
sigtab = cbind(as(sigtab, "data.frame"), as(tax_table(physeq)[rownames(sigtab), ], "matrix"))
head(sigtab)
```

```
##           baseMean log2FoldChange     lfcSE      stat       pvalue         padj
## 190162   12.055226      1.0276821 0.1760778  5.836522 5.330184e-09 1.707597e-07
## 134726    6.325778     -2.3081915 0.5322393 -4.336755 1.446018e-05 1.859769e-04
## 679245   13.244243     -2.2175788 0.4330141 -5.121262 3.034980e-07 6.218179e-06
## 4390365 446.543715      0.7107917 0.1897622  3.745698 1.798931e-04 1.460699e-03
## 3275562 148.487742      1.7768729 0.3074224  5.779906 7.474222e-09 2.310452e-07
## 4446902  14.718753     -0.9203260 0.1450865 -6.343293 2.249052e-10 1.132237e-08
##             Domain.       Phylum.             Class.                 Order.
## 190162  k__Bacteria p__Firmicutes      c__Clostridia      o__Clostridiales
## 134726  k__Bacteria p__Firmicutes         c__Bacilli    o__Lactobacillales
## 679245  k__Bacteria p__Firmicutes         c__Bacilli    o__Lactobacillales
## 4390365 k__Bacteria p__Firmicutes c__Erysipelotrichi o__Erysipelotrichales
## 3275562 k__Bacteria p__Firmicutes      c__Clostridia      o__Clostridiales
## 4446902 k__Bacteria p__Firmicutes         c__Bacilli         o__Gemellales
##                         Family.            Genus. Species
## 190162      f__Lachnospiraceae       g__Blautia     s__
## 134726     f__Lactobacillaceae g__Lactobacillus     s__
## 679245     f__Lactobacillaceae g__Lactobacillus     s__
## 4390365 f__Erysipelotrichaceae              g__     s__
## 3275562     f__Lachnospiraceae              g__     s__
## 4446902         f__Gemellaceae              g__     s__
```

```
## Exploring OTUs that were significant
# library("ggplot2")
theme_set(theme_bw())
scale_fill_discrete <- function(palname = "Set1", ...) {
    scale_fill_brewer(palette = palname, ...)
}
# Phylum order
x = tapply(sigtab$log2FoldChange, sigtab$Phylum, function(x) max(x))
x = sort(x, TRUE)
sigtab$Phylum = factor(as.character(sigtab$Phylum), levels=names(x))
# Genus order
x = tapply(sigtab$log2FoldChange, sigtab$Genus, function(x) max(x))
x = sort(x, TRUE)
sigtab$Genus = factor(as.character(sigtab$Genus), levels=names(x))
ggplot(sigtab, aes(x=Genus, y=log2FoldChange, color=Phylum)) + geom_point(size=6) +
  theme(axis.text.x = element_text(angle = -90, hjust = 0, vjust=0.5))
```


log2FoldChange
5
0
−5
−10
Genus
g__
g__Sutterella
g__Bacteroides
g__Slackia
g__Paraprevotella
g__Parabacteroides
g__Ruminococcus
g__Lachnospira
g__Actinobacillus
g__Varibaculum
g__[Ruminococcus]
g__Oscillospira
g__Faecalibacterium
g__Dorea
g__Roseburia
g__Clostridium
g__Blautia
g__Veillonella
g__Streptococcus
g__Granulicatella
g__Bifidobacterium
g__Actinomyces
g__Collinsella
g__Lactococcus
g__Enterococcus
g__Atopobium
g__Dialister
g__Bilophila
g__Lactobacillus
g__Prevotella
g__Porphyromonas
g__Haemophilus
g__Rothia
g__Leuconostoc
g__Sarcina
g__Eggerthella
g__Coprococcus
g__Acinetobacter
g__Megamonas
g__Megasphaera
g__Fusobacterium
g__Proteus
g__[Eubacterium]
g__Peptostreptococcus
Phylum
p__Firmicutes
p__Proteobacteria
p__Bacteroidetes
p__Actinobacteria
p__Cyanobacteria
p__Fusobacteria