Microbiome-Test-Project

Michael A. Meier

1/15/2022

```
# load required packages
library("tidyverse")
library("FactoMineR")
library("factoextra")
library("lme4")
```

Intro questions

1) Explain the advantages and disadvantages of 16S sequencing and Whole Genome Shotgun metagenomic sequencing.

16S amplicon sequencing targets the variable regions (V1-V9) of the bacterial 16S rRNA gene. As microbes in a community sample can be distinguished based on the amplicon sequence, this provides a rapid and cost-effective way to estimate the relative abundance of distinct microbes in a sample. One downside is that the 16S rRNA gene is only about 1.5 kb long, and most analyses target a subset or only one variable region. Precise identification of microbial taxa based on short reads is often not possible, because even distantly related microbial groups may have almost identical sequences, whereas in other cases closely related groups may show high variability in particular regions of the 16S rRNA gene. Furthermore, the PCR step involved in amplicon sequencing is a source of errors that can introduce point mutations through miscopying of DNA as well as chimaeric sequences, none of which are of biological origin.

Whole Genome Shotgun metagenomic sequencing sequences all DNA in a sample. While this approach is more time consuming and costly than 16S sequencing, the main advantage is that functional predictions about the microbiome can be made based on the presence/absence of protein coding genes. Given that whole genomic information is sequenced, it is possible to reach higher taxonomic resolution of microbial groups than by relying on 16S sequences. However, it can be challenging to reconstruct bacterial genomes from community samples, especially in samples with high diversity.

2) What is the difference between Operational Taxonomic Units (OTUs) and Amplicon Sequence Variants (ASVs)?

OTUs are artificial bins of 16S sequences, in which sequences are grouped by sequence similarity (commonly 97% or 99% identity within groups). This approach was originally proposed to limit the problems of possible DNA amplification and sequencing errors, and inconclusive taxonomic annotation based on short reads (see 1). While OTUs provide a standardized way to define microbial groups in a community, the same OTUs are often difficult to reproduce between experiments, or when adding additional sequences to an existing dataset, as the grouping of sequences may change depending on the initial set of sequences. Furthermore, OTUs may be difficult to interpret as they may or may not correspond to biologically distinct actors within a microbial community. As I have observed in my own research (https://journals.asm.org/doi/full/10.1128/AEM.03132-20), subgroups of microbes within a single OTU may show different and even opposite responses to experimental treatments. These

limitations together with improved methods of sequencing quality control suggested the direct use of ASVs, i.e. unaltered 16S sequences derived from amplicon sequencing. One advantage of ASVs is that they are unambiguous and the very same ASVs can be identified in subsequent experiments. However, the danger of using variants that arose due to sequencing errors still persists, and diligent data filtering is required to remove ASVs with low representation.

3) Briefly describe how would one go about doing functional analysis based on the 16S microbiome data and what would be the main limitations of such analysis?

Functional analysis of microbial communities aims to establish the mechanisms by which individual community members respond to treatments and influence the host phenotype.

The first challenge is to define these “community members”. As outlined above, OTUs are difficult to interpret, and resolution at the ASV level may be too high to be practical. In a recent experiment (https://www.biorxiv.org/content/10.1101/2021.11.01.466815v1) I devloped a method in which I use experimental data in conjunction with 16S sequence information to group ASVs at the genus and species level with better confidence.

Once meaningful community members are defined, the next step will be to investigate the function of each microbial group. A part of this test project is dedicated towards finding associations between microbe abundance and metabolites. However, even if associations are found, metabolic capabilities of the microbial community cannot be directly inferred from 16S microbiome data. I suggest using PICRUSt2 (https://huttenhower.sph.harvard.edu/picrust/), which takes advantage of microbial genome databases to reconstruct a metagenome from a list of microbes identified through 16S sequencing. This preliminary analysis may be sufficient to narrow down the list of microbial community members to a list of candidates that may have a hand in producing a given host phenotype (e.g. capability to produce selected metabolites).

General Microbiome Analysis

1) What is the difference between PCA and PCoA?

Principal Component Analysis is a form of ordination analysis with the aim of reducing the dimensionality of a dataset into principle components, i.e the most relevant summary statistics derived from the measurement of multiple variables. On a 2D PCA plot, the distance between individual samples is plotted (how distant or how different samples are from one another along the first two dimensions). There are different ways to calculate this distance. In PCA, the euclidean distance is used (essentially the pythagorean theorem for n dimensions). Apart from euclidean distance, there are other ways to calculate distances between samples that are popular in microbial community analysis such as Jaccard, Bray-Curtis or UniFrac, which also takes into account the phylogeny of ASVs. These methods are summarized as Principal Coordinates Analysis PCoA.

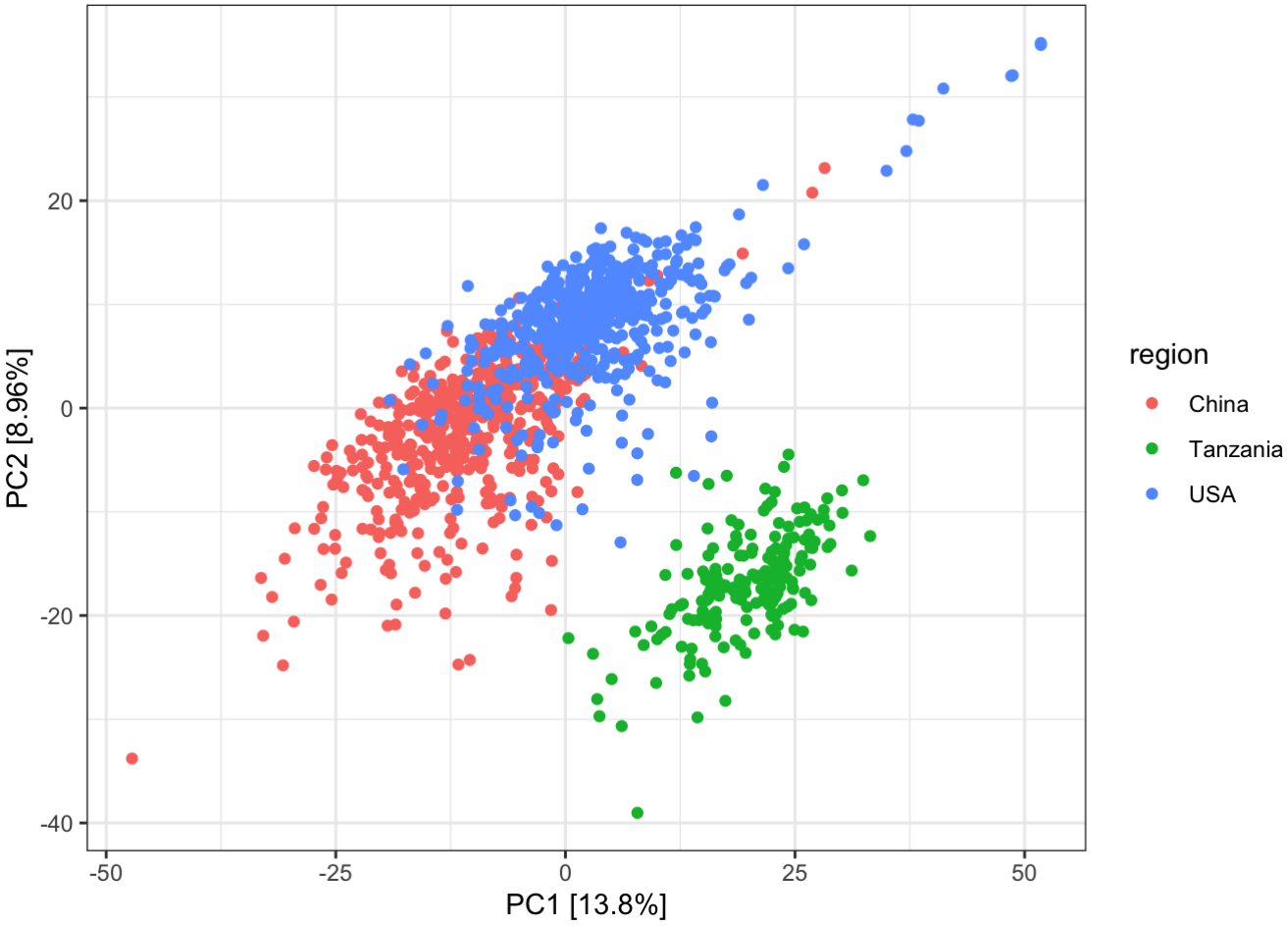
2a) Normalize the microbial counts and run PCA on the microbiome data.

```
test_microbial_abundance <- read_delim("test_microbial_abundance.txt", delim = "\t")# cr
eate n x p ASV table
# for normalization use log transformed relative abundance
asvtab <- test_microbial_abundance %>%
  pivot_wider(names_from = asv_id, values_from = asv_count) %>%
  replace(is.na(.), 0) %>% ## replace missing ASVs in each sample with zero
  mutate_if(is.numeric, function(x) x+1) %>% # add pseudocount
  pivot_longer(cols = starts_with("asv_"), names_to = "asv_id", values_to = "asv_count")
%>%
  group_by(sampleid) %>%
  mutate(total_obs = sum(asv_count)) %>%
  mutate(logrel = log(asv_count/total_obs)) %>% # calculate log relative abundance
  dplyr::select(-total_obs, -asv_count, -region) %>%
  pivot_wider(names_from = asv_id, values_from = logrel) %>%
  column_to_rownames(var = "sampleid")


# run PCA

res_pca <- PCA(asvtab, scale.unit = FALSE, ncp = 5, graph = FALSE)

res_pca
```

```
## **Results for the Principal Component Analysis (PCA)**
## The analysis was performed on 1186 individuals, described by 752 variables
## *The results are available in the following objects:
##
##    name               description
## 1  "$eig"             "eigenvalues"
## 2  "$var"             "results for the variables"
## 3  "$var$coord"       "coord. for the variables"
## 4  "$var$cor"         "correlations variables - dimensions"
## 5  "$var$cos2"        "cos2 for the variables"
## 6  "$var$contrib"     "contributions of the variables"
## 7  "$ind"             "results for the individuals"
## 8  "$ind$coord"       "coord. for the individuals"
## 9  "$ind$cos2"        "cos2 for the individuals"
## 10 "$ind$contrib"     "contributions of the individuals"
## 11 "$call"            "summary statistics"
## 12 "$call$centre"     "mean of the variables"
## 13 "$call$ecart.type" "standard error of the variables"
## 14 "$call$row.w"      "weights for the individuals"
## 15 "$call$col.w"      "weights for the variables"
```

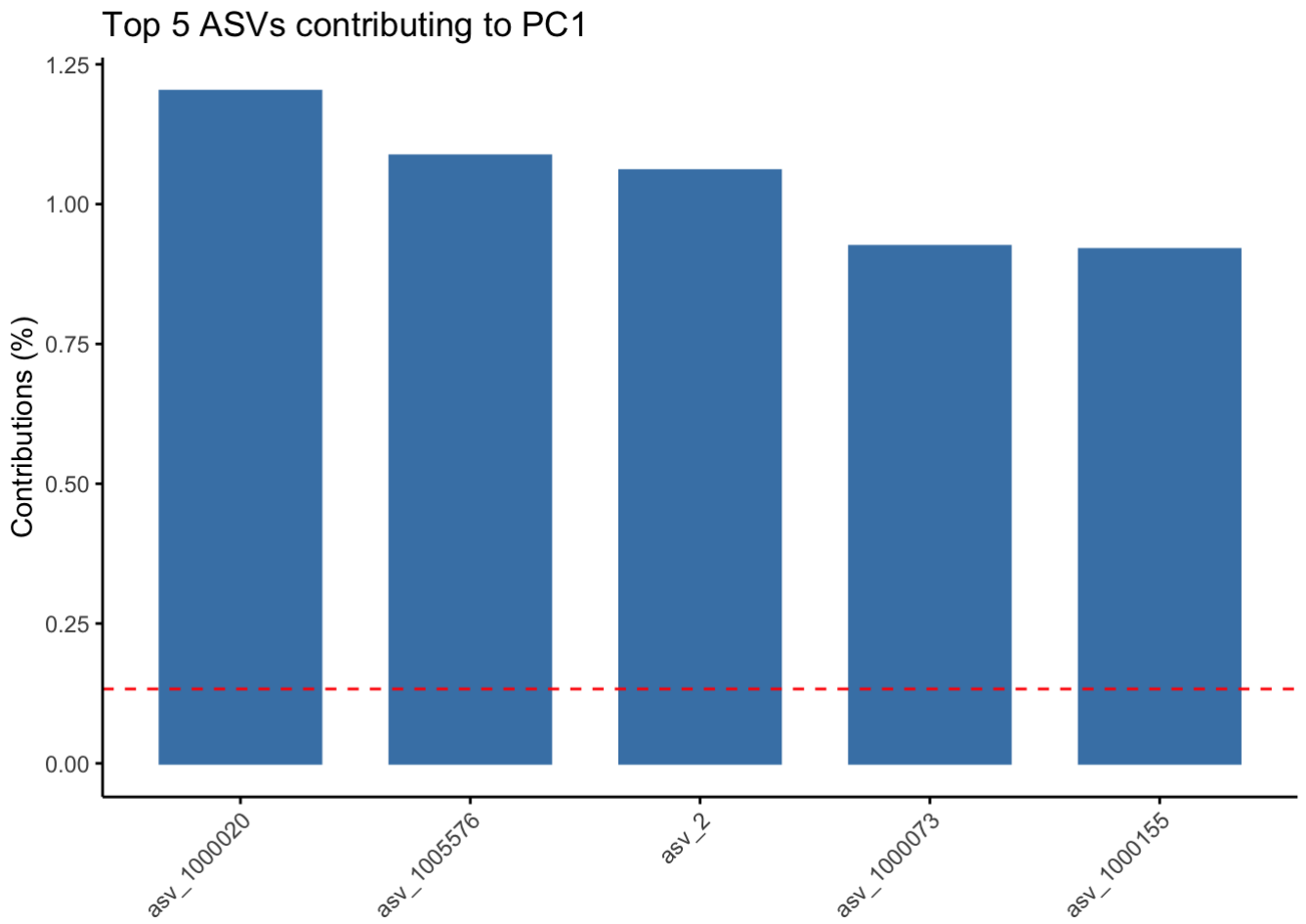
2b) Generate a 2D PCA plot with the first two principal components, color points by the geographical location (region column).

```
coords <- data.frame(res_pca$ind$coord)

plot_data <- coords %>%
  rownames_to_column(var = "sampleid") %>%
  left_join(unique(test_microbial_abundance[, c("sampleid", "region")]))xlab <- paste0(
"PC1 [", round(res_pca$eig[1, c("percentage of variance")], 2),"%]" )
ylab <- paste0("PC2 [", round(res_pca$eig[2, c("percentage of variance")], 2),"%]" )


pca_plot <- ggplot(plot_data, aes(x = Dim.1, y = Dim.2, color = region)) +
  geom_point() +
  xlab(xlab) +
  ylab(ylab) +
  theme_bw()

pca_plot
```

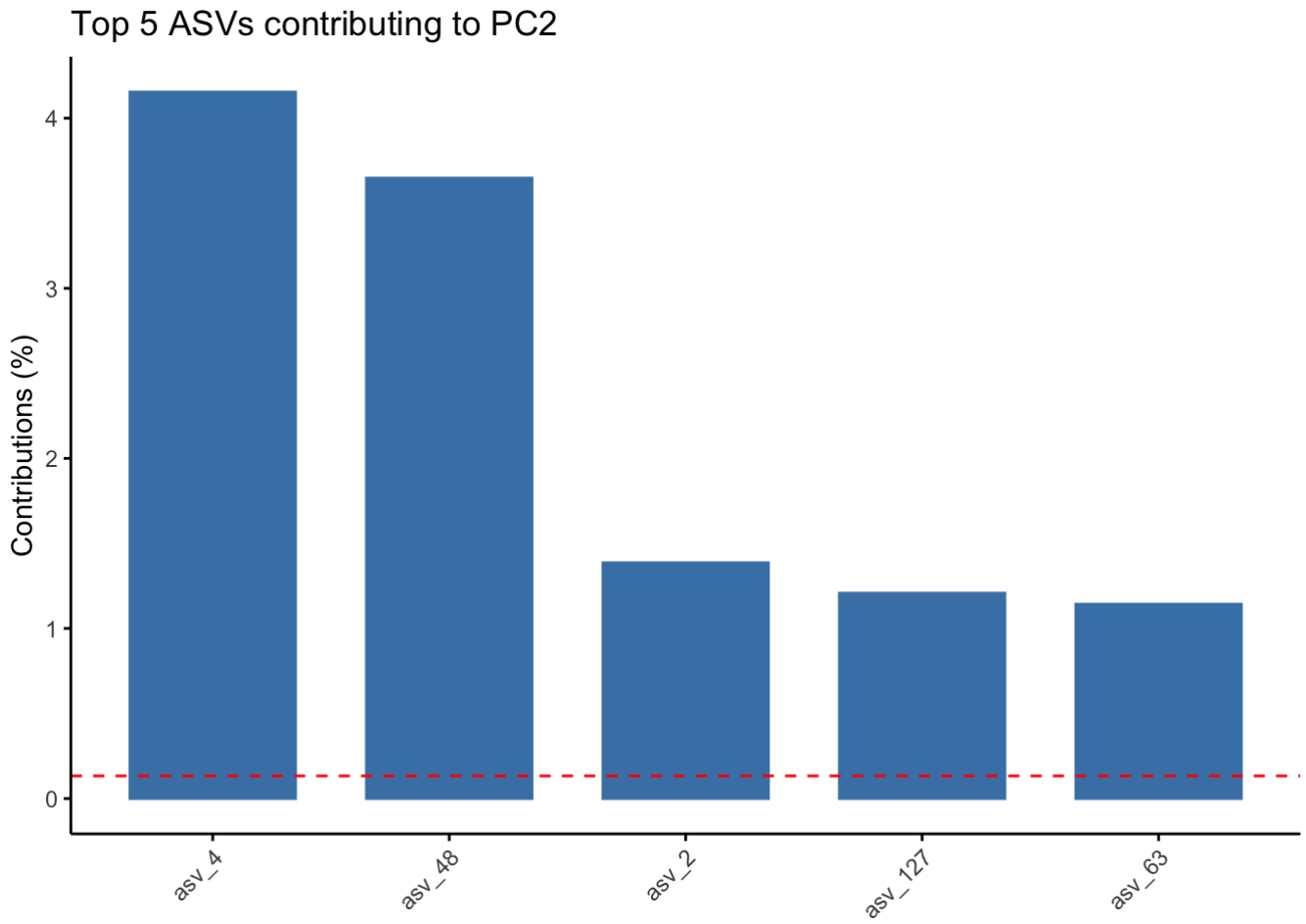


2c) What are the top 5 microbes (in terms of ASV IDs) that contribute the most to the first two principal components?

```
## get ASV contributions
asv_contributions <- res_pca$var$contrib

## plot top 5 ASV contributions using library factoextra

fviz_contrib(res_pca, choice = "var", axes = 1, top = 5, ggtheme = theme_classic()) +
  ggtitle("Top 5 ASVs contributing to PC1")
```




```
fviz_contrib(res_pca, choice = "var", axes = 2, top = 5, ggtheme = theme_classic()) +
  ggtitle("Top 5 ASVs contributing to PC2")
```



3) Find ASV IDs associated with BMI, adjusted for age and gender, using a method of your choice (briefly explain why you used that method) and show a list of the top 5 most significantly associated ASV IDs.

Since there are multiple variables in the dataset to control for (age, gender, region), I choose to try a modeling / variance partitioning approach, which will measure the effects of all factors in one fell swoop.

Another way would be to divide the samples into two gropus with high BMI (>25) and low BMI (<25) and look at differential abundance of each ASV between both groups.

For some samples, the data for one or more phenotyping variables is missing. Explain why/how you dealt with this missing data in the subsequent analyses.

There are generally two ways to deal with missing data. Either the samples with missing data are excluded from the analysis, or missing values can be imputed. In our case BMI data is missing from some samples. Since there is no obvious pattern to the missing data (e.g. BMI missing from all China samples), I could replace missing values with the mean BMI for all females or males, respectively, or with a mean BMI specific to region and gender if there is a difference. For simplicity, and since only 70/840 (8.3%) of samples have missing BMI data, I choose to simply exclude them.

```
# prepare data

asv_data <- asvtab %>%
  rownames_to_column( var = "sampleid") #%>%
  #filter(!(startswith(sampleid, "tanzania")))

test_phenotypes <- read_delim("test_phenotypes.txt", delim = "\t")phenotype_data <- test
_phenotypes %>%
  dplyr::select(sampleid, age, gender, bmi, region) %>%
  filter(!(is.na(bmi))) %>%
  group_by(gender, region) %>%
  mutate(rep = paste0("rep", row_number()))


test_data <- phenotype_data %>%
  left_join(asv_data, by = "sampleid")



## function to build lmer model using experimental factors
get_fit <- function(dat, y){
  formula <- as.formula(paste(y, "~ (1|bmi) + (1|age) + (1|gender) + (1|region) + (1|re
p)"))
  fit <- lmer(data = dat, formula)
  return(fit)
}


## define columns that hold asv data
cols <- colnames(test_data)[grepl("asv_", colnames(test_data))]

## data frame to hold results
df <- data_frame(grp =c("bmi", "age", "gender", "region", "rep", "residual"))## for each
asv find portion of variance explained by each factor
for (y in cols){
  #print(paste(y, as.character(which(cols == y)), "out of", as.character(length(cols))))
  fit <- get_fit(test_data, y)
  vc <- as.data.frame(lme4::VarCorr(fit))
  vc[, y] <- round(vc$vcov/sum(vc$vcov)*100, 8) # calculate portion of total variance in
%
  vc <- vc[, c("grp", y)]
  df <- left_join(df, vc, by = "grp")
}

pcvar <- as.data.frame(t(df[, -1]))
colnames(pcvar) <- df$grp
pcvar <- rownames_to_column(pcvar, var = "asv_id")


## top 5 asv ids most significantly associated with bmi (i.e. largest portion of varianc
e explained by bmi)
head(arrange(pcvar, -bmi), 5)
```

asv_id <chr>	bmi <dbl>	age <dbl>	gender <dbl>	region <dbl>	rep <dbl>	residual <dbl>
1 asv_1000794	40.51689	0.00000000	0e+00	19.74667	0.0000000	*NA*
2 asv_454	40.34421	0.00000000	0e+00	19.13853	0.8488053	*NA*
3 asv_579	39.13652	0.00000001	0e+00	18.48778	0.5242916	*NA*
4 asv_1000376	36.87711	0.16967621	0e+00	21.77007	0.0000000	*NA*
5 asv_1000600	33.04912	0.94240961	2e-08	27.64763	0.4603430	*NA*

5 rows

Microbial-metabolite associations

```
# load data
mm_microbial_abundances <- read_delim("mm_microbial_abundances.txt", delim = "\t")
mm_metabolite_values <- read_delim("mm_metabolite_values.txt", delim = "\t")
```

1) What are the potential problems of using linear regression to microbe-metabolite associations?

linear regression assumes there is a straight line (linear) relationship between microbe abundance and metabolite values. This may or may not be the case. linear regression is sensitive to outliers (because of squared distances to regression line), which could be a potential problem. Lastly, linear regression requires independent data, which means the metabolite value or microbe abundance of one sample must have nothing to do with the value of another. I don't know where the samples in mm_metabolite_values.txt and mm_microbial_abundances.txt come from, but if as in the first example they are taken from patients in different demographic groups (China female, China male, USA female, USA male), they would not be independent.

2) What could be some of the approaches alternative to linear regression to find metabolites potentially produced by microbes? (Do not need to run them, just describe.)

Other forms of regression such as nonlinear regression or Bayesian regression models. Machine learning approaches: e.g. MelonnPan (https://www.nature.com/articles/s41467-019-10927-1) uses a model trained on samples for which both sequencing data and experimentally measured metabolite abundances are available to predict metabolic profiles of a microbiome sample or MMVEC (https://www.nature.com/articles/s41592-019-0616-3), which uses neural networks to estimate the probability that a metabolite is present given the presence of a specific microbe. Make use of metabolic profiles derived from microbial cultures: A recent study (https://www.nature.com/articles/s41586-021-03707-9#auth-Justin_L_-Sonnenburg) characterized the metabolic profile of 178 gut microorganism strains using a library of 833 metabolites.

3) In a typical dataset, there may be many microbes and many metabolites so the number of all possible combinations might be too large to run in a reasonable time. Typically, not every combination needs to be tested either. Use some meaningful way of selecting a subset metabolites and microbes to run a reduced number of regressions.

```
# Do PCA on metabolites, select top metabolites affecting top PCs

metabolites <- mm_metabolite_values

metabolite_table <- metabolites %>%
  pivot_wider(names_from = mtb_id, values_from = mtb_value) %>%
  #replace(is.na(.), 0) %>%
  column_to_rownames(var = "sampleid")

# run PCA

res_pca <- PCA(metabolite_table, scale.unit = FALSE, ncp = 50, graph = FALSE)
#save(res_pca, file = "cache/res_pca_metabolites.rda")

head(res_pca$eig, 32)
```

```
##           eigenvalue percentage of variance cumulative percentage of variance
## comp 1  7.341377e+14             32.41269192                           32.41269
## comp 2  6.215950e+14             27.44385269                           59.85654
## comp 3  4.078063e+14             18.00493537                           77.86148
## comp 4  1.731984e+14              7.64683179                           85.50831
## comp 5  6.834468e+13              3.01746538                           88.52578
## comp 6  4.179457e+13              1.84525971                           90.37104
## comp 7  3.888378e+13              1.71674626                           92.08778
## comp 8  3.765561e+13              1.66252162                           93.75030
## comp 9  1.846028e+13              0.81503426                           94.56534
## comp 10 1.823587e+13              0.80512657                           95.37047
## comp 11 1.213633e+13              0.53582767                           95.90629
## comp 12 1.144178e+13              0.50516240                           96.41146
## comp 13 7.382012e+12              0.32592099                           96.73738
## comp 14 6.652163e+12              0.29369765                           97.03107
## comp 15 5.578422e+12              0.24629125                           97.27737
## comp 16 4.797386e+12              0.21180796                           97.48917
## comp 17 3.802842e+12              0.16789814                           97.65707
## comp 18 3.371303e+12              0.14884538                           97.80592
## comp 19 2.889222e+12              0.12756119                           97.93348
## comp 20 2.786013e+12              0.12300441                           98.05648
## comp 21 2.444636e+12              0.10793239                           98.16441
## comp 22 2.229758e+12              0.09844536                           98.26286
## comp 23 2.167074e+12              0.09567785                           98.35854
## comp 24 2.138305e+12              0.09440766                           98.45295
## comp 25 2.102936e+12              0.09284611                           98.54579
## comp 26 1.940032e+12              0.08565378                           98.63145
## comp 27 1.808403e+12              0.07984224                           98.71129
## comp 28 1.615371e+12              0.07131977                           98.78261
## comp 29 1.586407e+12              0.07004100                           98.85265
## comp 30 1.550077e+12              0.06843699                           98.92109
## comp 31 1.399456e+12              0.06178695                           98.98287
## comp 32 1.114373e+12              0.04920037                           99.03207
```

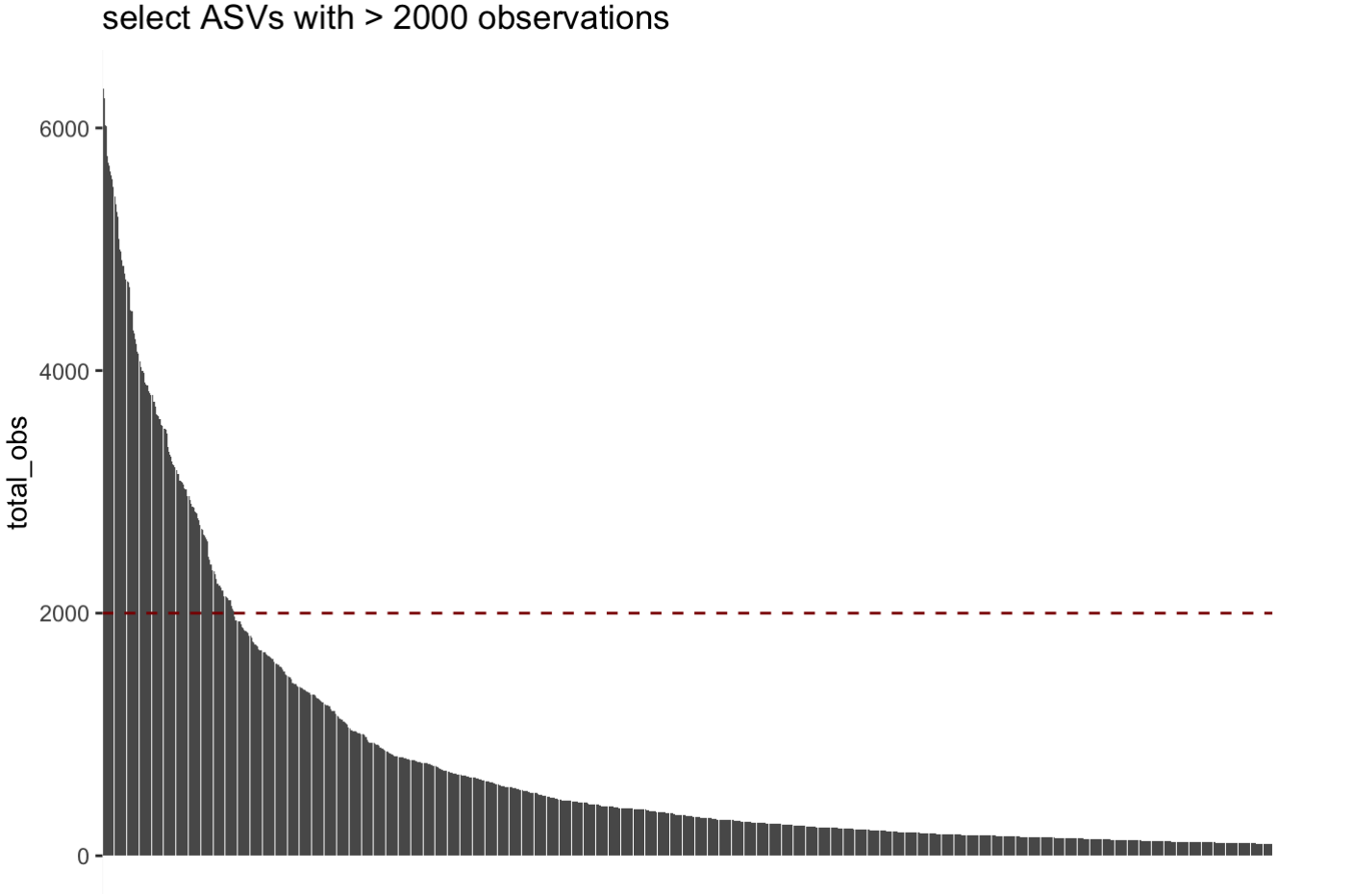
```
## the first 32 PCs explain > 99% of total variance

## get metabolite contributions
metabolite_contributions <- data.frame(res_pca$var$contrib)

#for first 32 PCs select top 100 metabolites contributing to these PCs
top_metabolites <- c()
npcs <- 32
n <- 10

for (i in 1:npcs){

  topn <- rownames(head(metabolite_contributions[order(-metabolite_contributions[,
 i]),], n))

  top_metabolites <- c(top_metabolites, topn)
}

top_metabolites <- unique(top_metabolites)

## selected 113 top metabolites for further analysis
top_metabolites
```

```
##   [1] "mtb_73045"  "mtb_75311"  "mtb_44962"  "mtb_84912"  "mtb_50826"
##   [6] "mtb_50597"  "mtb_70809"  "mtb_61764"  "mtb_50836"  "mtb_66324"
##  [11] "mtb_63719"  "mtb_61809"  "mtb_97702"  "mtb_49445"  "mtb_61794"
##  [16] "mtb_63723"  "mtb_19198"  "mtb_49562"  "mtb_103684" "mtb_85060"
##  [21] "mtb_113112" "mtb_40866"  "mtb_41536"  "mtb_97709"  "mtb_97537"
##  [26] "mtb_70219"  "mtb_95267"  "mtb_70452"  "mtb_70844"  "mtb_76722"
##  [31] "mtb_76850"  "mtb_57269"  "mtb_102982" "mtb_113036" "mtb_87677"
##  [36] "mtb_100437" "mtb_92973"  "mtb_108239" "mtb_87684"  "mtb_62561"
##  [41] "mtb_66462"  "mtb_76740"  "mtb_76731"  "mtb_76852"  "mtb_63730"
##  [46] "mtb_66920"  "mtb_61668"  "mtb_54897"  "mtb_25608"  "mtb_25621"
##  [51] "mtb_25393"  "mtb_75323"  "mtb_100380" "mtb_112831" "mtb_85069"
##  [56] "mtb_127765" "mtb_113120" "mtb_112726" "mtb_84801"  "mtb_55077"
##  [61] "mtb_105996" "mtb_119312" "mtb_60984"  "mtb_53153"  "mtb_65620"
##  [66] "mtb_43320"  "mtb_65736"  "mtb_61091"  "mtb_42980"  "mtb_56042"
##  [71] "mtb_53164"  "mtb_28050"  "mtb_28037"  "mtb_135287" "mtb_68259"
##  [76] "mtb_41820"  "mtb_41548"  "mtb_78893"  "mtb_48910"  "mtb_135114"
##  [81] "mtb_61675"  "mtb_117659" "mtb_52764"  "mtb_61883"  "mtb_73190"
##  [86] "mtb_73182"  "mtb_73198"  "mtb_78424"  "mtb_122933" "mtb_78419"
##  [91] "mtb_83693"  "mtb_55361"  "mtb_74326"  "mtb_78433"  "mtb_73754"
##  [96] "mtb_110261" "mtb_75554"  "mtb_97299"  "mtb_59446"  "mtb_48589"
## [101] "mtb_68594"  "mtb_59324"  "mtb_61724"  "mtb_51798"  "mtb_70535"
## [106] "mtb_81092"  "mtb_123711" "mtb_98447"  "mtb_60309"  "mtb_60438"
## [111] "mtb_57935"  "mtb_134213" "mtb_127058"
```

```
#save(top_metabolites, file = "cache/top_metabolites.rda")

# For ASVs I apply an abundance filter

microbes <- mm_microbial_abundances


total_obs <- microbes %>%
  group_by(asv_id) %>%
  tally(name = "total_obs")


# select ASVs with at least 2000 observations

top_asvs <- filter(total_obs, total_obs >= 2000)$asv_id

## selected 126 top ASVs for further analysis
top_asvs
```

```
##   [1] "asv_1"       "asv_1000020" "asv_1000025" "asv_1000031" "asv_1000036"
##   [6] "asv_1000056" "asv_1000061" "asv_1000068" "asv_1000074" "asv_1000085"
##  [11] "asv_1000091" "asv_1000103" "asv_1000105" "asv_1000106" "asv_1000107"
##  [16] "asv_1000119" "asv_1000127" "asv_1000132" "asv_1000133" "asv_1000138"
##  [21] "asv_1000142" "asv_1000143" "asv_1000146" "asv_1000149" "asv_1000157"
##  [26] "asv_1000174" "asv_1000184" "asv_1000190" "asv_1000193" "asv_1000194"
##  [31] "asv_1000203" "asv_1000205" "asv_1000245" "asv_1000317" "asv_1000324"
##  [36] "asv_1000327" "asv_1000349" "asv_1000375" "asv_10251"   "asv_11113"
##  [41] "asv_11718"   "asv_12071"   "asv_12110"   "asv_12252"   "asv_12565"
##  [46] "asv_12566"   "asv_12631"   "asv_12864"   "asv_12874"   "asv_12878"
##  [51] "asv_12880"   "asv_12881"   "asv_12916"   "asv_12920"   "asv_12925"
##  [56] "asv_12935"   "asv_12950"   "asv_12962"   "asv_12963"   "asv_12973"
##  [61] "asv_12977"   "asv_12990"   "asv_12995"   "asv_15196"   "asv_15987"
##  [66] "asv_16969"   "asv_215"     "asv_2243"    "asv_2311"    "asv_2365"
##  [71] "asv_2594"    "asv_3225"    "asv_3248"    "asv_3912"    "asv_4048"
##  [76] "asv_4049"    "asv_4051"    "asv_4053"    "asv_4059"    "asv_4071"
##  [81] "asv_4073"    "asv_4078"    "asv_4079"    "asv_4086"    "asv_4087"
##  [86] "asv_4097"    "asv_4112"    "asv_4133"    "asv_4140"    "asv_4185"
##  [91] "asv_4202"    "asv_4284"    "asv_4291"    "asv_4327"    "asv_4361"
##  [96] "asv_4464"    "asv_4480"    "asv_4590"    "asv_4693"    "asv_4839"
## [101] "asv_4931"    "asv_5336"    "asv_5486"    "asv_5497"    "asv_5844"
## [106] "asv_6214"    "asv_6651"    "asv_7291"    "asv_7294"    "asv_7347"
## [111] "asv_7697"    "asv_8140"    "asv_8381"    "asv_8493"    "asv_9245"
## [116] "asv_9470"    "asv_9471"    "asv_9472"    "asv_9481"    "asv_9483"
## [121] "asv_9486"    "asv_9487"    "asv_9488"    "asv_9489"    "asv_9493"
## [126] "asv_9887"
```

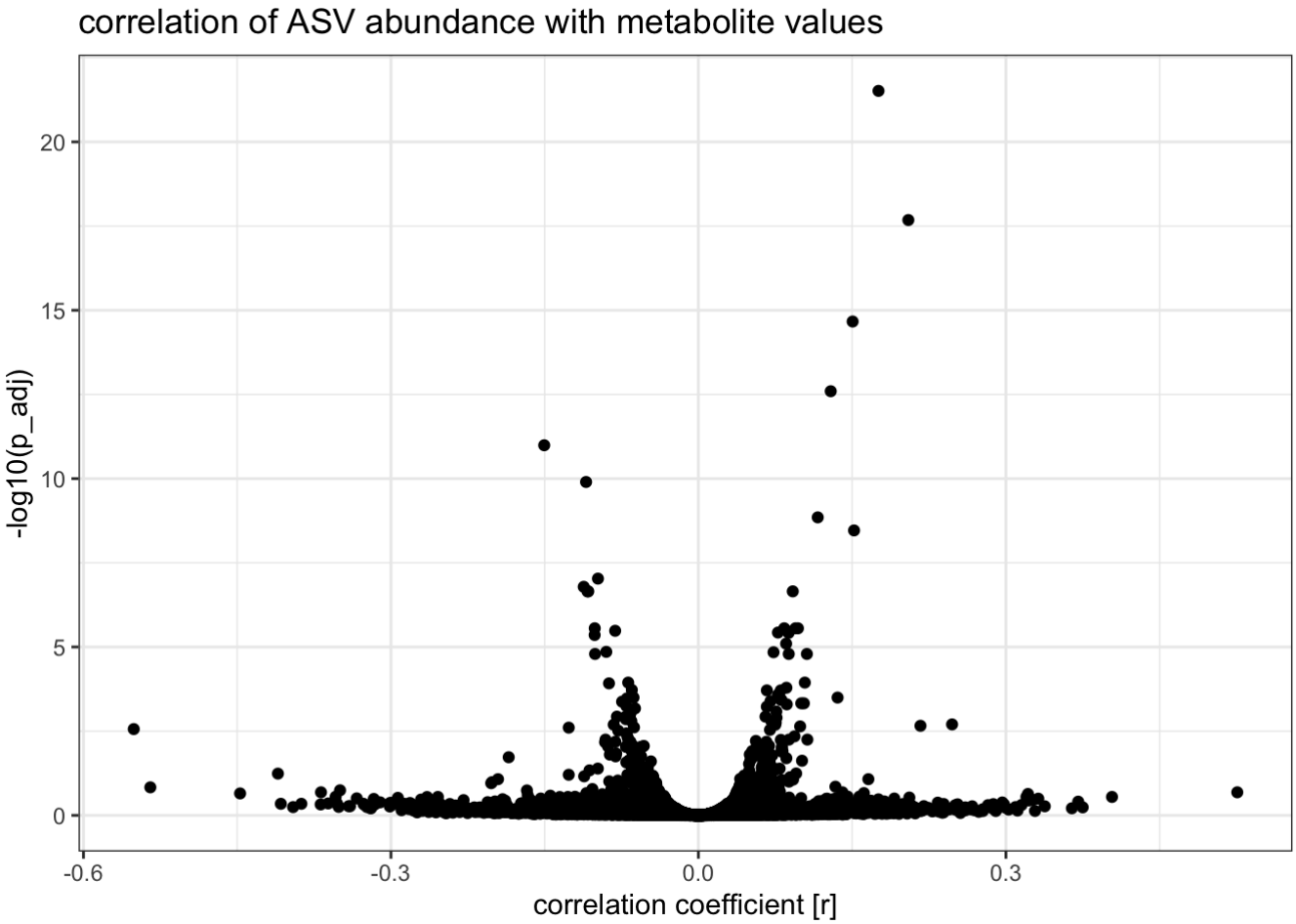
```
#save(top_asvs, file = "cache/top_asvs.rda")

total_obs_plot <- ggplot(total_obs, aes(x = reorder(asv_id, -total_obs), y = total_obs))
+
  geom_bar(stat = "identity") +
  ggtitle("select ASVs with > 2000 observations") +
  xlab("ASVs ranked by total observations") +
  geom_hline( yintercept = 2000, linetype="dashed", color = "darkred") +
  theme(axis.title.x=element_blank(),
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank())

total_obs_plot
```



4) Write a code to run these regressions on your laptop or desktop computer and report the top 10 microbial-metabolite hits (in terms of ASV IDs and MTB IDs), including the p-values and confidence intervals for the effect size.

```
## linear regression for 126 ASVs and 113 metabolites

# prepare data

top_mm_metabolite_values <- filter(mm_metabolite_values, mtb_id %in% top_metabolites)

#save(top_mm_metabolite_values, file = "cache/top_mm_metabolite_values.rda")

top_mm_microbial_abundances <- filter(mm_microbial_abundances, asv_id %in% top_asvs)
## convert ASV counts to log transformed relative abundance
top_mm_microbial_abundances <- top_mm_microbial_abundances %>%
  pivot_wider(names_from = asv_id, values_from = asv_count) %>%
  replace(is.na(.), 0) %>% ## replace missing ASVs in each sample with zero
  mutate_if(is.numeric, function(x) x+1) %>% # add pseudocount
  pivot_longer(cols = starts_with("asv_"), names_to = "asv_id", values_to = "asv_count")
%>%
  group_by(sampleid) %>%
  mutate(total_obs = sum(asv_count)) %>%
  mutate(logrel = log(asv_count/total_obs)) %>% # calculate log relative abundance
  dplyr::select(asv_id, sampleid, logrel)

#save(top_mm_microbial_abundances, file = "cache/top_mm_microbial_abundances.rda")


## regression

## function to get data for asv/metabolite pair
get_data <- function(asv, metabolite){

  asvdat <- top_mm_microbial_abundances %>%
    filter(asv_id == asv) %>%
    dplyr::select(sampleid, logrel)

  mtbdat <- top_mm_metabolite_values %>%
    filter(mtb_id == metabolite) %>%
    dplyr::select(sampleid, mtb_value)

  dat <- left_join(asvdat, mtbdat, by = "sampleid") %>%
    filter(!is.na(mtb_value)) %>%
    #mutate(mtb_value = log(mtb_value)) %>% # log transform metabolite value
    filter(logrel >= -8) # set minimum ASV abundance

  return(dat)
}


## function to run linear regression
run_lin_reg <- function(asv, metabolite){
  dat <- get_data(asv, metabolite)
  fit <- lm(mtb_value ~ logrel, data = dat)

  comparison <- paste0(asv,"X", metabolite)
```

```
  p <- summary(fit)$coefficients[2,4]
  r <- cor(dat$logrel, dat$mtb_value)
  confint <- confint(fit, 'logrel', level=0.95)

  values <- c(comparison, asv, metabolite, r, p, confint[1], confint[2])
  return(values)

}


## collect results in data frame
reg_results <- data.frame(matrix(ncol = 7, nrow = 0))

#counter <- 0

## run all 126x113 (14238) comparisons

for (asv in top_asvs){
  for (mtb in top_metabolites){
    res <- run_lin_reg(asv, mtb)
    reg_results <- rbind(reg_results, res)
    #counter <- counter + 1
    #print(counter)
  }
}

colnames(reg_results) <- c("comparison", "asv_id", "mtb_id", "r", "p", "conf2.5", "conf9
7.5")

reg_results$r <- as.numeric(reg_results$r)
reg_results$p <- as.numeric(reg_results$p)
reg_results$conf2.5 <- as.numeric(reg_results$conf2.5)
reg_results$conf97.5 <- as.numeric(reg_results$conf97.5)

#save(reg_results, file = "cache/reg_results.rda")


# adjust p-values for multiple comparisons, using Benjamini & Hochberg method
reg_results$p_adj <- p.adjust(reg_results$p, method = "BH", n = length(reg_results$p))


# plot correlations and p values
# I like to plot the correlation r as well to see if a relationship is positive or negat
ive
correlation_plot <- ggplot(reg_results, aes(x = r, y= -log10(p_adj))) +
  geom_point() +
  theme_bw() +
  ggtitle("correlation of ASV abundance with metabolite values") +
  xlab("correlation coefficient [r]")

correlation_plot
```




```
# report top 10 microbial-metabolite hits

top10 <- reg_results %>%
  arrange(p_adj) %>%
  dplyr::select("comparison", "p_adj", "conf2.5", "conf97.5", "r") %>%
  head(10)

top10
```

	comparison <chr>	p_adj <dbl>	conf2.5 <dbl>	conf97.5 <dbl>	r <dbl>
1	asv_1000138Xmtb_134213	3.053398e-22	81056.03	117361.02	0.17571309
2	asv_15196Xmtb_134213	2.091593e-18	94054.13	141060.67	0.20482700
3	asv_1000138Xmtb_119312	2.159305e-15	117395.72	183027.17	0.15048949
4	asv_1000056Xmtb_134213	2.535632e-13	47638.06	76732.87	0.12911316
5	asv_5497Xmtb_134213	1.020768e-11	-81237.72	-48977.21	-0.15047196
6	asv_4284Xmtb_134213	1.258136e-10	-65267.60	-38352.14	-0.10962601
7	asv_1000031Xmtb_134213	1.416008e-09	33392.49	58358.56	0.11644219
8	asv_15196Xmtb_119312	3.459111e-09	111850.18	197541.54	0.15185326

	comparison <chr>	p_adj <dbl>	conf2.5 <dbl>	conf97.5 <dbl>	r <dbl>
9	asv_4284Xmtb_119312	9.333551e-08	-100748.36	-54398.97	-0.09804482
10	asv_9887Xmtb_119312	1.634597e-07	-117423.36	-62786.07	-0.11190109

1-10 of 10 rows

```
# mtb_119312 and mtb_134213 seem to be of interest!
```

5) Briefly describe few ways to how to adjust p-values when we test large number of hypotheses and the advantages or disadvantages of adjusting p-values in such manner.

For large numbers of comparisons/hypotheses, some associations are expected to be significant purely by chance. p-values are adjusted to avoid such false positives. The simplest method to adjust p-values for large numbers of comparisons/hypotheses is Bonferroni correction in which the p-values are multiplied by the number of comparisons. However, especially for very large numbers of comparisons, this may push most (if not all) p_values over the significance level, leading to false negative discoveries. Less stringent methods have been proposed. A popular approach is Benjamini & Hochberg, which takes into account the ranking of all p-values and reduces false positives, but also minimises false negatives. Additional methods include Holm, Hochberg, Hommel, Benjamini & Yekutieli, and others. The point is to be skeptical about discoveries made through multiple comparisons, and to verify associations with independent experiments.