6210_Assignment5: Microbiome Analysis with DADA2 Pipeline and Phylogenetic Profile analysis

Sichong Xu

17/12/2021

https://github.com/SichongX/6210_A5.git

Introduction

For this assignment, I am choosing theme 11, which will use DADA2 to investigate the microbial community. Microbiome analysis has always been an on-going popular topic, and amplicon sequencing has been used to describe microbial communities for nearly a decade. According to Callahan et al.'s study and the DADA2 (Divisive Amplicon Denoising Algorithm) tutorial, the operational taxonomic units (OUTs) should be replaced by the exact amplicon sequence variants (ASVs) as the standard unit of microbiome marker-gene analysis. DADA2 could identify low-read sequences and sequence variants that differed by even 1 or 2 nucleotides; hence, obtained ASVs will reveal a more accurate measurement of the microbiome community with true biological meanings. (Callahan et al., 2016 & Callahan et al., 2017) It is essential to assess the ability of DADA2 to identify real variants and yield fewer spurious sequences from amplicon data, as the authors claimed. And to check if the ASV results inferred from DADA2 are appropriate for any subsequent analysis without obstacles.

To demonstrate the reliability and reproducibility of DADA2 on microbiome analysis, I will use the DADA2 pipeline to perform amplicon sequencing on marker-gene (16s rRNA) sequences in bacteria with cautious quality control. And evaluate the accuracy of output ASVs with a mock community. ASVs inferred by DADA2 will then be passed to the phyloseq package for phylogenetic analysis to investigate the possible phylogenetic biodiversity on samples. The ultimate goal is to provide an assessment and a guidance on using the DADA2 package for amplicon sequencing and one potential downstream analysis on ASVs.

Description of Data Set

Because the DADA2 pipeline requires very strict data, sample data must be demultiplexed, any non-biological nucleotides must be removed before input, and samples must be paired-end sequences with forward and reverse reads matched in order. I will use the data set (MiSeq_SOP) obtained from the DADA2 pipeline tutorial, downloaded from https://www.mothur.org/w/images/d/d6/MiSeqSOPData.zip. Data were generated by Illumina Miseq amplicon sequencing of the V4 region of the 16S rRNA gene, and each sequence is 250 bp long. The 20 paired-end fastq files contain gut bacteria fecal samples collected from a post-weaning mouse (first 150days, 9 time points, 4 early, 5 late) and a mock community containing 20 known bacterial strains.

Code Section 1

###load the required R packages:

```r
library(dada2)
library(phyloseq)
library(ShortRead)
library(tidyverse)
library(vegan)
library(Biostrings)
library(ggplot2)
library(ape)
```

Data Acquisition

Firstly set the path to the right directory of downloaded data.

```r
path <- "~/Desktop/6210 Software Tools/Assignment 5/MiSeq_SOP"
#Set path to fastq data file
```

```
## [1] "/Users/sx/Desktop/6210 Software Tools/Assignment 5/MiSeq_SOP/F3D0_S188_L001_R1_001.fastq"
## [2] "/Users/sx/Desktop/6210 Software Tools/Assignment 5/MiSeq_SOP/F3D1_S189_L001_R1_001.fastq"
## [3] "/Users/sx/Desktop/6210 Software Tools/Assignment 5/MiSeq_SOP/F3D141_S207_L001_R1_001.fastq"
## [4] "/Users/sx/Desktop/6210 Software Tools/Assignment 5/MiSeq_SOP/F3D142_S208_L001_R1_001.fastq"
## [5] "/Users/sx/Desktop/6210 Software Tools/Assignment 5/MiSeq_SOP/F3D143_S209_L001_R1_001.fastq"
```

These fastq files are paired-end sequencing data in forward and reverse read. All files have the same format: SAMPLENAME_R1_001.fastq and SAMPLENAME_R2_001.fastq. File name with R1 correspond to forward reads, file name with R2 correspond to the reverse reads.

Data Exploration

Extra the 20 sample names form the file names with string manipulation.

```
##  [1] "F3D0"   "F3D1"   "F3D141" "F3D142" "F3D143" "F3D144" "F3D145" "F3D146"
##  [9] "F3D147" "F3D148" "F3D149" "F3D150" "F3D2"   "F3D3"   "F3D5"   "F3D6"
## [17] "F3D7"   "F3D8"   "F3D9"   "Mock"
```

F3 is the experiment subject, D represents the day of post-weaning.

Take a look at how fastq file looks like.

```
## [1] "@M00967:43:000000000-A3JHG:1:1101:18327:1699 1:N:0:188"
## [2] "NACGGAGGATGCGAGCGTTATCCGGATTTATTGGGTTTAAAGGGTGCGTAGGCGGCCTGCCAAGTCAGCGGTAAAATTGCGGGGCTCAACCCCGT
## [3] "+"
## [4] "#>>AABABBFFFGGGGGGGGGGGGGGGGHHHHHHHGGGHHHHHGHGGGGGGHGGGGGGHHHHHHHHHHGGGGGHHHHGHGGGGGGHHBGHGDGG
## [5] "@M00967:43:000000000-A3JHG:1:1101:14069:1827 1:N:0:188"
## [6] "TACGGAGGATGCGAGCGTTATCCGGATTTATTGGGTTTAAAGGGTGCGTAGGCGGCCTGCCAAGTCAGCGGTAAAATTGCGGGGCTCAACCCCGT
## [7] "+"
## [8] "3AA?ABBDBFFBEGGEGGGGAFFGGGGGHHHCGGGGGGHFGHGGCFDEFGGGHGGGEGF1GGFGHHHHHGGEGGHHHHHFGGGGGGHHHHHGGGG
```

By viewing the first 8 lines of fastq file, we can find that each entry consist 4 lines of information. The first line is the sequence identifier, second line contains the sequence, the third line is a "+" separator, the last line represent the base-calling quality score of sequence encoded by Phred +33 with ASCII. (illumina, 2021)

When browsing different package online, I found package "ShortRead" can also be used to read fastq file, it's another way to view fastq file in detail.

```
rfq1 <- readFastq(fnF[1])
rrq1 <- readFastq(fnR[1])
rfq1
```

```
## class: ShortReadQ
## length: 7793 reads; width: 249..251 cycles
```

```
rrq1
```

```
## class: ShortReadQ
## length: 7793 reads; width: 247..251 cycles
```

For the first sample, there are 7793 reads in both the forward read and reverse read fastq files.

To reduce code redundancy, I create a function “fastqdetail” to view the fastq file details.First output is the first 6 base sequence in fastq file.Second output is the encoded Phred quality score of first 6 reads.Third output is the corresponding quality score in numerical value.

```
fastqdetail <- function(a){
  cat("Sequence:\n")
  sequence <-  sread(a)[1:6]
  print(sequence)
  cat("\nPhred Quality Score:\n")
  quality <-  quality(a)
  print(quality[1:6])
  cat("\nNumerical Quality Score:\n")
  quality_socre <- as(quality,"matrix")[1:2,1:3]
  print(quality_socre)
}
```

Detail of the first forward fastq file, first 6 sequences:

```
## Sequence:
## DNAStringSet object of length 6:
##     width seq
## [1]   251 NACGGAGGATGCGAGCGTTATCCGGATTTATTGG...ACGCTGAGGCACGAAAGTGCGGGGATCAAACAG
## [2]   251 TACGGAGGATGCGAGCGTTATCCGGATTTATTGG...ACGCTGAGGCACGAAAGTGCGGGGATCAAACAG
## [3]   251 TACGGAGGATGCGAGCGTTGTCCGGAATCACTGG...GACGCTGAGGCGCGAAAGCGTGGGGAGCAAACA
## [4]   251 TACGGAGGATGCGAGCGTTATCCGGATTTATTGG...ACGCTCATGCACGAAAGTGTGGGTATCGAACAG
## [5]   251 TACGTAGGTGGCAAGCGTTATCCGGATTTACTGG...GACGCTGAGGCGCGAAAGCGTGGGGAGCAAACG
## [6]   251 TACGGAGGATGCGAGCGTTATCCGGATTTATTGG...ACGCTCATGCACGAAAGTGTGGGTATCGAACAG
##
## Phred Quality Score:
## class: FastqQuality
## quality:
## BStringSet object of length 6:
##     width seq
## [1]   251 #>>AABABBFFFGGGGGGGGGGGGGGGGHHHHHH...FFBDDFFFFFEFADFFFFFBAFFFA?EFFFBFF
## [2]   251 3AA?ABBDBFFBEGGEGGGGAFFGGGGGHHHCGG...FFFFFDDFAFFFFF.AF9/FBBBBB.EAFFE?F
## [3]   251 BA@BBBABBFFFGGGGGGGGGHGGGGGHHHHGH...FFFFFBDA9EEFFF>DFFFFDDFFFADDFFFF.
## [4]   251 BCBCCCCCCFFFGGGGGGGGGHGGGGGHHHHHH...FFAFFFFFFFFFFFFFFBFFFFFFFFFFEFFFF
## [5]   251 BBCCBFFCDFCFGGGGGGGGGGHGGGGGHHHHHH...FFFDFFFBFF.:;F-@FFBDAFAAC.;9?FFE.
## [6]   251 CCCCCCCCCFFFGGGGGGGGGGHGGGGGHHHHHH...FFFFFFDFFFFFFADFFFFFFFFAFFFFFFFFF
##
## Numerical Quality Score:
##      [,1] [,2] [,3]
## [1,]    2   29   29
## [2,]   18   32   32
```

Detail of the first reverse fastq file:

```
## Sequence:
## DNAStringSet object of length 6:
##     width seq
## [1]   251 CCTGTTTGATCCCCGCACTTTCGTGCCTCAGCGT...CCAATAAATCCGGATAACGCCCGCCTCCTCCGT
## [2]   251 CCTGTTTGATCCCCGCACTTTCGTGCCTCAGCGT...CCAATAAATCCGGGTAAAGCTCGCATCCTCCGG
## [3]   251 CCTGTTTGCTCCCCACGCTTTCGCGCCTCAGCGT...CCCAGTGATTCCGGACAAAGCCCGCCTCCCCCG
## [4]   251 CCTGTTCGATACCCACACTTTCGTGCATGAGCGT...CCAATAAATCCGGATAACGCTCGCATCCTCCGT
## [5]   251 CCTGTTTGCTCCCCACGCTTTCGCGCCTCAGCGT...CCCAGTAAATCCGGATAACGCTTGCCACCTACG
## [6]   250 CCTGTTCGATACCCACACTTTCGTGCATGAGCGT...CCCAATAAATCCGGATAACGCTCGCATCCTCCG
##
## Phred Quality Score:
## class: FastqQuality
## quality:
## BStringSet object of length 6:
##     width seq
## [1]   251 BABBBFFFFFFFFGEGGGGGGGGGHGGHHHHHGGG.../AFAB/B/B/FFF.AAFBA--@=F-.AA.;B..
## [2]   251 BABBBFFFFFFFFGGGGGGGGGGGHGGHHHHHHGG...FF?EFFFFB/FDC-99B9FB/;AF-ABFFFF.-
## [3]   251 >AAABFFFFFFFFGGGGGGGGGGGGGGGGGGHGGG...:9E..;//BB/:/9.9-.::B.9---9:..9D-
## [4]   251 ABBBBFFBBFBBGGGGGGGGGGGGHGGHGGHHHGG.../;EE./;::/9.@.A.B:.9.9;-999B99/.;
## [5]   251 >AABAFFFFFFFFGGGGGGGGGGGGGGGGGHHHGG...FB?EFFFF/BFFD;?FFFBD=AEEFB/FFFFF?
## [6]   250 BCCCCFFCCFCCGGGGGGGGGGGGHGGHGHHHHGG...B9AEEF0FF009.=...0B.-.9--;.@B0:B;
##
## Numerical Quality Score:
##      [,1] [,2] [,3]
## [1,]   33   32   33
## [2,]   33   32   33
```

A Phred quality score of 30 represents a base call error probability of 0.001, accuracy of 99.9%. 30 or higher is typically regarded as good quality. For the downstream analysis, we will need to trim the sequence length based on the quality score of 30 to ensure the algorithm's sensitivity to rare sequence variants. So it is important to visualize the quality profile to determine the trimming length.

```r
plotQualityProfile(fnF[1:4])
```

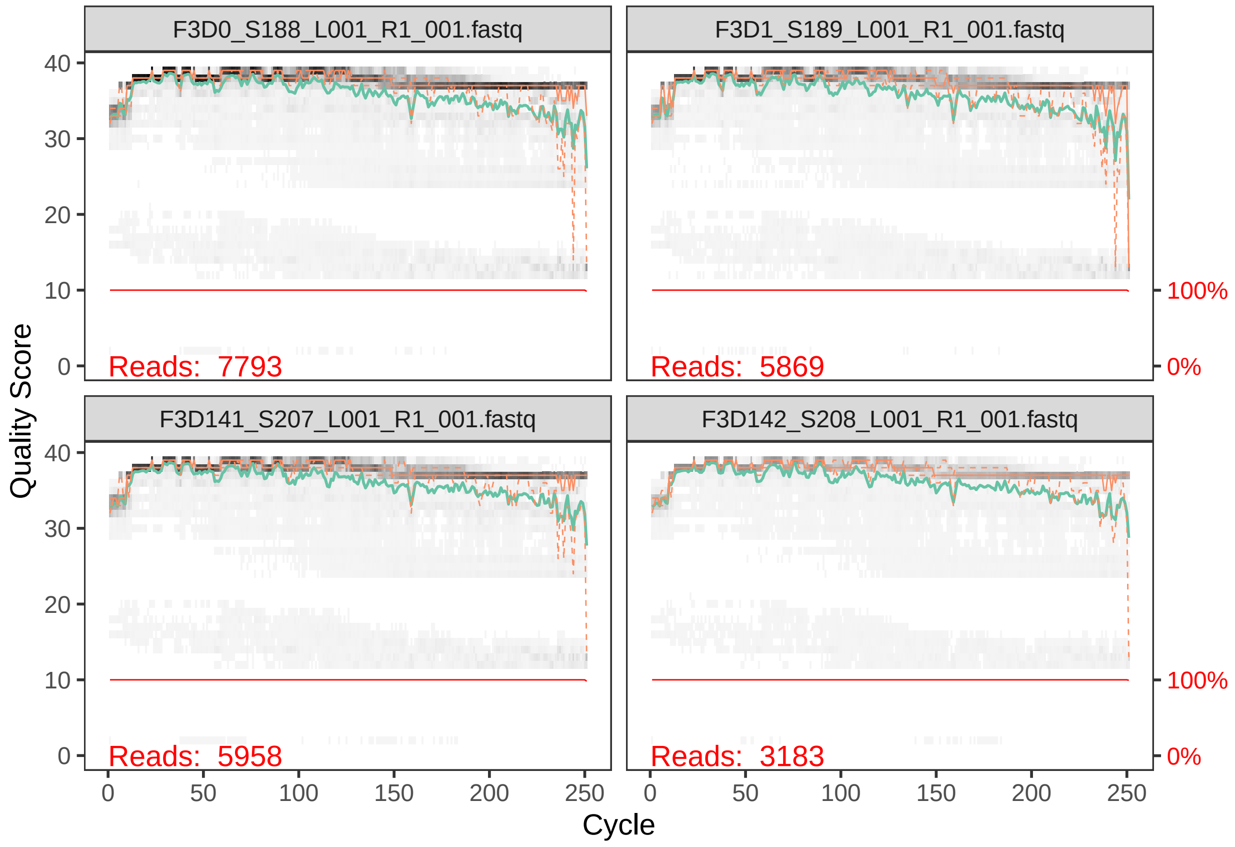

Figure 1. Plots of quality score of sequences in the forward read for first 4 samples. The mean quality score at each base position is in green, orange is the median, and the dashed orange lines are the 25th and 75th quantiles.

For the forward reads, overall quality is good. I will firstly cut off the 10 nucleotides at the beginning, as the quality score are not ideal. Then cut off 10 nts at the end of sequence to avoid less well-controlled errors as Benjamin suggested in the tutorial, length will be set as 240 bp. Therefore, sequence length of 230 bp for forward reads will be obtained after trimming.

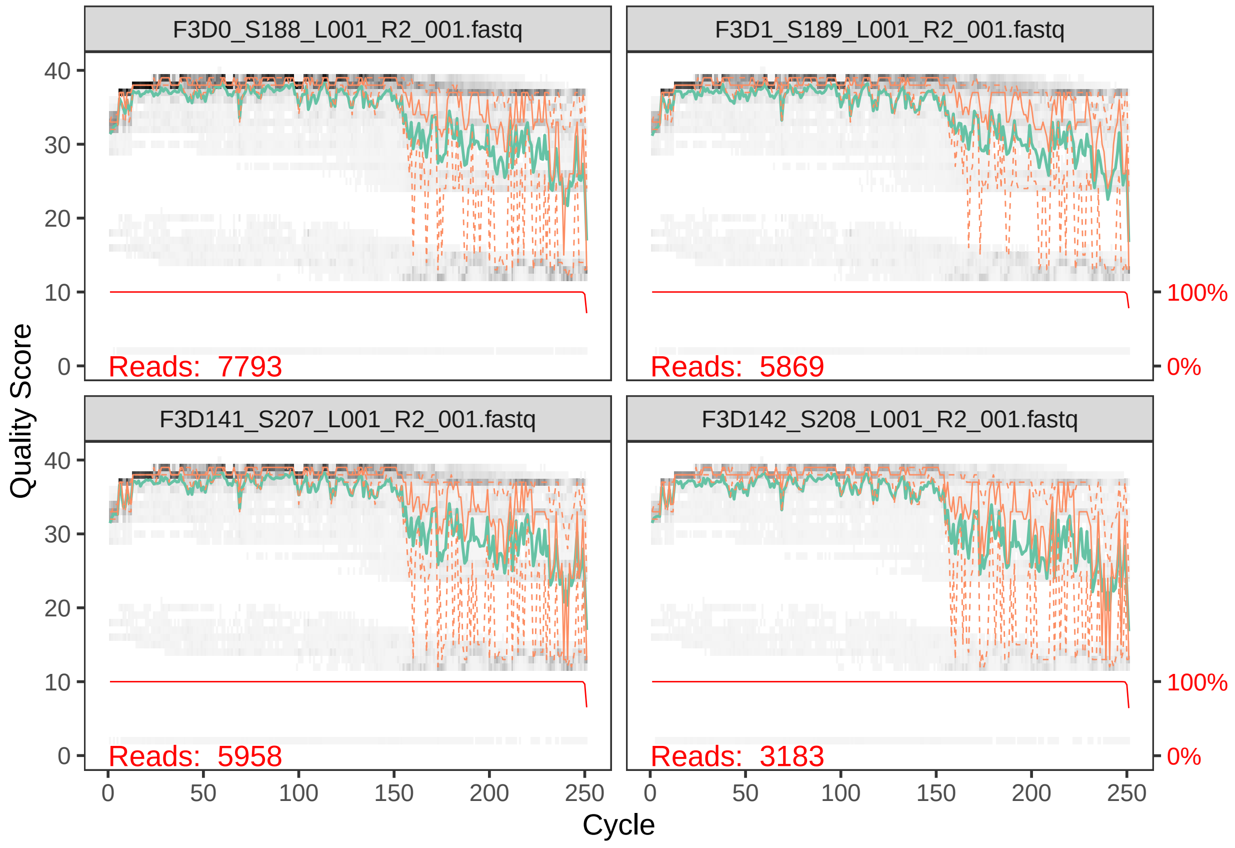

Figure 2. Plots of quality score of sequences in the reverse read for first 4 sample

Comparatively, forward read sequences have a better performance on quality score drop-off than reserve read sequences, especially towards end of sequence. The quality for all 4 reserve reads files start to drop below 30 around 160 bp position, so I will truncate reverse sequence at 160 bp for the following procedure.

Main Software Tools

DADA2

DADA2 adopts a new quality-aware model of Illumina amplicon errors, which presents a relative more robust performance than other software on marker gene analysis for microbiome without constructing operational taxonomic units. DADA2 is most sensitive on detecting amplicon sequence variants, and output few false positives. It has better sensitivity on rare species and rare variants (1 or 2 nt difference), so it can retain low-read sequence, and detect true variants that are present in the sample at a low frequency.

phyloseq

The phyloseq is a tool includes all knids of function of importing, storing, analysing, and visually displaying complex sequencing data, especially OTUs/ASVs with taxonomic assignment. This package combines many advantages from existing tools for ecology and phylogenetic analysis (vegan, ade4, ape, picante) to produce high-quality measurements. The package is also equipped with advanced/flexible graphic systems (ggplot2) to produce publication-quality graphics.

Code Section 2 - Main Analysis

Quality Control

Sequences trimming and filtering

Here I am creating a sub-directory (filtered) in the current path to save filtered fastq files in the compressed format for both forward and reverse reads (compressed file formats are supported).

```
filtF <- file.path(path, "filtered", paste0(sample.names, "_F_filt.fastq.gz"))
filtR <- file.path(path, "filtered", paste0(sample.names, "_R_filt.fastq.gz"))
names(filtF) <- sample.names
names(filtR) <- sample.names
```

As the preliminary quality control, I use function filterAndTrim to filter and trim forward and reverse sequences. By this step, forward reads are firstly trimmed at 240 bases, reverse reads are trimmed at 160 bases. For all the reads, sequences with Ns (maxN=0) will be discarded; (maxEE=c(2,2)) expected errors are set as 2, so sequences have more than two errors are filtered out; sequences with quality score lower than 2 will be filtered (truncQ=2); sequences will be trimmed 10 nts from start (trimLeft = 10); the multithread is set as TRUE to reduce the computational time by parallel filtration.

```
output <- filterAndTrim(fwd = fnF, filt = filtF, rev = fnR, filt.rev = filtR,
                        truncLen=c(240,160), maxN=0, maxEE=c(2,2), truncQ=2,
                        trimLeft = 10, rm.phix=TRUE, compress=TRUE,
                        multithread=TRUE)
head(output)
```

```
##                               reads.in reads.out
## F3D0_S188_L001_R1_001.fastq       7793      7139
## F3D1_S189_L001_R1_001.fastq       5869      5314
## F3D141_S207_L001_R1_001.fastq     5958      5478
## F3D142_S208_L001_R1_001.fastq     3183      2926
## F3D143_S209_L001_R1_001.fastq     3178      2955
## F3D144_S210_L001_R1_001.fastq     4827      4323
```

```
#Shows the difference of reads before and after filtering and trimming
```

Data dereplication

To save computing resource and memory once more, I use derepFastq to dereplicate amplification sequences in fastq file. By this step, identical reads are collapsed together into unique sequences. Consensus quality information and count of each unique sequence are recorded.

```
derepF <- derepFastq(filtF, verbose=TRUE)
derepR <- derepFastq(filtR, verbose=TRUE)
```

Here I create another function to check the detailed features of sequences after dereplication. “derepdetail” will return the general information of dereplicated sequences, the two most abundant sequences with their count, a statistical summary and a histogram of the quality score profile.

```r
derepdetail <- function(a){
  print(a)
  cat("\nThe Most Abundant Unique Sequences and Their Count:\n")
  uniques <- head(a[["uniques"]],2)
  print(uniques)
  cat("\nThe Summary of Quality Score:\n")
  summary <- summary(as.vector(a[["quals"]]))
  print(summary)
  hist(as.vector(a[["quals"]]), xlab = "Quality Score", ylab="Base/Position",
       main = "Quality Score Distribution")
}
```

```r
derepdetail(derepF[[1]])
```

```
## derep-class: R object describing dereplicated sequencing reads
## $uniques: 7139 reads in 1866 unique sequences
##   Sequence lengths: min=230, median=230, max=230
## $quals: Quality matrix dimension:  1866 230
##   Consensus quality scores: min=12, median=37.74074, max=39
## $map: Map from reads to unique sequences:  2 2 49 1 62 ...
##
## The Most Abundant Unique Sequences and Their Count:
## GCGAGCGTTATCCGGATTTATTGGGTTTAAAGGGTGCGCAGGCGGAAGATCAAGTCAGCGGTAAAATTGAGAGGCTCAACCTCTTCGAGCCGTTGAAACT
##
## GCGAGCGTTATCCGGATTTATTGGGTTTAAAGGGTGCGTAGGCGGCCTGCCAAGTCAGCGGTAAAATTGCGGGGCTCAACCCCGTACAGCCGTTGAAACT
##
##
## The Summary of Quality Score:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   12.00   37.00   37.74   35.81   38.00   39.00
```

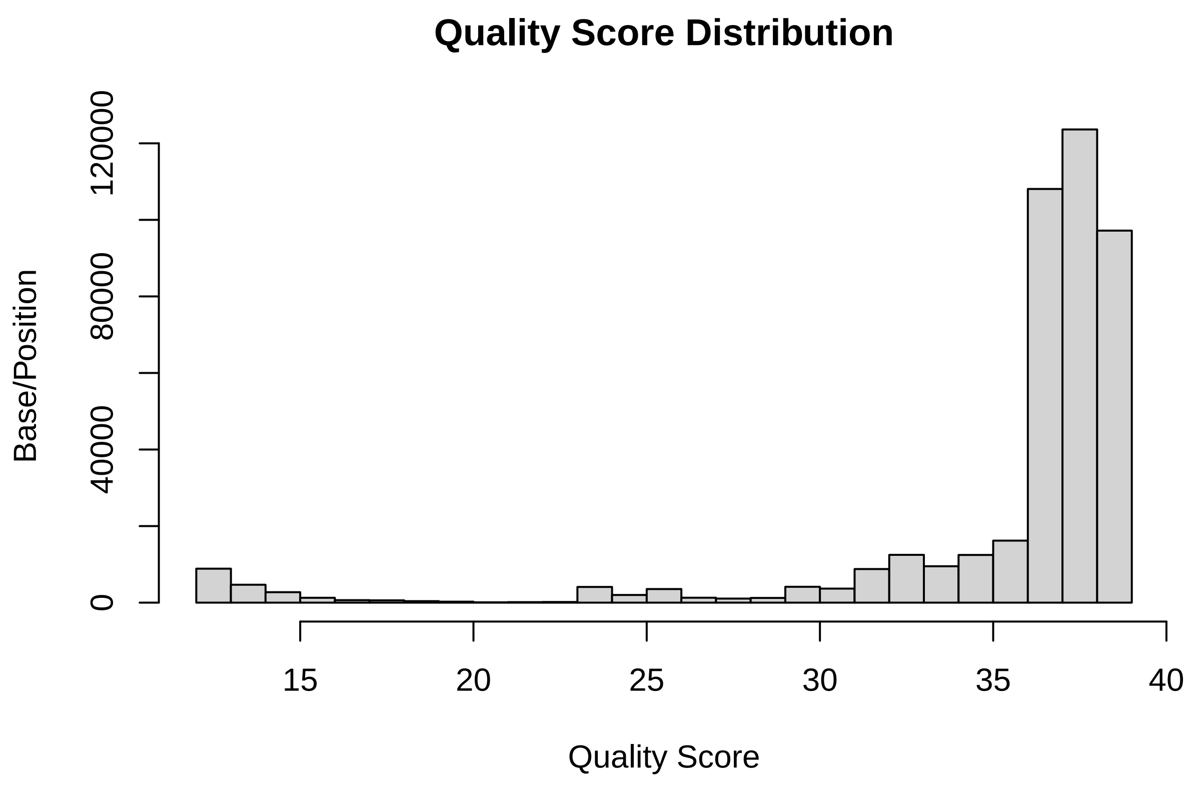

Figure 3. The distribution of quality score per base position for forward reads.

For forward reads, there are 1866 unique sequences out of 7139 in the first sample. All sequence length is 230 bp as expected.For reverse reads, there are 1567 unique sequences, all sequence is 150 bp. Both forward reads and reverse reads, mean (35.81, 35.90) and median (37.74, 38) are high as expected, which indicates the previous filtering and trimming step are necessary to ensure the overall quality. Meanwhile, both histogram show that among all, only a few positions are with low quality score (<20).

Error model inference

The following steps are to learn the parameterized error model from filtered and trimmed amplicon sequences (both forward and reverse read). I use the dereplicated sequences for error model learning, because it is recommended to use dereplicated sequences to save computational time.

```
errdF <- learnErrors(derepF, multithread=TRUE)
```

```
## 32228750 total bases in 140125 reads from 20 samples will be used for learning the error rates.
```

```
errdR <- learnErrors(derepR, multithread=TRUE)
```

```
## 21018750 total bases in 140125 reads from 20 samples will be used for learning the error rates.
```

Then we can visualize the estimated error rate for each possible transition (eg. biological A->C/G/T), in total 16 × 41 transition probabilities.

```
plotErrors(errdF, nominalQ=TRUE)
```

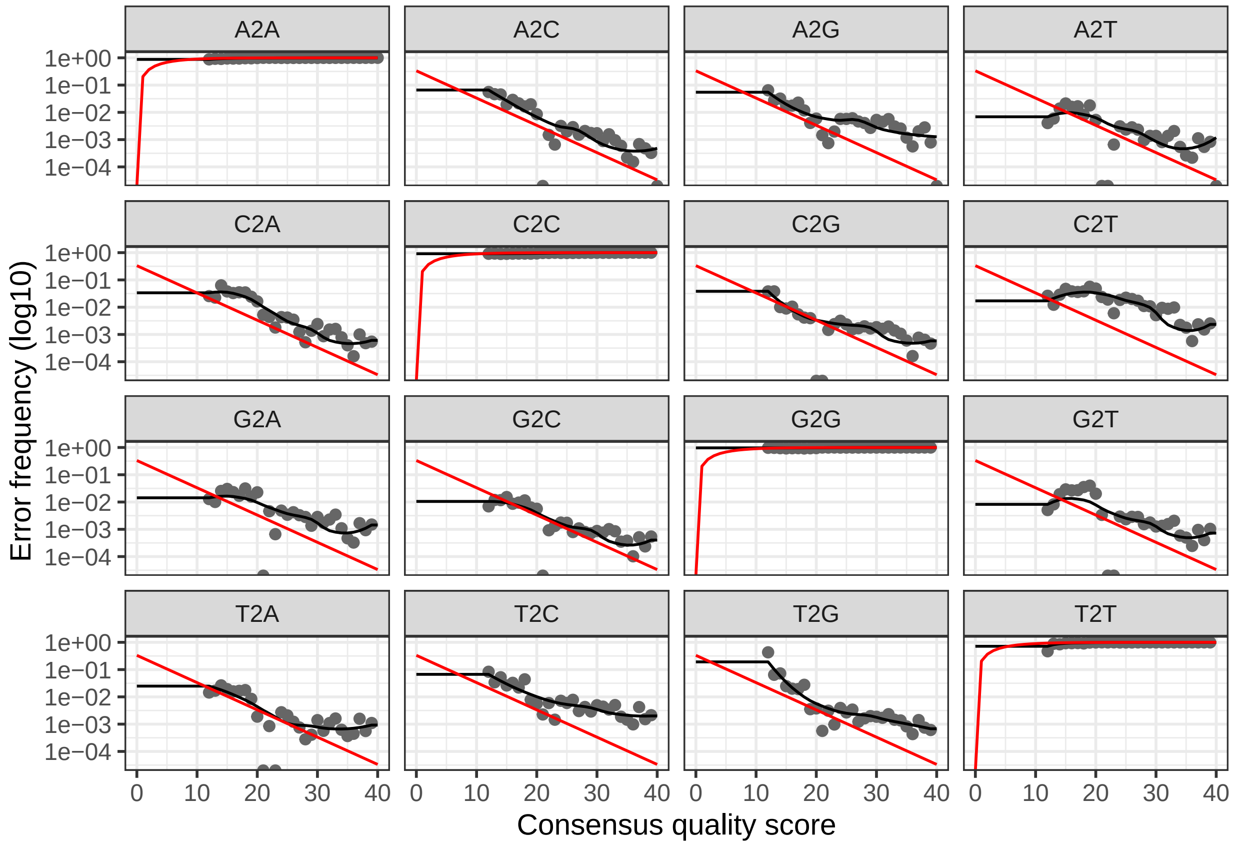

Figure 4. The error information for forward reads. Each point represents the observed error rate for each quality score. The black line represents the error rate estimated by the convergence of algorithmic. The red line shows the expected error rate with the nominal definition of the Q-score.

All the plots show a good fit between observed (points) and estimated error rate (black lines), and the error rate decreases as quality score increases, indicating the error models are dependable for following procedures.

Sample inference

Here I am applying function "dada", the core sample inference algorithm "Divisive Amplicon Denoising Algorithm", to the filtered and trimmed sequence data of 20 samples. The "dada" will remove all sequencing errors and return the inferred composition of the samples.

```
dadaF <- dada(derepF, err = errdF, multithread = TRUE)
```

```
## Sample 1 - 7139 reads in 1866 unique sequences.
## Sample 2 - 5314 reads in 1543 unique sequences.
## Sample 3 - 5478 reads in 1415 unique sequences.
## Sample 4 - 2926 reads in 870 unique sequences.
## Sample 5 - 2955 reads in 901 unique sequences.
## Sample 6 - 4323 reads in 1227 unique sequences.
## Sample 7 - 6762 reads in 1690 unique sequences.
## Sample 8 - 4580 reads in 1373 unique sequences.
## Sample 9 - 15695 reads in 3423 unique sequences.
## Sample 10 - 11448 reads in 2629 unique sequences.
## Sample 11 - 12064 reads in 2905 unique sequences.
## Sample 12 - 5054 reads in 1503 unique sequences.
## Sample 13 - 18130 reads in 3493 unique sequences.
## Sample 14 - 6275 reads in 1410 unique sequences.
## Sample 15 - 4068 reads in 1148 unique sequences.
## Sample 16 - 7394 reads in 1760 unique sequences.
## Sample 17 - 4772 reads in 1127 unique sequences.
## Sample 18 - 4890 reads in 1302 unique sequences.
## Sample 19 - 6525 reads in 1613 unique sequences.
## Sample 20 - 4333 reads in 845 unique sequences.
```

```
dadaR <- dada(derepR, err = errdR, multithread = TRUE)
```

```
## Sample 1 - 7139 reads in 1567 unique sequences.
## Sample 2 - 5314 reads in 1285 unique sequences.
## Sample 3 - 5478 reads in 1291 unique sequences.
## Sample 4 - 2926 reads in 832 unique sequences.
## Sample 5 - 2955 reads in 849 unique sequences.
## Sample 6 - 4323 reads in 1249 unique sequences.
## Sample 7 - 6762 reads in 1749 unique sequences.
## Sample 8 - 4580 reads in 1210 unique sequences.
## Sample 9 - 15695 reads in 3280 unique sequences.
## Sample 10 - 11448 reads in 2398 unique sequences.
## Sample 11 - 12064 reads in 2692 unique sequences.
## Sample 12 - 5054 reads in 1376 unique sequences.
## Sample 13 - 18130 reads in 3138 unique sequences.
## Sample 14 - 6275 reads in 1356 unique sequences.
## Sample 15 - 4068 reads in 1108 unique sequences.
## Sample 16 - 7394 reads in 1590 unique sequences.
## Sample 17 - 4772 reads in 1051 unique sequences.
## Sample 18 - 4890 reads in 1122 unique sequences.
## Sample 19 - 6525 reads in 1453 unique sequences.
## Sample 20 - 4333 reads in 701 unique sequences.
```

Let's take a look at the sequence composition for forward reads after dada.

```
## dada-class: object describing DADA2 denoising results
## 130 sequence variants were inferred from 1866 input unique sequences.
## Key parameters: OMEGA_A = 1e-40, OMEGA_C = 1e-40, BAND_SIZE = 16
```

```
## GCGAGCGTTATCCGGATTTATTGGGTTTAAAGGGTGCGCAGGCGGAAGATCAAGTCAGCGGTAAAATTGAGAGGCTCAACCTCTTCGAGCCGTTGAAACT
##
## GCGAGCGTTATCCGGATTTATTGGGTTTAAAGGGTGCGTAGGCGGCCTGCCAAGTCAGCGGTAAAATTGCGGGGCTCAACCCCGTACAGCCGTTGAAACT
##
```

```
##          [,1]     [,2]     [,3]
## [1,] 36.73220 36.87966 37.82712
## [2,] 36.87375 36.86373 37.84770
```

For the first sample, 130 sequence variants were inferred from 1866 input unique sequences (forward reads), abundance of each sequence, and average quality per position are recorded.

Merge paired reads and ASV table

By this step, denoised pairs of forward and reverse reads are merged together to produce full sequence, pairs that contain mismatches in the overlap region are discarded.

```
mergers <- mergePairs(dadaF, derepF, dadaR, derepR, verbose = TRUE)
head(mergers[[1]])
```

```
##
## 1 GCGAGCGTTATCCGGATTTATTGGGTTTAAAGGGTGCGCAGGCGGAAGATCAAGTCAGCGGTAAAATTGAGAGGCTCAACCTCTTCGAGCCGTTGAAAC
## 2 GCGAGCGTTATCCGGATTTATTGGGTTTAAAGGGTGCGTAGGCGGCCTGCCAAGTCAGCGGTAAAATTGCGGGGCTCAACCCCGTACAGCCGTTGAAAC
## 3 GCGAGCGTTATCCGGATTTATTGGGTTTAAAGGGTGCGTAGGCGGGCTGTTAAGTCAGCGGTCAAATGTCGGGGCTCAACCCCGGCCTGCCGTTGAAAC
## 4 GCGAGCGTTATCCGGATTTATTGGGTTTAAAGGGTGCGTAGGCGGGCTTTTAAGTCAGCGGTAAAAATTCGGGGCTCAACCCCGTCCGGCCGTTGAAAC
## 5 GCGAGCGTTATCCGGATTTATTGGGTTTAAAGGGTGCGCAGGCGGACTCTCAAGTCAGCGGTCAAATCGCGGGGCTCAACCCCGTTCCGCCGTTGAAAC
## 6 GCGAGCGTTATCCGGATTTATTGGGTTTAAAGGGTGCGTAGGCGGGATGCCAAGTCAGCGGTAAAAAAGCGGTGCTCAACGCCGTCGAGCCGTTGAAAC
##   abundance forward reverse nmatch nmismatch nindel prefer accept
## 1       582       1       1    148         0      0      1   TRUE
## 2       495       2       2    148         0      0      1   TRUE
## 3       451       3       4    148         0      0      1   TRUE
## 4       442       4       3    148         0      0      2   TRUE
## 5       345       5       6    148         0      0      1   TRUE
## 6       283       6       5    148         0      0      2   TRUE
```

Final results are returned as a list of data frames. Each data frame collects the merged unique sequences (sequence) and their count (abundance); forward and reverse represent the index of the forward/reverse sequence contributed to the merged sequence. Merged sequence = forward-only + exact overlapping + reverse-only portion.

To create an amplicon sequence variant table, sample-by-sequence.

```
seqtab <- makeSequenceTable(mergers)
dim(seqtab)
```

```
## [1]  20 286
```

Table has 20 samples with 286 true amplicon sequence variants, we are able to see each inferred sequence and its count in each sample.

Chimeras removal

Chimeras are easily introduced during PCR due to incomplete amplification. As substitution and indel errors are removed by previous denoising, sequences with chimeras will be removed once more by "removeBimeraDenovo".

```
seqtab.nochim <- removeBimeraDenovo(seqtab, method="consensus",
                                    multithread=TRUE, verbose=TRUE)
```

```
## Identified 52 bimeras out of 286 input sequences.
```

```
## [1]  20 234
```

There are 234 biological amplicon sequence variants (ASV) remained, that chimeras make up about 3.348728% of the total merged sequences in 20 samples.

Track reads through the DADA2 pipeline

This step is performed as a sanity check, tracking the number of sequence read throughout the whole pipeline for all samples

```
##        input filtered/trimmed denoisedF denoisedR merged nonchimera
## F3D0    7793             7139      7021      7024   6667       6655
## F3D1    5869             5314      5252      5252   5028       5028
## F3D141  5958             5478      5369      5370   4995       4867
## F3D142  3183             2926      2822      2847   2611       2546
## F3D143  3178             2955      2837      2885   2598       2564
## F3D144  4827             4323      4175      4230   3690       3540
```

For each sample, compared to the input data, majority sequences are kept as the true biological sequences after all the quality control steps and the final merge.

Taxonomy assignment

To check if the sample size and sample completeness are appropriate through a sample-based amplicon sequence variants accumulation curve with vegan package function "specaccum"

```r
AC <- specaccum(seqtab.nochim)
plot(AC, xlab = "Number of Sample", ylab = "True ASV",
     main = " Sample-based ASV Accumulation Curve")
```

234 biological ASVs are determined from 20 samples. As the curve starts to level off slightly at the end, the sample size would be considered enough for this specific dataset to reveal biodiversity. More data will continue to produce a small number of new ASVs (taxon).

It is interesting to see that DADA2 provides the naive Bayesian classifier method to assign the taxonomy. I will use function "assignTaxonomy" for the taxonomy assignment of 234 upstream derived ASVs from a training set of reference sequences with known taxonomy. Reference database Silva 132 for 16S is obtained from https://zenodo.org/record/1172783#.Ybr6VC_71hA. (Please place Silva 132 files into the same folder as other fastq files.)

```
##    Kingdom        Phylum       Class          Order          Family       Genus
## 1 Bacteria Bacteroidetes Bacteroidia Bacteroidales Muribaculaceae        <NA>
## 2 Bacteria Bacteroidetes Bacteroidia Bacteroidales Muribaculaceae        <NA>
## 3 Bacteria Bacteroidetes Bacteroidia Bacteroidales Muribaculaceae        <NA>
## 4 Bacteria Bacteroidetes Bacteroidia Bacteroidales Muribaculaceae        <NA>
## 5 Bacteria Bacteroidetes Bacteroidia Bacteroidales Bacteroidaceae Bacteroides
## 6 Bacteria Bacteroidetes Bacteroidia Bacteroidales Muribaculaceae        <NA>
##   Species
## 1    <NA>
## 2    <NA>
## 3    <NA>
## 4    <NA>
## 5    <NA>
## 6    <NA>
```

Most of the 16s rRNA sequences are identified to the family level, very few species are assigned. This is acceptable since sequences from V4 region only contain partial information among all nine hypervariable regions, and reference for mouse gut microbiota is limited in Silva 132 database.

Evaluate accuracy

There is a mock sample in the original input data, with a mixture of sequences from 20 known bacteria strains, which can be used a the reference to assess the accuracy of DADA2 inferred results.

```
dada.mock <- seqtab.nochim["Mock",]
dada.mock <- sort(dada.mock[dada.mock>0], decreasing = TRUE)
#Remove NAs for ASVs in the mock sample
cat("DADA2 inferred", length(dada.mock), "sequences present in the Mock community.\n")
```

```
## DADA2 inferred 20 sequences present in the Mock community.
```

```
#Check if ASVs identified are the exact matches of reference sequences in mock community.
mock.ref <- getSequences(file.path(path, "HMP_MOCK.v35.fasta"))
match.ref <- sum(sapply(names(dada.mock), function(x) any(grepl(x, mock.ref))))
match.ref
```

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

All 20 ASVs determined from DADA2 are exact matches to the reference sequences. The DADA2 analysis on this dataset produces an accuracy of 100%.

Phylogenetic Visulation

Following, I will use package "phyloseq" to visualize the taxonomic profile of microbial communities constructed with DADA2 pipeline from post-weaning mouse's fecal samples.

```
#Gather information that is encoded in sample names.
subject <- sapply(strsplit(sample.names, "D"), `[`, 1)
day <- as.integer(sapply(strsplit(sample.names, "D"), `[`, 2)) #The day of post-weaning.
samdf <- data.frame(Subject=subject, Day=day)
samdf$When <- "Early"
samdf$When[samdf$Day>100] <- "Late"
rownames(samdf) <- sample.names
```

```
#Build phyloseq-class object with ASV table produced and taxonomy annotated table.
physeq <- phyloseq(otu_table(seqtab.nochim, taxa_are_rows = FALSE),
               sample_data(samdf),
               tax_table(taxa))
#Remove sample of mock community.
physeq <- prune_samples(sample_names(physeq) != "Mock", physeq)
#Save ASV sequences to physeq$refseq.
dna <- DNAStringSet(taxa_names(physeq))
names(dna) <- taxa_names(physeq)
physeq <- merge_phyloseq(physeq, dna)
taxa_names(physeq) <- paste0("ASV", seq(ntaxa(physeq)))
physeq
```

```
## phyloseq-class experiment-level object
## otu_table()   OTU Table:         [ 234 taxa and 19 samples ]
## sample_data() Sample Data:       [ 19 samples by 3 sample variables ]
## tax_table()   Taxonomy Table:    [ 234 taxa by 7 taxonomic ranks ]
## refseq()      DNAStringSet:      [ 234 reference sequences ]
```

I will select top 20 sequences from all 234 sequences, and present their taxonomic distribution (Family).

```
top20 <- names(sort(taxa_sums(physeq), decreasing = TRUE))[1:20]
physeq.top20 <- transform_sample_counts(physeq, function(OTU) OTU/sum(OTU))
physeq.top20 <- prune_taxa(top20, physeq.top20)
plot_bar(physeq.top20, fill="Family", title = "Microbiome Profile (Family)") +
  facet_wrap(~When, scales="free_x")
```

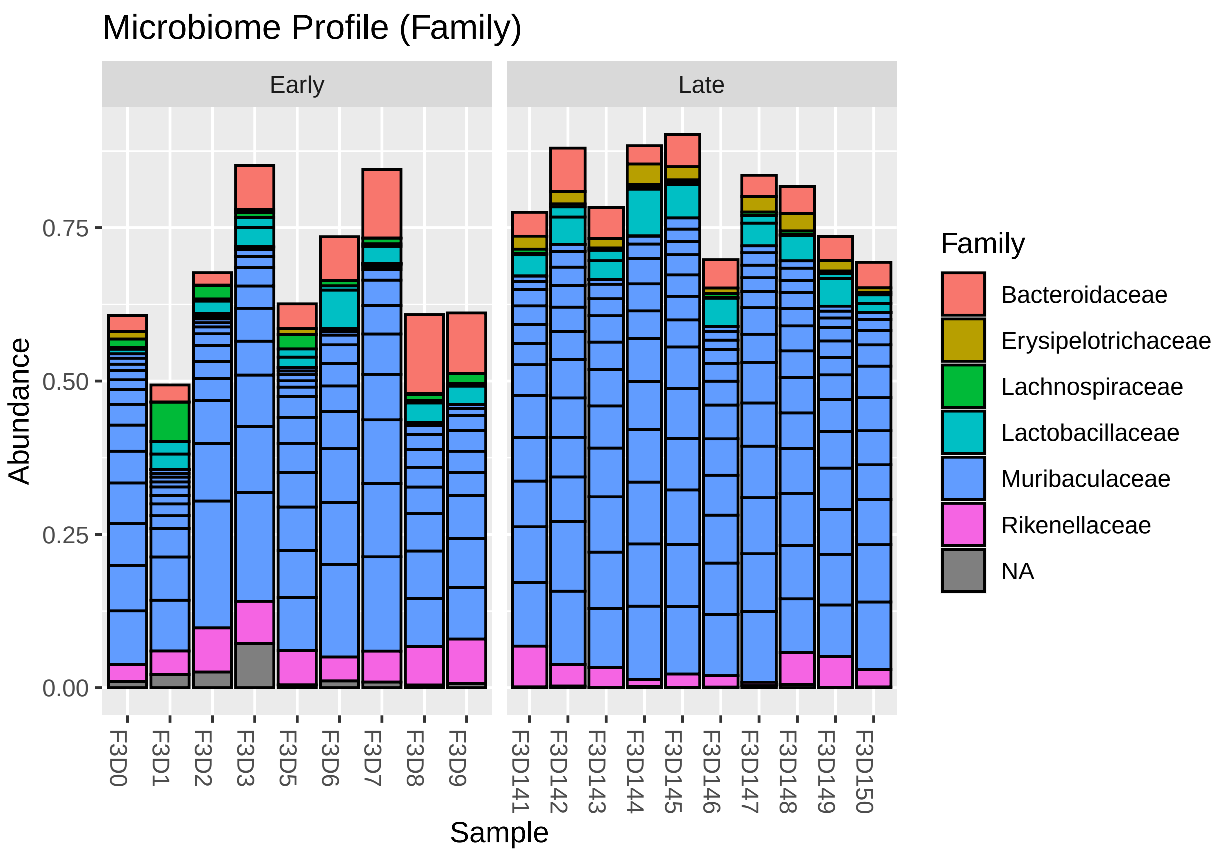

Figure 5. Family profile on the first 20 sequences, *Muribaculaceae* is the most abundant.

Comparing the early samples and late samples, both taxonomic profile barplots show higher abundance of total bacteria in the late samples.

```
#use principal coordinate analysis to ordinate the samples.
physeq.prop <- transform_sample_counts(physeq, function(otu) otu/sum(otu))
ord.nmds.bray <- ordinate(physeq.prop, method="PCoA")
plot_ordination(physeq.prop , ord.nmds.bray, color="When",
                title="Principal Coordinate Analysis Plot")
```

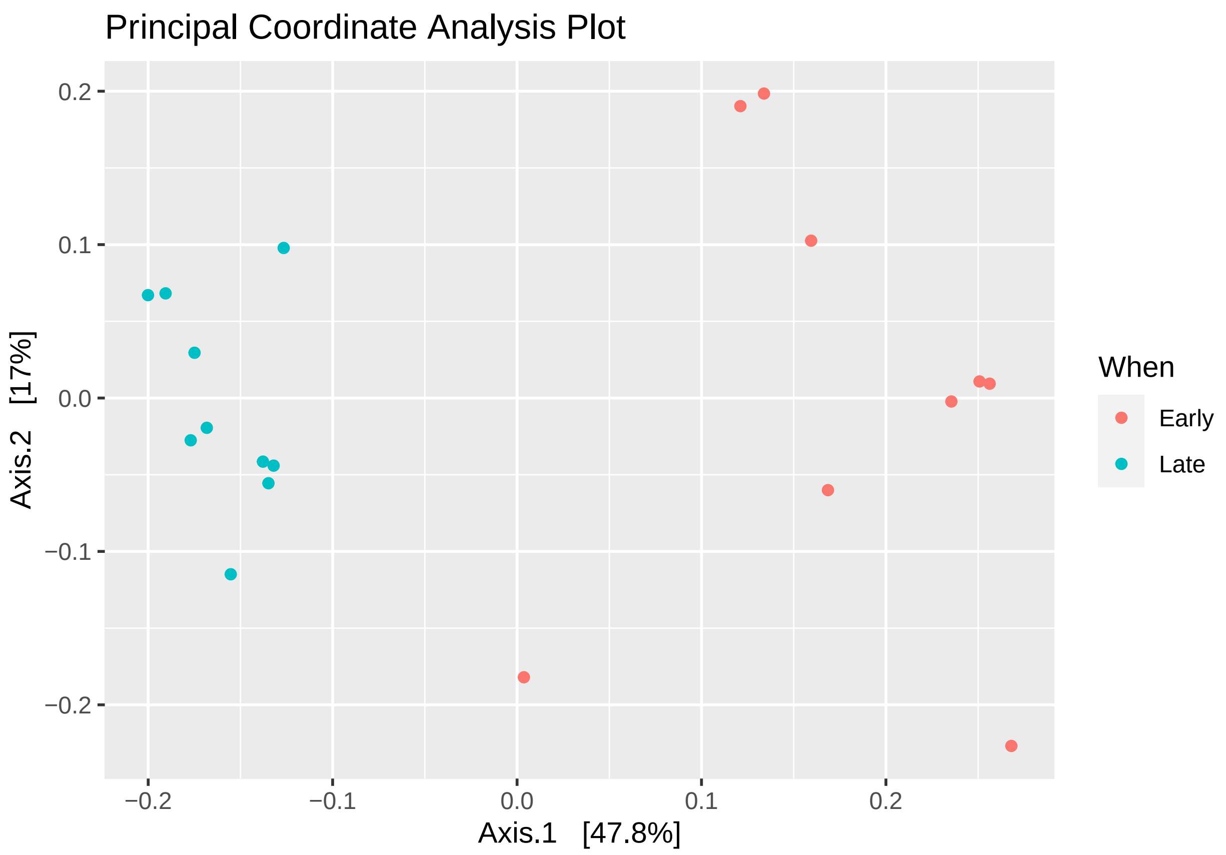

Figure 6. Principal Coordinate Analysis Plot. Early samples (red) and late samples (blue) formed 2 separate clusters along the x-axis, which indicates a significant difference based on the post-post-weaning days.

We can also visualize the alpha diversity, Shannon and Simpson measures are used.

```r
plot_richness(physeq, x = "Day", color = "When", measures = c("Shannon", "Simpson"))
```

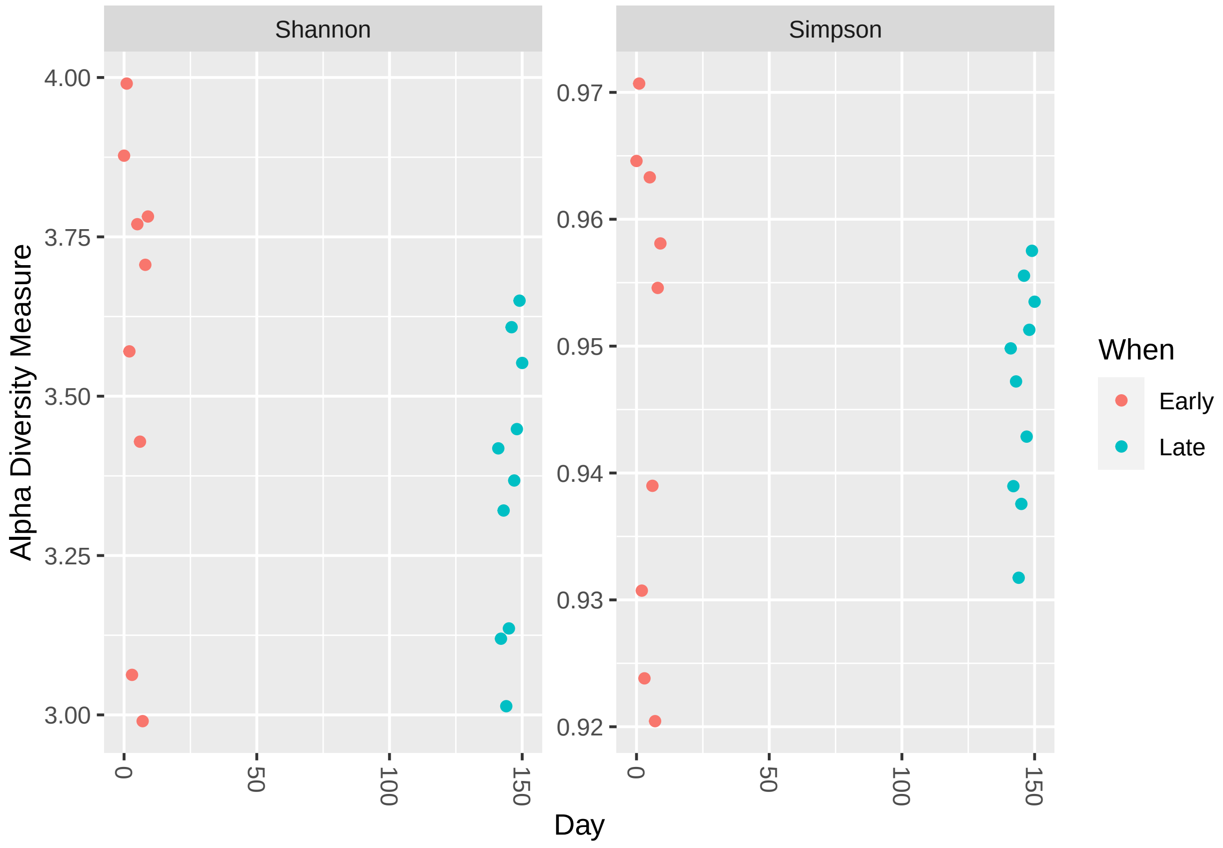

Figure 7. The alpha diversity of samples based on the post-weaning day. The larger the Shannon index, the higher the diversity of bacterial community is. The smaller the Simpson index is, the higher the diversity. No obvious difference on alpha-diversity between early and late samples can be concluded from these 2 figures.

As phyloseq provide all kinds of visualization tools for exploring microbiome profile, I found it is possible to generate phylogenetic trees with function "phy_tree". Therefore, I will try to create a tree for the ASVs inferred from DADA2.

```
#Create a random phylogenetic tree with the "ape" package
random_tree = rtree(ntaxa(physeq), rooted=TRUE, tip.label=taxa_names(physeq))
#Merged the random tree into previous phyloseq-class object
physeq1 = merge_phyloseq(physeq, random_tree)
physeq1.20 = names(sort(taxa_sums(physeq1), decreasing = TRUE)[1:20])
ps20 = prune_taxa(physeq1.20, physeq1)
plot_tree(ps20, color = "Family", shape="Phylum", label.tips= "taxa_names",
          title = "Phylogenetic Tree for the Top 20 ASVs", text.size = 3)
```

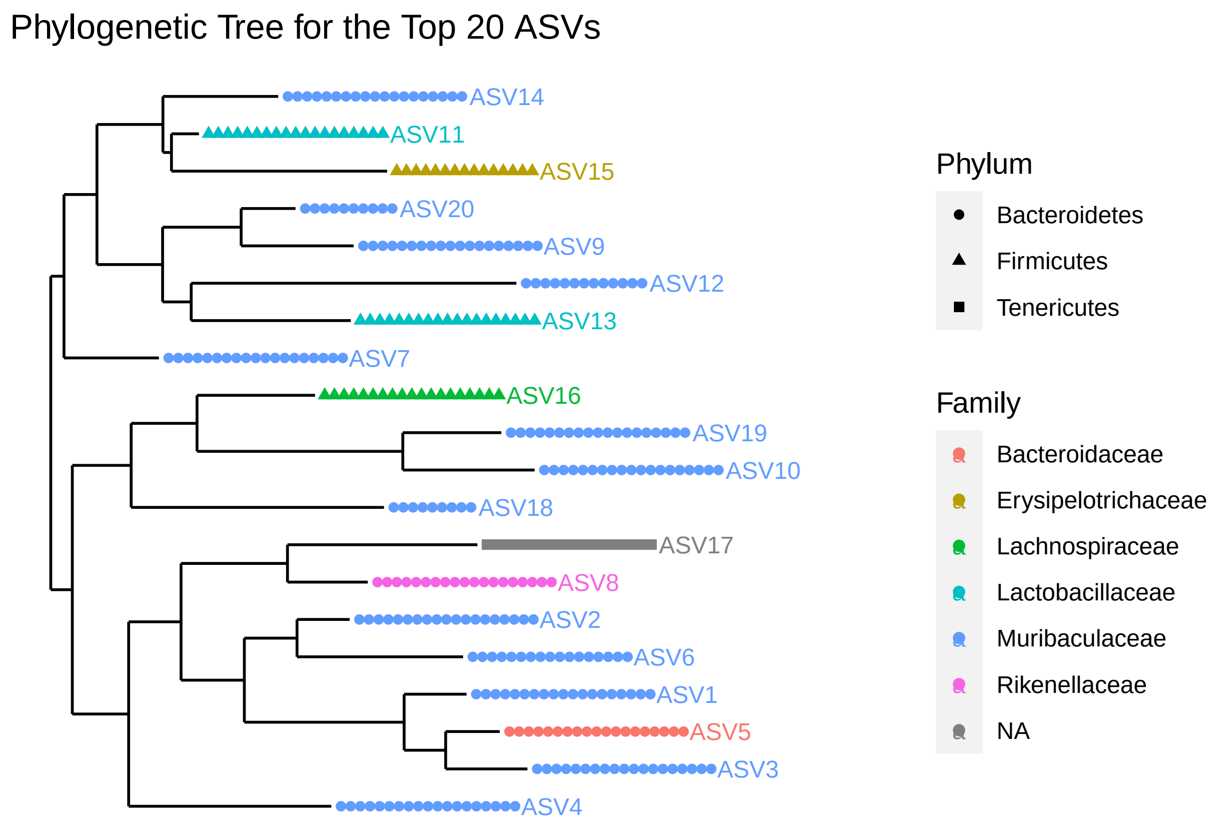

Figure 8. The phylogenetic tree for the Top 20 ASVs. Different shapes represent different Phylum, *Bacteroidetes* is the most abundant. Different color represent different Family, *Muribaculaceae* is the most abundant.

Results and Discussion

This assignment aims to perform a microbiome analysis using 16s amplicon sequences. The whole DADA2 pipeline runs smoothly on my 2017 MacBook Pro, though certain main analysis steps do require relatively longer processing time (comparatively small-scale fastq files are used for analysis). DADA2 is capable of conducting a comprehensive and accurate measurement of the microbiome community. As DADA2 has strict criteria on its input sequence data, the first step is to ensure dataset meeting all the requirements, some pre-processing steps might need to be done before input into DADA2. I used 20 pair-end fastq files (forward and reverse reads) to demonstrate the workflow. In the DADA2 pipeline, several quality control procedures are essential for deleting the possible errors produced by previous Illumina sequencing (filtering, trimming, dereplication, denoising, chimeras removing). Depending on the different data inputs, user can customize various parameters to trim and filter sequences based on the quality score(Figure 1 & 2). Meanwhile, depending on the device's computing power, users could change parameter settings to increase the processing speed while maintaining the accuracy of analysis, but sacrifice a little on the reads of output. I have tried several different filtering parameters for this assignment; however, in order to reduce the computational burden (computer crashed many times), I only retain the most efficient one with the highest accuracy. And data dereplication would be highly recommended for any large-scale data to reduce processing time, that repetitive sequences will be collapsed together, the abundance of each unique sequence, and the mean quality score will be recorded for the following steps. Finally, 234 biological amplicon sequence variants are inferred from 20 samples. By comparing the ASVs obtained from DADA2 with the mock community, it is pleasing to see that there is no residual error remaining after the DADA2 pipeline when analysis using the errors model inferred by DADA2. As a result, DADA2 achieves 100% accuracy on sequences inference of this specific data set.

Meanwhile, DADA2 includes a convenient build-in function for taxonomy assignment,and I was able to assign the majority of the sequences to Family. A more in-depth reference database is required to perform the species-level assignment. I further explore the phylogenetic diversity by combining the ASVs and taxa information with phyloseq. 9 merged fecal sample sequences are separated into groups (early vs. late) based on the post-weaning days. Fecal samples in late groups have a higher overall count on bacteria. By looking at the taxonomic abundance plot, Phylum *Bacteroidetes* and Family *Muribaculaceae* are the most abundant in all mouse fecal samples (Figure 5). When comparing the two groups, the early group exhibits a larger abundance of *Lactobacillaceae* in the first two samples, which might be because the mouse was still digesting milk and not eating anything else. In addition, the principal coordinate analysis plot indicates a significant difference between groups, that two groups form clear clusters along the x-axis (Figure 6).

For future analysis, I would continue to work on creating a more complex and sophisticated tree for all the ASVs identified by DADA2, rather than just the top 20, with phyloseq or other tree-generating packages. Furthermore, performing analysis on a large-scale dataset with DADA2 could be a meaningful next step, that could be potentially employed into a larger project.

Acknowledgements

Jacqueline May

Thanks for the generous general help and tips during help session.

Dr. Sarah Adamowicz

Really appreciate all the tutorial, extra materials, and guidance provided for this specific topic.

References

https://bioconductor.org/packages/devel/bioc/vignettes/dada2/inst/doc/dada2-intro.html

http://joey711.github.io/phyloseq/index.html

https://benjjneb.github.io/dada2/tutorial.html

https://mothur.org/wiki/miseq_sop/

https://www.yunbios.net/phyloseq.html

Illumina (2021). FASTQ Files Explained. https://support.illumina.com/bulletins/2016/04/fastq-files-explained.html.

Callahan, McMurdie, P. J., & Holmes, S. P. (2017). Exact sequence variants should replace operational taxonomic units in marker-gene data analysis. The ISME Journal, 11(12), 2639–2643. https://doi.org/10.1038/ismej.2017.119

Callahan, McMurdie, P. J., Rosen, M. J., Han, A. W., Johnson, A. J. A., & Holmes, S. P. (2016). DADA2: High-resolution sample inference from Illumina amplicon data. Nature Methods, 13(7), 581–583. https://doi.org/10.1038/nmeth.3869

McMurdie, & Holmes, S. (2013). phyloseq: an R package for reproducible interactive analysis and graphics of microbiome census data. PloS One, 8(4), e61217–e61217. https://doi.org/10.1371/journal.pone.0061217