An alternative method to processing sequences for taxonomic classification

Kaitlyn Hobbs

August 2018

4483E

Supervisor: Dr. Gloor

Contents

1 Terminology

Term	Definition
ASV	Amplicon Sequence Variant
OTU	Operational Taxonomic Unit
Feature	Either an OTU or ASV depending on the pipeline applied
Technical variant/technical error	Sequencing error introduced by PCR or Illumina
Read/sequence variant	Sequence obtained from HTS instrument
MS	Multiple sclerosis

2 Abstract

Microbial communities influence host health. To profile microbes and identify imbalances in these communities, the 16S ribosomal RNA gene is sequenced using high-throughput technology. Nonetheless, technical error introduced during sequencing coupled with read processing and data analysis can influence results. Deducing amplicon sequence variants is an alternative method to generating operational taxonomic units for read processing. In this study, an open-access 16S rRNA dataset was used to compare and assess the robustness of the ASV standard operative procedure against OTU methods. Additionally, a dataset comparing gut microbiomes between multiple sclerosis patients and healthy individuals with unknown compositions was used in attempt to validate published findings [1]. Ultimately, the ASV-based approach proved to generate more reliable results by excluding technical errors from its output and evading problems associated with clustering. In contrast to published results, no significant differences were found between MS patients and healthy individuals; however, vague reporting of computational methods prevented an exact replication of the study.

3 Introduction

3.1 Sequencing Microbiomes

50% of cells comprising humans belong to microbes, most which reside in the gastrointestinal tract [2, 3]. Microbial communities are distinguished by their environment, or "biome", and are known to partake in symbiotic or commensal relationships with their host [4]. Profiling these communities can help in identifying dysbiosis: an imbalance in a microbiome, which often directly impacts host health. For example, individuals with irritable bowel syndrome (IBS) have a different gut microbiota relative to healthy individuals [5]. This discovery provides insight to the potential role of gut microbes in IBS pathogenesis and opens the door for microbiota-directed therapies.

Typically, microbiota are profiled through sequencing of all or part of the 16S ribosomal RNA (rRNA) gene found natively in micro-organisms. It contains nine variable regions that differentiate with taxa and is used to associate microbes to taxonomic classes. Sequences are most frequently obtained through Next Generation Sequencing (NGS), processed using a computational pipeline, and referenced to an external database for taxonomic classification [6]. Though the high-throughput, low cost, and low false positive generation of Illumina instruments is preferred, technical errors can be introduced through sequence-by-synthesis [7, 8, 9]. DNA polymerases may introduce errors during PCR amplification, Illumina primer extension and amplification. Read processing pipelines output a count table, which includes the number of instances that a sequence was observed by the pipeline in each sample along with taxonomic classifications. From there, data can be analyzed to evaluate the relative abundance of a microbe in a biome. Errors introduced in the preliminary cycles of amplification can be propagated, ultimately, misrepresenting the identity and proportions of microbes [7, 8]. Accounting for this error depends on how the sequences are processed.

3.2 Methods to Processing Sequences

There are two main approaches to processing reads: by clustering sequences into operational taxonomic units (OTUs), or by deducing amplicon sequence variants (ASVs). OTUs are clusters of reads that share an arbitrary threshold of dissimilarity; for example, if a dissimilarity threshold was chosen at 3%, sequences that are 97% identical would be grouped together to form one OTU. This OTU is aligned with sequences housed in an external database that also share 97% sequence identity and assigned the corresponding taxonomy [10]. This processing method does not consider technical error introduced by DNA polymerases in PCR and Illumina instruments. As a result, sequence variation within an OTU cluster may generate both false positive and negative results by including technical errors and natural variation, respectively [11]. Incorrectly grouped technical errors misrepresent a sequence from one taxonomy for another. False negatives arise if a sequence containing natural variation is inappropriately clustered with sequence variants belonging to a distinct taxonomy. Consequently, data obtained by processing reads using OTU clustering are less reproducible and less ideal for inter-dataset comparisons [7, 11, 10].

Alternatively, inclusion of technical errors in count tables can be mediated by inferring the error rate for each base in each position using an ASV approach. Erroneous reads can be identified by their low abundance relative to true biological sequences and low beta-association with other sequences [11, 12]. Relative abundances provide an inference of the total count of a sequence in an environment [7, 10, 13]. ASV methodolgy allows for single nucleotide resolution, distinction between natural and technical variation, and evaluation of relative abundances [7]. Together, interpretations made from generating ASVs are more robust, reproducible, and better suited for inter-dataset comparisons. Though the ASV-based approach is specific to Illumina, analogous methods exist for other NGS procedures [10].

3.3 Research Premise

The bidirectional relationship between gut microbiota and the central nervous system has caught the attention of researchers [14]. Relationships found between microbes and hosts have ignited the Human Microbiome Project, which aims to use

high-throughput technologies to profile host-associated microbiomes and elucidate their role in medical conditions [15]. In pursuit of these goals, it is necessary to optimize accuracy of each stage in profiling microbial communities. Unfortunately, although many publications have drawn conclusions supporting the influence of the gut microbiome on host health, these claims has yet to be reproduced. This research study has two focuses: testing the robustness of the ASV-based processing method in effort to improve accuracy in microbiome profiling and validating the presence of dysbiosis in multiple sclerosis shown by published researches [1].

4 Datasets

Jangi et al reported that individuals with multiple sclerosis (MS) have gut dysbiosis on the phylum and genera level compared to healthy individuals. Specifically, the *Prevotella*, *Sarcina*, *Sutterella*, *Butyricimonas*, *Akkermansia*, and *Methanobrevibacter* genera differed in abundances. They also found that all MS patients (untreated or treated with either Interferon or Copaxone) differed in abundance from healthy individuals in the *Euryarcheota* and *Verrucomicrobia* phyla [1]. The researchers clustered reads at a 97% identity similarity and used DESeq2 to perform statistical tests for significant differences in taxa with a p-value of 0.05. The p-value was corrected using the Benjamini-Hochberg method with a false discovery rate threshold of 0.1 [1].

The control dataset comprises both mock communities containing 15 known microbes and variable regions synthesized by Tourlousse et al with no known sequence identity to microbial taxonomic groups [16].

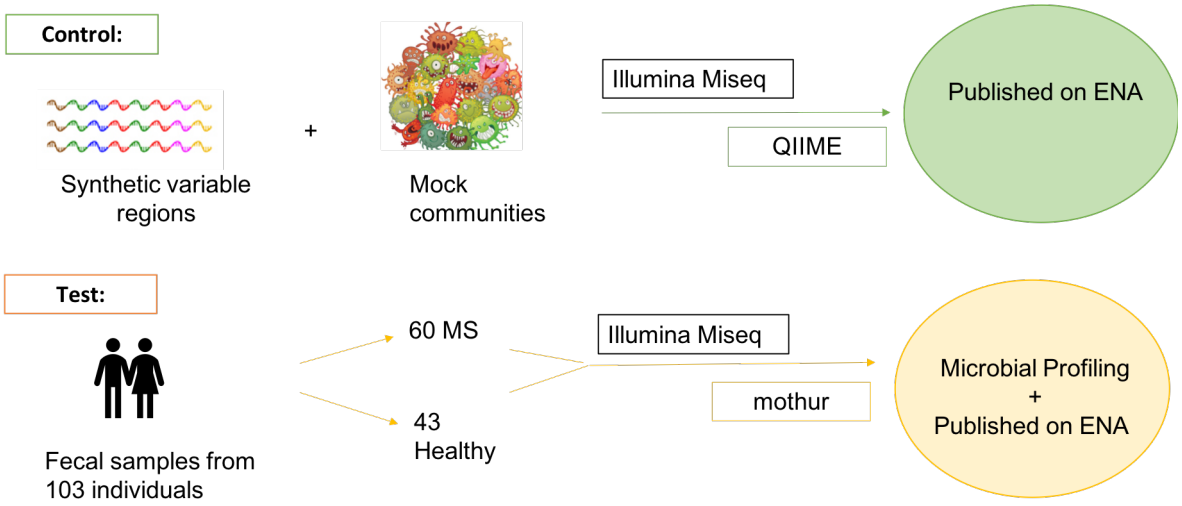

Figure 1: **Overview of control and test datasets.** Datasets were retrieved from the European Bioinformatics Institute (EBI). The control dataset is comprised of synthetic variable regions with no known microbe identity and a series of mock communities composed of 15 known species. The test dataset is composed of variable 4 regions obtained from fecal samples of 43 healthy individuals and 60 multiple sclerosis patients. Sequences for control and test datasets were obtained from Illumina Miseq, processed using OTU-based pipelines, QIIME and mothur, respectively, and published on the European Nucleotide Archive (ENA). Both mothur and DADA2 pipelines reference the SILVA database while QIIME references GreenGene and Broad Microbiome Utilities' 16S Gold.

5 Methods

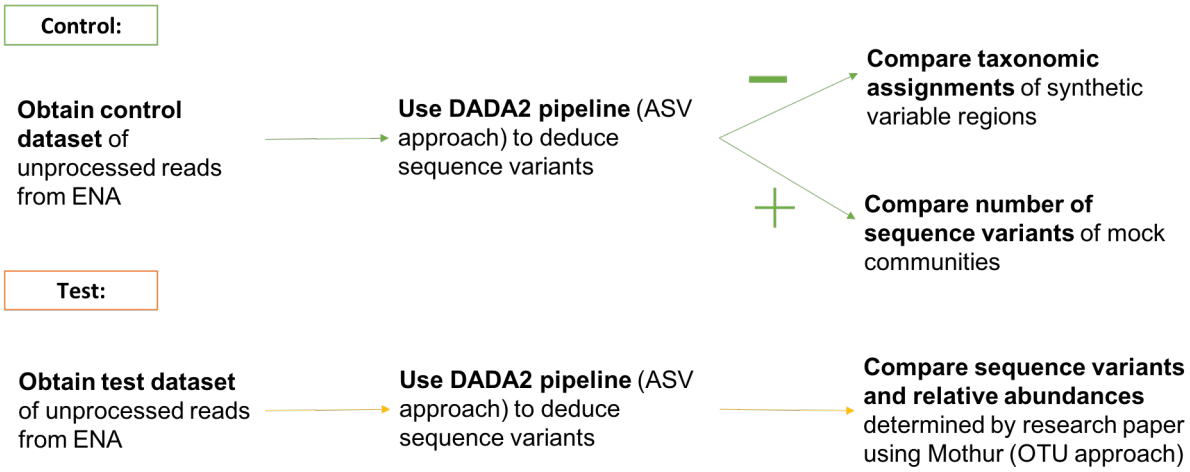

Figure 2: **An overview of methods**. Both raw Illumina Miseq reads from control and test datasets were retrieved from the European Nucleotide Archive (ENA) and processed using the ASV-based pipeline, DADA2, which references the SILVA metagenomic database to assign taxonomy. Synthetic variable regions from Tourlousse et al served as a negative control and taxonomic assignments were observed. Mock communities served as a positive control where the number of ASVs generated by DADA2 were compared to the number of OTUs found by QIIME. The MS dataset from Jangi et al was processed by DADA2 and compared to findings published by mothur [1]

R scripts used are included under Computational Methods in Supplementary Data.

6 Hypotheses

Since synthetic variable regions contain no sequence identity to known microbes, it was expected that they would not be given a taxonomic assignment after processing by DADA2 [16]. Conversely, because the mock communities have a known composition, DADA2 was expected to accurately identify these sequences with high resolution and detect more sequence variants than QIIME. Finally, the number of

reads from the multiple sclerosis study processed using DADA2 was thought to differ from the number of reads generated by mothur. This is because DADA2 is expected to give fewer false positives and negatives leaving only true biological variants for taxonomic assignment. Large variation within prokaryotics taxonomies makes comparing sequence variants as opposed to classifications more reliable [11].

7 Results

7.1 Positive Control

The DADA2 count table generated less than half the number of features found by QIIME. Subsequently, both count tables were run through the filtR function in R to identify technical variants. Technical errors are identified by two parameters: low beta-association between two features (ASVs or OTUs), as represented by ρ, and low abundance of each feature across all samples, represented by the center-log ratio (clr). filtR reported that DADA2 included no technical variants in its count table while QIIME included one (Table 1).

	DADA2	QIIME
No. of samples used	24	24
No. of features	985	1752
No. of technical variants*	0	1
Reference database	SILVA	Broad Microbiome Utilities' 16S Gold

Table 1: **Contrast between methodology and sequence variants produced of the DADA2 and QIIME pipelines.** 24 samples of mock communities spiked with synthetic variable regions were run through DADA2 and compared to results published by Tourlousse et al using QIIME. *Technical variants were determined using the filtR R package with a ρ cutoff of 0.7 and a clr cutoff of 5.

7.2 Negative Control

Since Illumina reads are short fragments, DADA2 generated ASVs were found in the longer synthetic spike-in sequences by nBLAST and taxonomic assignments of ASVs that aligned with the spike-ins were noted. Twelve spike-ins were found in 143 sequence variants within the control samples. Of those, 142 were recognized as members of the bacteria kingdom, twenty-five were assigned a phylum, and one was given a class (Figure 3).

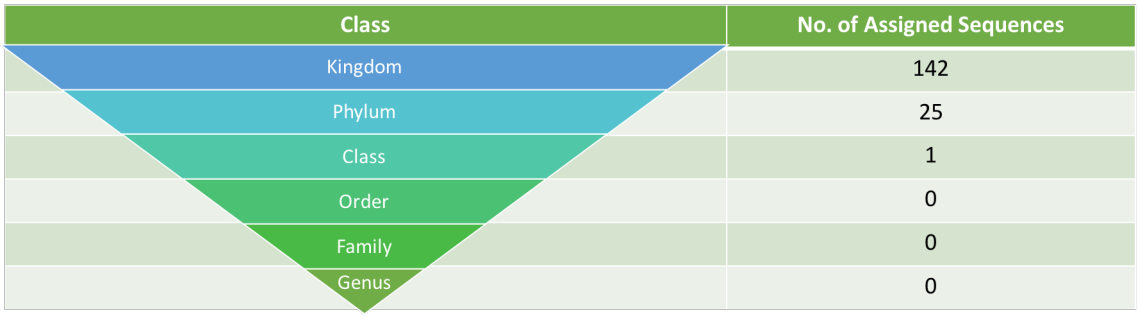

Figure 3: **Number of ASVs that were assigned a taxonomy by SILVA.** BLAST results with 100% identity and equal nucleotide length between DADA2 ASVs and the synthetic spike-in sequences confirmed the presence of the spike-ins in specific variants. If the variants were assigned a taxonomic class at a low rank, then they were considered a false positive.

7.3 Multiple Sclerosis Study

The OTU pipeline, mothur, found half the amount of features as DADA2 and included more technical errors (Table 2). In contrast to findings reported by Jangi et al, multiple analysis tools used to analyze the processed sequences did not find significant differences between gut microbiomes of MS patients and healthy individuals with either mothur or DADA2 count tables.

To qualitatively observe variances between subgroups in the MS dataset, a compositional biplot was created using data analysis tool, CoDaSeq [17]. In contrast to results published by Jangi et al, Illumina Miseq reads processed by DADA2 lead

	DADA2	mothur
No. of samples used	105	105
No. of features	2243	1191
No. of technical variants*	0	4
Reference database	SILVA	SILVA

Table 2: **Contrast between sequence variants produced and methodology of the DADA2 and mothur pipelines.** 105 fecal samples from 63 individuals were run through DADA2 and compared to results generated by mothur. Features are either ASVs or OTUs depending on the pipeline used. *Technical variants were determined using the filtR R function with a ρ cutoff of 0.7 and a clr cutoff of 5.

to no significant difference between subgroups (Figure 4A). Principal component analysis (PCA) is a multivariate analysis tool that observes compositional data in n-dimensional space while the PCA biplot projects the first and second dimensions, or principal components 1 and 2, respectively, that explain the most variation [18, 17]. It should be noted that the principal components are relatively low, indicating low explanatory power in the dataset (Figure 4). Low power suggests that any variation explained occurs due to random chance. Generally, the loadings biplot show any driving forces for variation observed in the sample biplot. In this dataset, the loadings are clustered around zero for both DADA2 and mothur, indicating that no features separate MS patients and healthy individuals. This is also represented by the indistinguishability of the sample subgroups (Figures 4 and 5).

Since multivariate analysis showed no separation between subgroups, univariate analysis was pursued to confirm that no features drive variation. For all two-way analysis, subgroups were paired as follows: control and untreated, control and patients treated with Copaxone, and control and patients treated with Interferon. Effect plots generated by the differential abundance tool, ALDEx2, complimented the biplot results showing no features to be significantly different between groups. In fact, more variation within subgroups than between is evident [17, 19] (Figures 6, S9, S10).

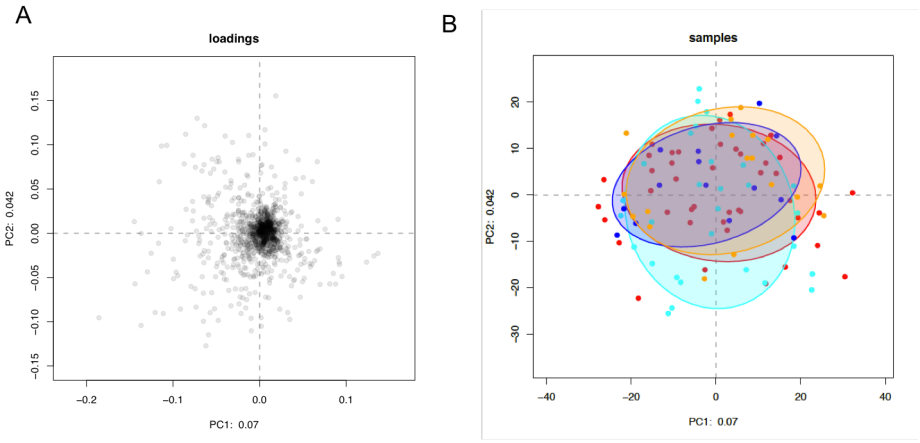

Figure 4: **Compositional biplot comparing control and untreated groups processed using DADA2.** A. A loadings biplot. Each dot represents a loading (feature) identified by the pipelines. Distinctive loadings drive variation between subgroups in the coloured biplot. B. A coloured biplot, which visually explains compositional differences between subgroups. Control group is represented by red, untreated by cyan, patients treated with Copaxone by blue, and patients treated with Interferon by orange. Principal components 1 and 2 serve as the x- and y-axis, respectively and indicate axes with the most variance explained.

Comparison of data processing protocols used in this study to those used by Jangi et al's research team showed an inconsistency in the data analysis tools used (Table 3). While this study uses ALDEx2, Jangi et al ran DESeq2, which differs in data normalization, dealing with zero counts, and mean-variance assumptions [13]. To reduce confounding variables, count tables derived from both pipelines were processed with ALDEx2 and DESeq2. The standard DESeq2 protocol was followed with default parameters as this was what was expected to be what was employed by the researchers [20]. Results found significant increases and decreases in abundances for tables generated by both pipelines (Tables S4, S5); however, none were observed by ALDEx2 (Tables S6, S7). Upon evaluation of MA plots, DESeq2 plots were abnormal and unreadable indicative of inappropriate data normalization methods (Figure 7A). Features clustered around zero, as shown in the ALDEx2-generated MA plots, represent normal data with low variation between groups and

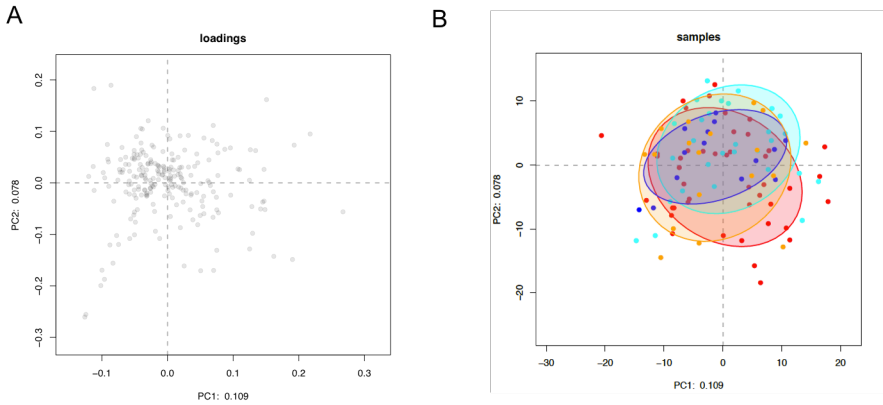

Figure 5: **Compositional biplot comparing control and untreated groups processed using DADA2.** A. A loadings biplot. Each dot represents a loading (feature) identified by the pipelines. Distinctive loadings drive variation between subgroups in the coloured biplot. B. A coloured biplot, which visually explains compositional differences between subgroups. Control group is represented by red, untreated by cyan, patients treated with Copaxone by blue, and patients treated with Interferon by orange. Principal components 1 and 2 serve as the x- and y-axis, respectively and indicate axes with the most variance explained.

high variation within (Figure 7B).

Method Component	Present Study	Publication Methods
Processing reads	ASV-based approach	OTU-based approach
Data analysis tool	ALDEx2	DESeq2
Multiple test correction	Benjamini-Hochberg = 0.1	Benjamini-Hochberg = 0.1
Significance test	Wilcoxon rank sum	Wilcoxon rank sum

Table 3: **Comparing data processing protocols used in the present study and by Jangi et al.**

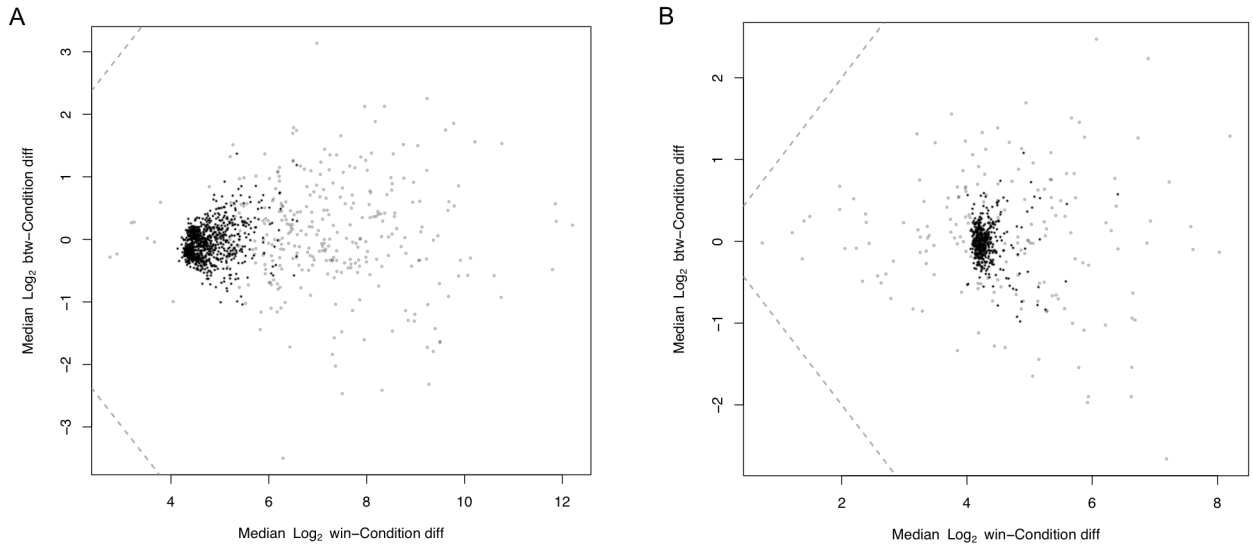

Figure 6: **A. Effect plot comparing control and untreated groups processed using DADA2. B. Effect plot comparing control and untreated groups processed using mothur.** Each point on the plot represents one feature. Effect is measured by taking a ratio of the difference between groups and the difference within. The diagonal lines represent the line of equivalence (effect is 1:1) and any points preceding the line represents a feature that has more variation between groups than within.

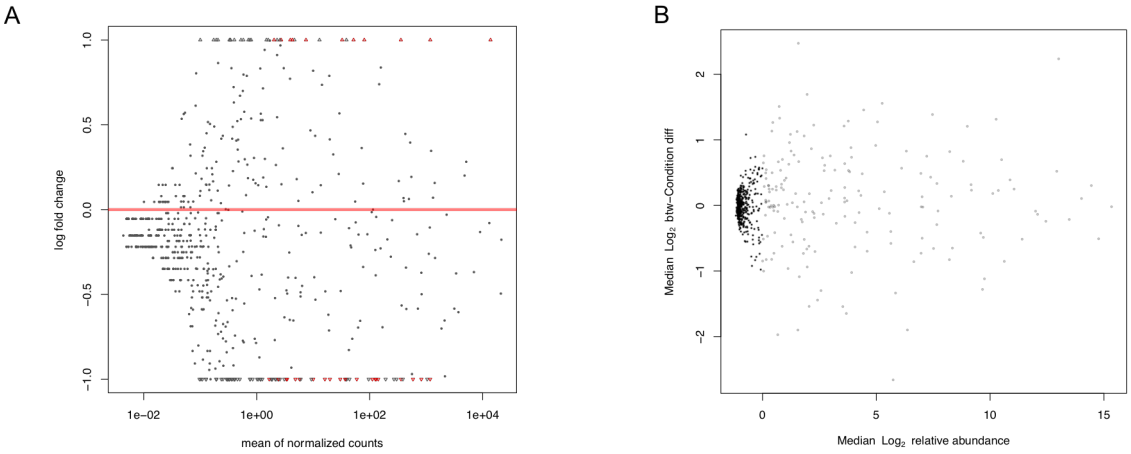

Figure 7: **A. MA plot comparing control and untreated groups processed using DADA2. B. MA plot comparing control and untreated groups processed using mothur.** MA plots plot the difference between measurements of two samples or subgroups."M" represents the log ratio of the difference between samples on the x-axis and "A" displays the mean average along the y-axis.

8 Discussion and Limitations

8.1 Positive Control

Although the QIIME output had almost double the number of features as DADA2, filtR results indicated that only a small proportion are technically introduced false positives. The remaining difference may be attributed to the clustering method employed by QIIME. QIIME is an open reference pipeline that references sequences to the external database, GreenGene. Rare variants that aren't found in the database form *de novo* clusters and are referenced to an alternative database. To cluster reads, Tourlousse et al utilized the tool USEARCH, which applies centroid clustering and references the Broad Microbiome Utilities' 16S Gold database [21]. Centroid clustering arbitrarily selects a feature to become the center of a cluster and sequentially aligns other sequences to evaluate their inclusion. Sequences are included in the cluster if they meet the chosen percent similarity to the center or average of the sequences within the cluster. The drawback of this method lies in the order of which sequences are selected as the centroid or added to the cluster. As shown in Figure 8, if a sequence highly dissimilar to the remainder of a dataset is chosen as the center, many more clusters will be generated. This may explain the vast difference in the number of OTUs generated by QIIME compared to the number of ASVs generated by DADA2.

Since there were only 15 microbes in the mock communities and each feature corresponds to a taxonomic assignment, it was expected that approximately 15 features would be detected. Nonetheless, both DADA2 and QIIME grossly overestimated the diversity in the communities. Throughout the duration of this study, updated versions of the pipelines were released and improvements are ongoing. These fixes along with more stringent filtering methods may help collapse features that represent the same microbe to reduce false positive results. Another study comparing sequence processing pipelines found that DADA2 identified more rare ASVs in a host-associated biome but at the cost of false positives [21]. The same study also found that OTU pipelines consistently identified more features than DADA2, likely as a result of the clustering method while a separate study attributed the reported

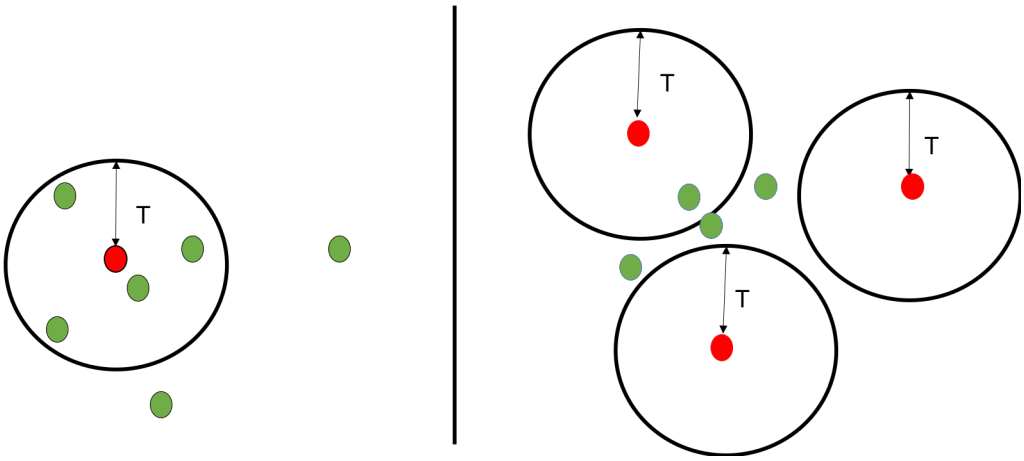

Figure 8: **Visualization of centroid clustering.** Each circle represents a sequence and proximity of sequences represents sequence similarity. Black circles represent resulting clusters with a defined radius threshold (T) signified by a black arrow. Left: clustering initiated by a sequence highly similar to the remainder of the dataset. Right: clustering initiated by sequences highly dissimilar to the remainder of the dataset.

inflated diversities by QIIME to inadequate error filtering.

8.2 Negative Control

The team of scientists that created the synthetic 16S rRNA variable regions intended for the sequences to resemble bacteria, therefore it is not surprising that 142 of 143 spike-in ASVs were identified as members of the bacteria kingdom when referenced to the SILVA database [16]. A class assignment is an unusually high resolution for synthetic sequences with no reported identity and can be explained by differences in the taxonomic resolution and sequences housed of reference databases. To check the identity of the synthetic sequences, Tourlousse et al referenced NCBI's nt, est and est_human nucleotide sequence databases both before and after incorporation into 16S rRNA gene constructs. Although the NCBI databases contain more bacteria sequences, SILVA is regularly updated and quality checks sequences [22, 23]. Consequently, unexpected classifications of the synthetic regions may have been a result of referencing a more recent database since the time of original publication.

8.3 Test

The discrepancy between gut dysbiosis reported by Jangi et al and this study's results may be attributed to numerous factors. For one, low power of the study represented by small cohorts is evident (Table S8). Although the same hypothesis tests were used when generating MA plots to allow for a fair comparison, having a low powered study can still cause false results [24]. Abnormal MA plots generated from both mothur and DADA2 pipelines in DESeq2 are indicative of unnormalized data. The normalization method employed by this tool is encoded in the "DESeq" function without room to modify. If custom scripts were made to normalize the processed reads by Jangi et al, they were not included in the published study. Conversely, if MA plots were overlooked by the researchers the interpretations reported may be inaccurate. Moreover, while the analytical tool DESeq2 found significant differences in feature abundances for both mothur and DADA2 count tables, ALDEx2 found none. The consistency between DADA2 and mothur analysis results ultimately suggests that the discrepancy in the published article lies in the analytical tool, DESeq2, as opposed to in the mothur pipeline.

The Benjamini-hochberg correction attempts to mitigate the uprising of false positives in small sample sizes by calculating the fraction of false positives given a p-value and population size and adjusting according to an arbitrary threshold of "acceptable" false positives. A more appropriate tool that evades using the p-value significance testing entirely is the ALDEx2 effect plot [25]. Effect plots evaluate relative abundances by considering the ratio of differences between and within groups. As a result, they do not require a large sample size to accurately represent differences within and between datasets [18]. For this study, we can depend more heavily on ALDEx2 results because of its ability to eliminate low powered biases by generating effect plots based on relative abundances and provide evidence of correct data normalization. With that in mind and in contrast to results reported by Jangi et al, no significant difference was observed between MS patients and healthy individuals.

Another factor that influenced the reproducibility of the published result is the absence of detailed reporting of methods by the researchers. Custom python scripts used to filter sequences, exact primer sequences used, and the SILVA database ref-

erence file were omitted, all which impact crucial steps when running the mothur pipeline. This number of uncertain variables would undoubtedly lead to incomparable results thus replicating the employed protocol *ab initio* was abandoned.

Despite the apparent influence of analytical tools on interpreting results, it should be noted that mothur still produced technical variants detectable by filtR. Comparatively, DADA2 consistently detected different numbers of features than the OTU-based pipelines used in the test and control with no technical variants. This seems to suggest that DADA2's higher resolution permits a more thorough and robust distinction of true biological variants as well as effectively removes erroneous reads.

9 Conclusions

The ASV-based standard operative procedure is more reliable than OTU clustering because it is more durable against technical errors and avoids variable results produced by clustering methods. Yet, over-estimation of populations remains an issue for both processing methods.

A lack of detailed reporting of methodology lead to irreproducible results for the MS study. Consequently, this study could not confidently conclude a difference in gut microbiomes in MS patients relative to healthy individuals. Discrepancies in results may be attributed to the way data is analysed for interpretation after being run through pipelines since both mothur and DADA2 pipelines generated consistent results regardless of the tool used.

10 Further Directions

To establish confident conclusions on the role of the gut microbiome in multiple sclerosis, new datasets should be obtained with a larger number of samples to increase the power of the study. Currently, pipelines are being improved and latest versions should be used to process reads. Moreover, more reliable results can be ensured by using an ASV pipeline. Finally, raw data should continue to be reported and open access; however, much like adapted methods are reported in publications, revisions to computational scripts should be openly documented as well.

11 Acknowledgements

I'd like to thank the Gloor Lab for all their help and support throughout the summer. Specifically, I'd like to acknowledge Dr. Greg Gloor, Jean Macklaim, Ben Joris, and Dan Giguere for their patience with my question-asking and aid in my slow ascent over the learning curve that is coding in R. This lab has fostered a positive work environment that has considerably enhanced my knowledge and made each day in the lab enjoyable. I'd like to give a special thanks for Dr. Ball and Dr. Brandl for their feedback and encouragement in writing as well as managing the course. Finally, a big thank you to Ryan Szukalo for being my on-call Latex advisor and handyman, without whom this PDF would not be possible.

References

[1] Laura M. Cox Ning Li Felipe von Glehn Raymond Yan Bonny Patel Maria Antonietta Mazzola Shirong Liu Bonnie L. Glanz Sandra Cook Stephanie Tankou Fiona Stuart Kirsy Melo Parham Nejad Kathleen Smith Begum D. Topcuolu James Holden Pia Kivisakk Tanuja Chitnis Philip L. De Jager Francisco J. Quintana Gerge K. Gerber Lynn Bry Sushrut Jangi, Roopali Ganhi and Howard L. Weiner. Alterations of the human gut microbiome in multiple sclerosis. *Nature Communications*, 7(12015), 2016.

[2] Ron Milo Ron Sender, Shai Fuchs. Revised estimates for the number of human and bacteria cells in the body. *PLOS Biology*, 2016.

[3] TD Luckey. Introduction to intestinal microecology. *The American Journal of Clinical Nutrition*, 12:1292–4, 1972.

[4] DC Savage. Microbial ecology of the gastrointestinal tract. *Annual Review Microbiology*, 31:107–33, 1977.

[5] Stephen M. Collins. A role for the gut microbiota in ibs. *Nature Reviews Gastroenterology amp; Heptaology*, 11:497–505, 2014.

[6] William B. Whitman. Bergey's manual of systematic bacteriology. *Springer*, 4, 2010.

[7] Howard Ochman Victor Kunin, Anna Engelbrektson and Philip Hugenholtz. Wrinkles in the rare biosphere: pyrosequencing errors can lead to artifical inflation of diversity estimates. *Environmental Microbiology*, 12(1):118–123, 2010.

[8] Paul Coupland Thomas D Otto Simon R Harris Thomas R Connor Anna Bertoni Harold P Swerdlow Michael A Quail, Miriam Smith and Yong Gu. A tale of three next generation sequencing platforms: comparison of ion torrent, pacific biosciences and illumina miseq sequencers. *BMC Genomics*, 13(341):1471–2164, 2012.

[9] William A Walters Donna Berg-Lyons James Huntley Noah Fierer Sarah M Owens Jason Betley Louise Fraser Markus Bauer Niall Gormley Jack A Gilbert

Geoff Smith J Gregory Caporaso, Christian L Lauber and Rob Knight. Ultra-high-throughput microbial community analysis on the illumina hiseq and miseq platforms. *The ISME Journal*, 6:1621–1624, 2012.

[10] Paul J McMurdie Benjamin J Callahan and Susan P Holmes. Exact sequence variants should replace operational taxonomic units in marker-gene data analysis. *Nature*, 11:2639–2643, 2017.

[11] Rafael A Irizarry Kasper D Hansen, Zhijin Wu and Jeffery T Leek. Sequencing technology does not eliminate biological variability. *Nature Biotechnology*, 29(7):572–573, 2011.

[12] 2018.

[13] Jean M Macklaim Thomas A McMurrough David R Edgell Andrew D Fernandes, Jennifer NS Reid and Gregory B Gloor. Unifying the analysis of high-throughput sequencing datasets: characterizing rna-seq 16s rrna gene sequencing and selective growth experiments by compositional data analysis. *Microbiome*, 2(15):1–13, 2014.

[14] John F. Cryan Dinan and Timothy G. Mind-altering microorganisms: the impact of the gut microbiota on brain and behaviour. *Nature Reviews Neuroscience*, 13:701–712, 2012.

[15] Susan Garges The NIH HMP Working Group, Jane Peterson. The nih human microbiome project. 19:2317–2323, 2009.

[16] Akiko Ohashi Satoko Matsukura Naohiro Noda Dieter M. Tourlousse, Satowa Yoshiike and Yuji Sekiguchi. Synthetic spike-in standards for high-throughput 16s rrna gene amplicon sequencing. *Nucleic Acids Research*, 45(4), 2017.

[17] Gregor Reid Gregory B. Gloor. Compositional analysis: a valid approach to analyze microbiome high-throughput sequencing data. *Canadian Journal of Microbiology*, 26(5):322–329, 2016.

[18] Vera Pawlowsky-Glahn Gregory B. Gloor, Jean M. Macklaim and Juan J. Eqozcue. Microbiome datasets are compositional: And this is not optional. *Frontiers in Microbiology*, 8(2224), 2017.

[19] Jean M Macklaim-Gregor Reid Gregory B Gloor Andrew D Fernandes, TG Linn. Anova-like differential gene expression analysis of single-organism and meta-rna-seq. *PLoS ONE*, 8(7), 2013.

[20] Simon Anders Wolfgang Huber, Michael I Love. Moderated estimation of fold change and dispersion for rna-seq data with deseq2. *bioRxiv prepint*, 2014.

[21] Andre M. Comeau Morgan G.I. Langille Jacob T. Nearing, Gavin M. Douglas. Denoising the denoisers: an independent evaluation of microbiome sequence error-correction approaches. *PeerJ*, 2018.

[22] Monica Santamaria Bruno Fosso Arianna Consiglio Giorgio De Caro Giorgio Grillo Flavio Licciulli Sabino Liuni Marinella Marzano Daniel Alonso-Alemany Gabriel Valiente Graziano Pesole. Reference databases for taxonomic assignment in metagenomics. *Briefings in Bioinformatics*, 13(6):682–695, 2012.

[23] Christian Quast Elmar Pruesse Pelin Yilmaz Jan Gerken Timmy Schweer Pablo Yarza Jörg Peplies Frank Oliver Glöckner. The silva ribosomal rna gene database project: improved data processing and web-based tools. *Nucleic Acids Research*, 41(D1):D590–D596, 2013.

[24] Sarah L Vowler Gordon B Drummon Lewis G Halsey, Douglas Curran-Everett. The fickle p value generates irreproducible results. *Nature Methods*, 12(3):179–185, 2015.

[25] Daniel Kuang David Thissen, Lynne Steinberg. Quick and easy implementation of the benjamini-hochberg procedure for controlling the false positive rate in multiple comparisons. *Sage Journal of Educational and Behavioral Statistics*, 2002.

12 Supplementary Data

12.1 Figures and Tables

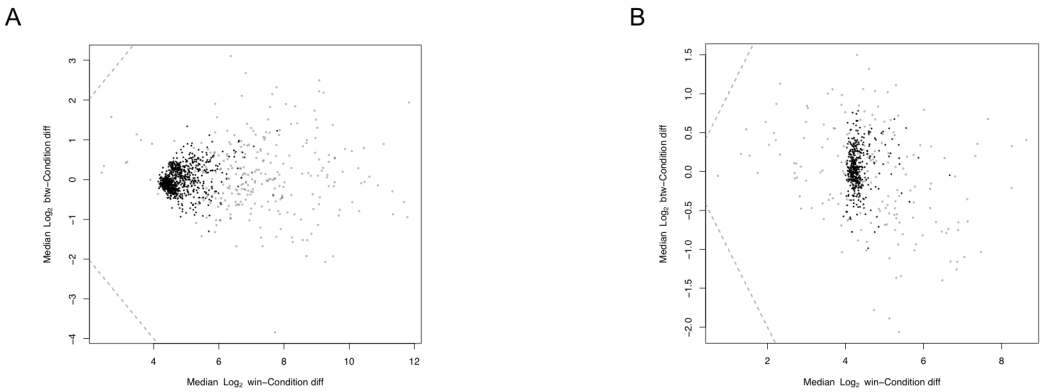

Figure S9: **A. Effect plot comparing control and patients treated with Interferon processed using DADA2. B. Effect plot comparing control and patients treated with Interferon processed using mothur**. Each point on the plot represents one feature. Effect is measured by taking a ratio of the difference between groups and the difference within. The diagonal lines represent the line of equivalence (effect is 1:1) and any points preceding the line represents a feature that has more variation between groups than within.

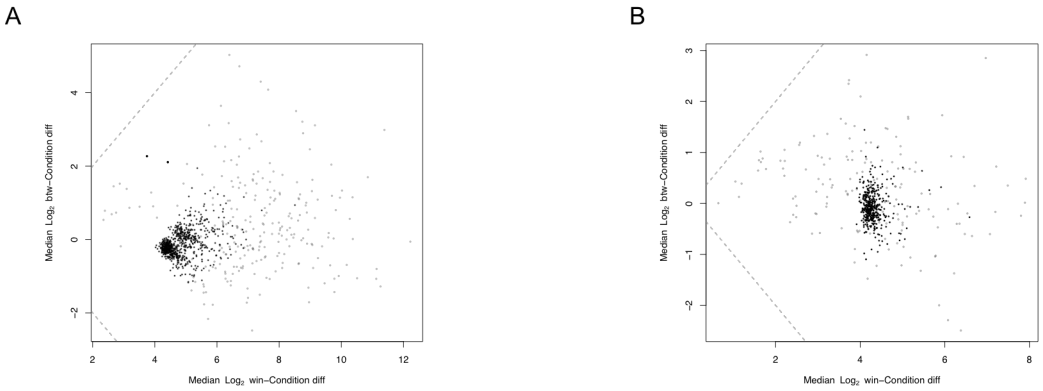

Figure S10: **A. Effect plot comparing control and patients treated with Copaxone processed using DADA2. B. Effect plot comparing control and patients treated with Interferon processed using mothur**. Each point on the plot represents one feature. Effect is measured by taking a ratio of the difference between groups and the difference within. The diagonal lines represent the line of equivalence (effect is 1:1) and any points preceding the line represents a feature that has more variation between groups than within.

Significant changes in abundance	control-untreated	control-copaxone	control-interferon
Increase	6	6	6
Decrease	20	48	32

Table S4: **Number of features found to increase or decrease significantly between subgroups using DADA2 followed by DESeq2.**

Significant changes in abundance	control-untreated	control-copaxone	control-interferon
Increase	6	6	6
Decrease	31	25	15

Table S5: **Number of features found to increase or decrease significantly between subgroups using mothur followed by DESeq2.**

Significant changes in abundance	control-untreated	control-copaxone	control-interferon
Increase	0	0	0
Decrease	0	0	0

Table S6: **Number of features found to increase or decrease significantly between subgroups using DADA2 followed by ALDEx2.** Using Wilcoxon rank-sum test.

Significant changes in abundance	control-untreated	control-copaxone	control-interferon
Increase	0	0	0
Decrease	0	0	0

Table S7: **Number of features found to increase or decrease significantly between subgroups using mothur followed by ALDEx2.** Using Wilcoxon rank-sum test.

Subgroup	Number of Samples
Control	44
Untreated	29
Copaxone	14
Interferon	18

Table S8: **Number of fecal samples in each cohort.** Subgroups were chosen by Jangi et al and sample information was retrieved from Project SRP075039 on The European Bioinformatics Institute. Copaxone and Interferon subgroups refer to samples from individuals diagnosed with MS and treated with Copaxone and Interferon, respectively.

12.2 Computational Methods

```
# Control dada2 workflow
# set up ----
library(dada2)

path <- "/Volumes/data/khobbs3/reads"
taxpath<-"/Volumes/data/annotationDB/dada2/silva_nr_v123_train_set.fa.gz" #agrajag
reads<-"demultiplex_reads"

# list files
list.files(path)

# sort fwd and rvs reads
fnFs <- sort(list.files(path, pattern="_1.fastq", full.names=TRUE))
fnRs <- sort(list.files(path, pattern="_2.fastq", full.names=TRUE))

# get file names only (remove path)
sample.names <- sapply(strsplit(basename(fnFs), "_"), `[`, 1)

# check for duplications
any(duplicated(sample.names))

# check read quality of a random subset of 4 samples ----
ids<-round(runif(4,1,length(sample.names)))

setwd("/Users/kt/Documents/Documents/Undergrad/4/4483E/control/dada2-output")

pdf("qualprofiles.pdf")
plotQualityProfile(fnFs[ids])
plotQualityProfile(fnRs[ids])
dev.off()

# filter reads based on QC ----
filtFs <- paste0(reads, "/", sample.names, "-F-filt.fastq")
filtRs <- paste0(reads, "/", sample.names, "-R-filt.fastq")


out<-filterAndTrim(fnFs, filtFs, fnRs, filtRs,
                   truncLen=c(200,150),
                   truncQ=2,
                   maxN=0,
                   maxEE=c(2,2),
                   compress=TRUE, verbose=TRUE, multithread=TRUE)

write.table(out, file="after_filter.txt", sep="\t", col.names=NA, quote=F)

# learn error rates ----
errF <- learnErrors(filtFs, multithread=TRUE, randomize=TRUE)
errR <- learnErrors(filtRs, multithread=TRUE, randomize=TRUE)

pdf("err.pdf")
plotErrors(errF, nominalQ=TRUE)
plotErrors(errR, nominalQ=TRUE)
dev.off()

# dereplication (combine identicals to make unique sequences) ----
derepFs <- derepFastq(filtFs, verbose=TRUE)
derepRs <- derepFastq(filtRs, verbose=TRUE)

# Name the derep-class objects by the sample names
names(derepFs) <- sample.names
names(derepRs) <- sample.names
```

```
# infer SVs ----
dadaFs <- dada(derepFs, err=errF, multithread=TRUE)
dadaRs <- dada(derepRs, err=errR, multithread=TRUE)

mergers <- mergePairs(dadaFs, derepFs, dadaRs, derepRs, verbose=TRUE)

# make sequence table
seqtab <- makeSequenceTable(mergers)
table(nchar(getSequences(seqtab)))

# remove chimeras
seqtab.nochim <- removeBimeraDenovo(seqtab, method="pooled", verbose=TRUE, multithread=TRUE)
dim(seqtab.nochim)

write.table(seqtab.nochim, file="temp_dada2_nochim.txt", sep="\t", col.names=NA, quote=F)

# How many reads made it through pipeline ----
getN <- function(x) sum(getUniques(x))
track <- cbind(out, sapply(dadaFs, getN), sapply(mergers, getN), rowSums(seqtab), rowSums(seqtab
    .nochim))
colnames(track) <- c("input", "filtered", "denoised", "merged", "tabled", "nonchim")
rownames(track) <- sample.names
write.table(track, file="control-readsout.txt", sep="\t", col.names=NA, quote=F)

# Assign taxonomy ----
taxa <- assignTaxonomy(seqtab.nochim, taxpath, multithread=TRUE)
colnames(taxa) <- c("Kingdom", "Phylum", "Class", "Order", "Family", "Genus")

#merge columns 1 to 6 to get the full taxonomy (to genus)
un.tax <- unname(taxa)
tax.vector <- apply(un.tax, 1, function(x){paste(x[1:6], collapse=":")})

# add new row of taxonomy names then transpose so that its a col
seqtab.nochim.tax<-rbind(seqtab.nochim, tax.vector)
t.seqtab.nochim.tax<-t(seqtab.nochim.tax)

# replace SV rownames with arbitrary numbers
sv.seqs<-rownames(t.seqtab.nochim.tax)
sv.num<-paste("SV", seq(from = 0, to = nrow(t.seqtab.nochim.tax)-1), sep="_")

rownames(t.seqtab.nochim.tax)<-sv.num

write.table(t.seqtab.nochim.tax, file="dada2_nochim_tax.txt", sep="\t", col.names=NA, quote=F)
write.table(sv.seqs, file="sv_seqs.txt", sep="\t", row.names=sv.num, col.names=F,  quote=F)



#Part Two: ALDEx2 for Differential Expression Analysis ----
library("ALDEx2")
setwd('/Users/kt/Documents/Documents/Undergrad/4/4483E/test/CoDa-Output/Part_2_-_ALDEx/')

# must run "dada2-biplots_updated.R" first to get dataframe "ddf"

# cvu = controls v untreated ====
# Step 1: create different pairwise condition vectors since ttests can only run for 2
    populations
#(must have subsetted data first):
cvu <- c(rep("controls", length(controls)),
         rep("untreated", length(untreated))
)

# Step 2: create new aldex df containing only controls and untreated columns
# run codapt1-biplots.R first to get ddf
aldex.cvu <- ddf[,c(controls, untreated)]
```

```
colnames(aldex.cvu)
dim(aldex.cvu)

#aldex:
# removed rows with sums == 0
# computed centre with all features - converts instances using centred log-ratio transform
# generates Monte Carlo samples of Dirichlet distribution for each sample

# Step 3: get the clr values
cvu.x <- aldex.clr(aldex.cvu, cvu, mc.samples=128, verbose=TRUE)

# Step 4: perform t-tests: Welches, Wilcoxon, and Benjamini-Hochberg multiple test correction
cvu.x.tt <- aldex.ttest(cvu.x, cvu, paired.test=FALSE)

# Step 5: estimate effect size
cvu.x.effect <- aldex.effect(cvu.x, cvu, include.sample.summary=TRUE, verbose=TRUE)
dim(cvu.x.effect)

# Step 6: merge data
cvu.x.all <- data.frame(cvu.x.tt, cvu.x.effect)

# Step 7: create table of merged data
write.table(cvu.x.all, file="cont_untreat_aldex_ttest.txt")

# Step 8: see plots
pdf("controls_untreated.pdf")
aldex.plot(cvu.x.all, type="MA", test="welch")
dev.off()

# Step 9: get features passing a specified significance level
sig <- which(cvu.x.all$we.eBH < 0.05)

# Step 10: get significant points that only point in the positive direction
psig <- which(cvu.x.all$we.eBH < 0.05 & cvu.x.all$diff.btw > 0)

# Step 11:
#plot diff btwn vs diff within
#plot significant points in a different color
#add the effect=1 and -1 lines
pdf("with_text-control_untreated_btw-win.pdf")
aldex.plot(cvu.x.all, type="MW", test="welch")
text(cvu.x.all$diff.win[psig], cvu.x.all$diff.btw[psig], labels=row.names(common3), cex= 0.5,
    pos=3, col="blue") #only positive sig points
text(cvu.x.all$diff.win[psig], cvu.x.all$diff.btw[psig], labels=row.names(common4), cex= 0.5,
    pos=3, col="red") #meaning only points where values >mean of sample
dev.off()

pdf("with_sig_points_-_control_untreated_btw-win.pdf")
aldex.plot(cvu.x.all, type="MW", test="welch")
text(cvu.x.all$diff.win, cvu.x.all$diff.btw, labels=row.names(common3), cex= 0.5, pos=3, col="
    blue") #all points
text(cvu.x.all$diff.win, cvu.x.all$diff.btw, labels=row.names(common4), cex= 0.5, pos=3, col="
    red")
dev.off()

# Step 12: make a table of significant taxa

cvu.table <- cvu.x.all[sig, c(4:7, 10,11)]
caption = "Table_of_significant_taxa"
digits=3
label="sig.table"
align=c("l",rep("r",6) )

write.table(cvu.table, file="cvu.table_sig_taxa.txt")
```

```
# repeat for other pairwise comparisons (annotated as subset versus subset - svs):

# Step 13: note how many SVs with zeros were removed
cvu.rmvd <- rownames(aldex.cvu)[rowSums(aldex.cvu) == 0]
cvu.rmvd #98

# cvcop = controls v copaxone ====
cvcop <- c(rep("controls", length(controls)),
           rep("copaxone", length(copaxone))
)
aldex.cvcop <- ddf[,c(controls, copaxone)]
cvcop.x <- aldex.clr(aldex.cvcop, cvcop, mc.samples=128, verbose=TRUE)
cvcop.x.tt <- aldex.ttest(cvcop.x, cvcop, paired.test=FALSE)
cvcop.x.effect <- aldex.effect(cvcop.x, cvcop, include.sample.summary=TRUE, verbose=TRUE)
cvcop.x.all <- data.frame(cvcop.x.tt, cvcop.x.effect)
write.table(cvcop.x.all, file="cont_copax_aldex_ttest.txt", sep="\t", quote=F, col.names=NA)

pdf("controls_copaxone.pdf")
aldex.plot(cvcop.x.all, type="MA", test="welch")
dev.off()

sig <- which(cvcop.x.all$we.eBH < 0.05)
sig
psig <- which(cvcop.x.all$we.eBH < 0.05 & cvcop.x.all$diff.btw > 0)
psig

pdf("control_copaxone_btw-win.pdf")
aldex.plot(cvcop.x.all, type="MW", test="welch")
dev.off()

cvcop.rmvd <- rownames(aldex.cvcop)[rowSums(aldex.cvcop) == 0]
dim(cvcop.rmvd)
length(cvcop.rmvd)



#cvint = controls v interferon ====
cvint <- c(rep("controls", length(controls)),
           rep("interferon", length(interferon))
)
aldex.cvint <- ddf[,c(controls, interferon)]
cvint.x <- aldex.clr(aldex.cvint, cvint, mc.samples=128, verbose=TRUE)
cvint.x.tt <- aldex.ttest(cvint.x, cvint, paired.test=FALSE)
cvint.x.effect <- aldex.effect(cvint.x, cvint, include.sample.summary=TRUE, verbose=TRUE)
cvint.x.all <- data.frame(cvint.x.tt, cvint.x.effect)
write.table(cvint.x.all, file="cont_interf_aldex_ttest.txt", sep="\t", quote=F, col.names=NA)

pdf("controls_interferon.pdf")
aldex.plot(cvint.x.all, type="MA", test="welch")
dev.off()

sig <- which(cvint.x.all$we.eBH < 0.05)
sig
psig <- which(cvint.x.all$we.eBH < 0.05 & cvint.x.all$diff.btw > 0)
psig

pdf("control_interferon_btw-win.pdf")
aldex.plot(cvint.x.all, type="MW", test="welch")
dev.off()

cvint.rmvd <- rownames(aldex.cvint)[rowSums(aldex.cvint) == 0]
length(cvint.rmvd)
```

```
# CoDa Methods
#Set-Up: ----
library("zCompositions")
library("devtools")

#Part 1: PCA ----
setwd("/Users/kt/Documents/Documents/Undergrad/4/4483E/test/Dada2-Output")
d <- read.table("dada2_nochim_tax.txt", header=T, sep="\t", stringsAsFactors=F, quote = "",
    check.names=F, row.names=1, comment.char="")
dd<-d[,1:ncol(d)-1]
rownames(dd)<-paste(d$tax.vector, rownames(d), sep=":")

## Note: did not pull a random sample from dataset
    #dd <-d[,1:ncol(d)-1] #cut table to remove taxonomy column...no need to do this if you
        exclude this col in filtering
sum(d == 0) #count the number of zeros
sum(d != 0) #number of non-zeros

ddf <- data.frame(dd[which(apply(dd[,1:ncol(dd)-1], 1, function(x){sum(x)}) > ncol(dd)), ],
    check.names=F) # this removes SVs with too many zeros?
ddf.czm <- cmultRepl(t(ddf), label=0, method="CZM") #replace count zeros and transform the data
     by taking a log

# Samples must be ROWs and features/OTUs as COLUMNS
head(ddf.czm) #check

ddf.clr <- t(apply(ddf.czm, 1, function(x){log(x) - mean(log(x))}))
head(ddf.clr) #check

ddf.pcx <- prcomp(ddf.clr)
  #principal components analysis (PCA), output class = prcomp
ddf.mvar <- sum(ddf.pcx$sdev^2) #summing total variance in each column(?) of PCA
PC1 <- paste("PC1:", round(sum(ddf.pcx$sdev[1]^2)/ddf.mvar, 3)) #joining variables to create
    axis labels
PC2 <- paste("PC2:", round(sum(ddf.pcx$sdev[2]^2)/ddf.mvar, 3))
PC1 # to look at PC1 variable constant
PC2


### Subsetting data:
controls <- c("SRR3501908", "SRR3501909", "SRR3501910", "SRR3501911", "SRR3501912", "SRR3501913"
    , "SRR3501914", "SRR3501915", "SRR3501916", "SRR3501925", "SRR3501936", "SRR3501945", "
    SRR3501946", "SRR3501947", "SRR3501948", "SRR3501949", "SRR3501950", "SRR3501951", "
    SRR3501952", "SRR3501953", "SRR3501954", "SRR3501955", "SRR3501956", "SRR3501957", "
    SRR3501958", "SRR3501959", "SRR3501969", "SRR3501973", "SRR3501974", "SRR3501975", "
    SRR3501976", "SRR3501977", "SRR3501978", "SRR3501979", "SRR3501980", "SRR3501981", "
    SRR3501982", "SRR3501983", "SRR3501984", "SRR3501985", "SRR3501986", "SRR3501987", "
    SRR3501991", "SRR3502002")
untreated <- c("SRR3501917", "SRR3501918", "SRR3501921", "SRR3501923", "SRR3501924", "SRR3501932
    ", "SRR3501933", "SRR3501944", "SRR3501960", "SRR3501961", "SRR3501962", "SRR3501963", "
    SRR3501964", "SRR3501965", "SRR3501966", "SRR3501967", "SRR3501988", "SRR3501989", "
    SRR3501990", "SRR3501992", "SRR3501993", "SRR3501994", "SRR3501995", "SRR3501996", "
    SRR3501997", "SRR3501998", "SRR3501999", "SRR3502000", "SRR3502010")
copaxone <- c("SRR3501919", "SRR3501927", "SRR3501930", "SRR3501931", "SRR3501935", "SRR3501940"
    , "SRR3501942", "SRR3501943", "SRR3501970", "SRR3501971", "SRR3502001", "SRR3502003", "
    SRR3502005", "SRR3502007")
interferon <- c("SRR3501920", "SRR3501922", "SRR3501926", "SRR3501928", "SRR3501929", "
    SRR3501934", "SRR3501937", "SRR3501938", "SRR3501939", "SRR3501941", "SRR3501968", "
    SRR3501972", "SRR3502004", "SRR3502006", "SRR3502008", "SRR3502009", "SRR3502011", "
    SRR3502012")
df<-ddf[,c(controls, untreated, copaxone, interferon)] # this stitches together the subsets into
     a data frame called df.
```

```
# make screeplot - plots variances against number of prcomps
screeplot(ddf.pcx, type = "barplot",
          col = "black" )


# CoDaSeq Biplot Functions----
library(CoDaSeq)

group.col = c("red", "cyan", "blue", "orange")
grps <- list(controls, untreated, copaxone, interferon)
?codaSeq.PCAplot

codaSeq.PCAplot(ddf.pcx, plot.groups=TRUE, plot.circles=TRUE, grp.col=group.col, grp=grps)
legend(-40,-15, col=group.col, legend=c("Cont", "Unt", "Cop","Int"), pch=19)

pdf("Dada2_MS_Biplots.pdf")
par(mfrow=c(1,2))
codaSeq.PCAplot(ddf.pcx, plot.groups=TRUE, plot.circles=TRUE, grp.col=group.col, grp=grps)
legend(-40,-15, col=group.col, legend=c("Cont", "Unt", "Cop","Int"), pch=19)
dev.off()




# DESeq2 Workflow

#source("http://bioconductor.org/biocLite.R")
#biocLite("DESeq2")
library( "DESeq2" )

# 2.4.3 build DESeqDataSet from a counts table ----
## import dada2 counts table:
setwd("/Users/kt/Documents/Documents/Undergrad/4/4483E/test/Dada2-Output/")
counts <- read.table("dada2_nochim_tax.txt", header=T, sep="\t", stringsAsFactors=F, quote = "",
      check.names=F, row.names=1, comment.char="")
counts.d <- counts[,1:ncol(counts)-1]
dim(counts.d)

## subsetting data:
controls <- c("SRR3501908", "SRR3501909", "SRR3501910", "SRR3501911", "SRR3501912", "SRR3501913"
     , "SRR3501914", "SRR3501915", "SRR3501916", "SRR3501925", "SRR3501936", "SRR3501945", "
     SRR3501946", "SRR3501947", "SRR3501948", "SRR3501949", "SRR3501950", "SRR3501951", "
     SRR3501952", "SRR3501953", "SRR3501954", "SRR3501955", "SRR3501956", "SRR3501957", "
     SRR3501958", "SRR3501959", "SRR3501969", "SRR3501973", "SRR3501974", "SRR3501975", "
     SRR3501976", "SRR3501977", "SRR3501978", "SRR3501979", "SRR3501980", "SRR3501981", "
     SRR3501982", "SRR3501983", "SRR3501984", "SRR3501985", "SRR3501986", "SRR3501987", "
     SRR3501991", "SRR3502002")
untreated <- c("SRR3501917", "SRR3501918", "SRR3501921", "SRR3501923", "SRR3501924", "SRR3501932
     ", "SRR3501933", "SRR3501944", "SRR3501960", "SRR3501961", "SRR3501962", "SRR3501963", "
     SRR3501964", "SRR3501965", "SRR3501966", "SRR3501967", "SRR3501988", "SRR3501989", "
     SRR3501990", "SRR3501992", "SRR3501993", "SRR3501994", "SRR3501995", "SRR3501996", "
     SRR3501997", "SRR3501998", "SRR3501999", "SRR3502000", "SRR3502010")
copaxone <- c("SRR3501919", "SRR3501927", "SRR3501930", "SRR3501931", "SRR3501935", "SRR3501940"
     , "SRR3501942", "SRR3501943", "SRR3501970", "SRR3501971", "SRR3502001", "SRR3502003", "
     SRR3502005", "SRR3502007")
interferon <- c("SRR3501920", "SRR3501922", "SRR3501926", "SRR3501928", "SRR3501929", "
     SRR3501934", "SRR3501937", "SRR3501938", "SRR3501939", "SRR3501941", "SRR3501968", "
     SRR3501972", "SRR3502004", "SRR3502006", "SRR3502008", "SRR3502009", "SRR3502011", "
     SRR3502012")

## import metadata table:
setwd("/Users/kt/Documents/Documents/Undergrad/4/4483E/test/")
sampleInfo <- read.csv("sampleInfo.csv", header=T, sep="\t", stringsAsFactors=F, quote = "",
```

```
    check.names=F, row.names=1, comment.char="")

library(stringr)
sampleInfo <- str_split_fixed(rownames(sampleInfo), ",", n = 2)
sampleInfo <- as.data.frame(sampleInfo)
colnames(sampleInfo) <- c("run", "condition")

sampleInfo2 <- DataFrame(sampleInfo[,2])
sampleInfo2

## change rownames so that they are the same as colnames of counts (check to make sure they are
    the same before running function)
rownames(sampleInfo2) <- NULL
rownames(sampleInfo2)  <- colnames(counts.d)
colnames(sampleInfo2) <- "condition"
head(sampleInfo2)

## combine matrix and metatable (counts and sampleInfo2)
ddsFullCountTable <- DESeqDataSetFromMatrix(
  countData = counts.d,
  colData = sampleInfo2,
  design= ~ condition
)
head(ddsFullCountTable)



# 3.1 Preparing data for DESeq2 pipeline ----
## did not subset by col since only one col represents col metadata (colData)
ddsFullCountTable$condition <- droplevels(ddsFullCountTable$condition)


# 3.2 Running DESeq2 pipeline ----
deseq.full <- DESeq(ddsFullCountTable)


# 3.3 Obtaining results ----
deseq.tax <- deseq.full
rownames(deseq.tax)<-paste(counts$tax.vector, rownames(deseq.tax), sep=":")

res <- results(deseq.tax, contrast=c("condition", "untreated", "controls"))
res

setwd("/Users/kt/Documents/Documents/Undergrad/4/4483E/test/deseq2-output/")
write.table(res.tax, file="deseqcvu.txt", sep="\t", quote=F, col.names=NA)

## obtain other "contrasts" (comparisons between conditions):
### copaxone v controls (control must be named second to appear on denominator)
res.cvcop <- results(deseq.tax, contrast=c("condition", "copaxone", "controls"))
res.cvcop

write.table(res.cvcop, file="deseqcvcop.txt", sep="\t", quote=F, col.names=NA)

### interferon v controls
res.cvint <- results(deseq.tax, contrast=c("condition", "interferon", "controls"))
res.cvint

write.table(res.cvint, file="deseqcvint.txt", sep="\t", quote=F, col.names=NA)



# 3.5 Inspecting results ----
# find number of significant results BEFORE adjusted p
sum( res$pvalue < 0.01, na.rm=TRUE )
sum( res.cvcop$pvalue < 0.01, na.rm=TRUE )
```

```
sum( res.cvint$pvalue < 0.01, na.rm=TRUE )

# find number of significant results AFTER adjusted p
sum( res$padj < 0.1, na.rm=TRUE )
sum( res.cvcop$padj < 0.1, na.rm=TRUE )
sum( res.cvint$padj < 0.1, na.rm=TRUE )

# subset the results to sort by log2 fold change est to get sig genes with strongest DECREASE/
    DOWN REG
resSig <- res[ which(res$padj < 0.1 ), ]
head( resSig[ order( resSig$log2FoldChange ), ] )
write.table(resSig, file="cvu-sig-down.txt", sep="\t", quote=F, col.names=NA)

resSig.cvcop <- res.cvcop[ which(res.cvcop$padj < 0.1 ), ]
head( resSig.cvcop[ order( resSig.cvcop$log2FoldChange ), ] )
write.table(resSig.cvcop, file="cvcop-sig-down.txt", sep="\t", quote=F, col.names=NA)

resSig.cvint <- res.cvint[ which(res.cvint$padj < 0.1 ), ]
head( resSig.cvint[ order( resSig.cvint$log2FoldChange ), ] )
write.table(resSig.cvint, file="cvint-sig-down.txt", sep="\t", quote=F, col.names=NA)

# subset to get strongest INCREASE/UP-REG
resSigUp <- tail( resSig[ order( resSig$log2FoldChange ), ] )
write.table(resSigUp, file="cvu-sig-up.txt", sep="\t", quote=F, col.names=NA)

resSigUp.cvcop <- tail( resSig.cvcop[ order( resSig.cvcop$log2FoldChange ), ] )
write.table(resSigUp.cvcop, file="cvcop-sig-up.txt", sep="\t", quote=F, col.names=NA)

resSigUp.cvint <- tail( resSig.cvint[ order( resSig.cvint$log2FoldChange ), ] )
write.table(resSigUp.cvint, file="cvint-sig-up.txt", sep="\t", quote=F, col.names=NA)



# 3.6 plots ----
## MA plots:
pdf("deseq2_cvu_MA_plot.pdf")
plotMA( res, alpha=0.1, main="MA_plot_for_controls_and_untreated_individuals", ylim = c(-1,1))
dev.off()

pdf("deseq2_cvcop_MA_plot.pdf")
plotMA( res.cvcop, alpha=0.1, main="MA_plot_for_controls_and_copaxone-treated_individuals", ylim
     = c(-1,1))
dev.off()

pdf("deseq2_cvint_MA_plot.pdf")
plotMA( res.cvint, alpha=0.1, main="MA_plot_for_controls_and_interferon-treated_individuals",
    ylim = c(-1,1))
dev.off()


# filtR protocol
devtools::install_github('bjoris33/filtR')
library(ALDEx2)
library(propr)

## TEST ----
### dada2
setwd("/Users/kt/Documents/Documents/Undergrad/4/4483E/test/Dada2-Output/")
mytable <- filtR::filtR(count_file = "dada2_nochim_tax.txt", rho_CO = 0.5, clr_CO = 3)

pdf("dada2_filtR.pdf")
filtR::filtR(count_file = "dada2_nochim_tax.txt", rho_CO = 0.5, clr_CO = 3)
dev.off()

setwd("/Users/kt/Documents/Documents/Undergrad/4/4483E/test/filtR")
```

```
write.table(mytable, file="dada2-rho0.5-clr5.txt", sep="\t")


### mothur
setwd("/Users/kt/Documents/Documents/Undergrad/4/4483E/test/mothuranalysis/")
mothur <- read.table("SRP075039_taxonomy_abundances_v3.0.tsv", header=T, sep="\t",
    stringsAsFactors=F, quote = "", check.names=F, row.names=1, comment.char="")
mothurf <- mothur[,1:105]
write.table(mothurf, "mothur_illumina_only.txt", sep="\t")
mytable <- filtR::filtR(count_file = "mothur_illumina_only.txt", rho_CO = 0.7, clr_CO = 5, plot
    = TRUE)


pdf("mothur_filtR.pdf")
filtR::filtR(count_file = "SRP075039_taxonomy_abundances_v3.0.tsv", rho_CO = 0.7, clr_CO = 5,
    plot = TRUE)
dev.off()

setwd("/Users/kt/Documents/Documents/Undergrad/4/4483E/test/filtR")
write.table(mytable, file="mothur-rho0.7-clr5.txt", sep="\t")


## CONTROL ----
### dada2
setwd("/Users/kt/Documents/Documents/Undergrad/4/4483E/control/dada2-output")
filtR::filtR(count_file = "dada2_nochim_tax.txt", rho_CO = 0.7, clr_CO = 5, plot = TRUE)

pdf("dada2_control_filtR.pdf")
dada2.control <- filtR::filtR(count_file = "dada2_nochim_tax.txt", rho_CO = 0.7, clr_CO = 5,
    plot = TRUE)
dev.off()

### QIIME
setwd("/Users/kt/Documents/Documents/Undergrad/4/4483E/control")
qiime <- filtR::filtR("control_24_samples_tax.tsv", rho_CO = 0.7, clr_CO = 5, plot = TRUE)

pdf("qiime_filtR.pdf")
filtR::filtR("control_24_samples_tax.tsv", rho_CO = 0.7, clr_CO = 5, plot = TRUE)
dev.off()

setwd("/Users/kt/Documents/Documents/Undergrad/4/4483E/control/filtR/")
write.table(qiime, "qiime_filtR_table.txt", sep="\t")



# BLAST Analysis
setwd("/Users/kt/Documents/Documents/Undergrad/4/4483E/control/dada2-output")
d <- read.table("dada2_nochim_tax.txt", sep="\t")
dd <- d[2:nrow(d),]
colnames(dd) <- c("seqvar", "SRR3832162", "SRR3832163", "SRR3832164", "SRR3832165", "SRR3832166"
    , "SRR3832167", "SRR3832168", "SRR3832169", "SRR3832170", "SRR3832171", "SRR3832172", "
    SRR3832173", "SRR3832174", "SRR3832175", "SRR3832176", "SRR3832177", "SRR3832178", "
    SRR3832179", "SRR3832180", "SRR3832181", "SRR3832182", "SRR3832183", "SRR3832184", "
    SRR3832185", "tax.vector")

# import and format spike-in alignments to DADA2 sequence variants from BLAST:
setwd("/Users/kt/Documents/Documents/Undergrad/4/4483E/control/blast")
spike <- read.table("LC140931-Alignment.txt", sep="\t")
colnames(spike) <- c("query_acc.ver", "SV", "identity", "alignment_length", "mismatches", "gap_
    opens", "q.start", "q.end", "s.start", "s.end", "evalue", "bit_score")

# subset alignment to only keep SVs with 100% identity and a sequence length that matches the
    query:
blast <- subset(spike, identity == 100.000 & alignment_length == 253 & q.end == 253)
blast
```

```
# match the sequence variants from the filtered blast output to the dada2 tax table:
library("data.table")
res <-setDT(dd)[seqvar %chin% blast1$SV]
write.table(res, file="spike1_results.txt", sep = "\t")
```