D. S. Lymperopoulou **dlympero@berkeley.edu**

16S Amplicon Sequencing Data Analysis

April 19, 2016

Introduction

Amplicon analysis is a multi-step process. There are decisions to be made in analysis and implementation. This tutorial contains the basic procedures using Mothur to analyze **16S rRNA gene sequences** obtained by **high-throughput sequencing** technology, and specifically Illumina sequencing. The flowchart and example provided in the tutorial is especially useful for **Beginners.** We will go through an example touching on a couple of the issues in amplicon analysis. We will then briefly touch on ways to explore the data using R.

Before the workshop, you need to:
1.Download and unzip Mothur software (v1.35.1) according to your OS.
https://github.com/mothur/mothur/releases/tag/v1.35.1
2. Download and install "FastQC" – make sure that you have Java installed too:
http://www.bioinformatics.babraham.ac.uk/projects/download.html#fastqc
3. Download the "Amplicon_Workshop_CGRL" folder – it contains two subfolders. One subfolder is called "Workshop_files" and includes the fastq (sequencing) files, the stability.files (mapping file), and reference files (SILVA). These reference files can be downloaded here for your future analyses http://www.mothur.org/wiki/Silva_reference_files. The second subfolder contains the output files of mothur from the analysis that we are going to run. I have already run it and I give you the files in case you get stuck somewhere in the tutorial today, so that we can all proceed with the analysis.
4. Download and install:
R: http://cran.rstudio.com
RStudio: http://www.rstudio.com/products/rstudio/download/

Optional
5. Install the following packages in R: "calibrate", "shape", "direct labels", "knitr", "clustsig", "ellipse, "plyr", "vegan", "ggplot2", and "phyloseq". Except for "phyloseq", all packages can simply be installed by running: **install.packages("package_name")**. Please follow the directions to successfully install "phyloseq": http://joey711.github.io/phyloseq/install

The working environment

We are going to use "mothur" for this workshop, but there are other software programs available (see reference section). Each of these works in a similar way, where you invoke scripts to analyze your data. Mothur is written in C++ and many tutorials are available online, including SOPs by the developers (http://www.mothur.org/wiki/Analysis_examples, http://www.mothur.org/wiki/MiSeq_SOP).

Preparing the sequences

The sequences that you have been generated with the 515F-806 R primers, from the Earth Microbiome Project (http://www.earthmicrobiome.org). Open terminal, and change directory ("cd") to the Amplicon_Workshop directory or wherever you have your files and mothur.

```
$ cd  Desktop/Amplicon_Workshop
```

You can see what files are in that folder with the following command.

```
$ ls
```

Paired sequences from the UC Berkeley facility are returned already separated by barcodes, each with a forward read (R1) and a reverse read (R2). Begin by concatenating all the R1 reads together; this just links all the sequences from the different files in series.

```
$ cat *R1*.fastq > R1.fastq
```

Repeat for the R2 reads.

```
$ cat *R2*.fastq > R2.fastq
```

A check on whether the correct files were concatenated is that the two files should be exactly the same size. What are the sizes of the files?

R1.fastq	
R2.fastq	

We are now going to use FastQC, to check on the quality of reads. This will inform if/where you want to trim sequences.

```
$ fastqc
```

This opens a graphic user interface (GUI). Click File then Open and navigate to the Amplicon_Workshop directory. Open your newly concatenated R1.fastq file. After loading, you can view Per Base Quality .png graph. Repeat for the R2.fastq file.

Describe the quality of the reads. Would you use both directions or just one? Would you trim?

R1.fastq	
R2.fastq	

There are several separate, stand-alone programs that trim the ends of reads (e.g. Trimmomatic, FastX-Toolkit). Mothur can do that too (trim.seqs – maxlength, minlength etc.)

Working in mothur

In Windows, you can double-click the executable mothur icon. In Mac, you can either right click and hit opern or open it from the terminal ($ ./mothur).

When using, please cite:
Schloss, P.D., et al., Introducing mothur: Open-source, platform-independent, community-supported software for describing and comparing microbial communities. Appl Environ Microbiol, 2009. 75(23):7537-41.

In the first command, we will extract the information from the forward (R1) and reverse (R2) fastq files.
"soil" = any fastq name in your folder, e.g., 2A_S3_L001_R1_001.fastq

mothur > fastq.info(fastq=soil.R1.fastq)
10000
……..
374958

This command will extract the sequence and quality score data from your fastq files.

Output File Names:
soil.R1.fasta
soil.R1.qual

[WARNING]: your sequence names contained ':'. I changed them to '_' to avoid problems in your downstream analysis.

In every step of the analysis, we can see what the sequences look like with the "summary.seqs" command.

mothur > summary.seqs(fasta=soil.R1.fasta, processors=4)

Using 4 processors.

	Start	End	NBases	Ambigs	Polymer	NumSeqs
Minimum:	1	250	250	0	3	1
2.5%-tile:	1	250	250	0	4	9374
25%-tile:	1	250	250	0	4	93740
Median:	1	250	250	0	5	187480
75%-tile:	1	250	250	0	5	281219
97.5%-tile:	1	250	250	0	8	365585
Maximum:	1	250	250	0	59	374958
Mean: 1	250	250	0	4.72324		
# of Seqs:	374958					

Output File Names:
soil.R1.summary

It took 4 secs to summarize 374958 sequences.

Repeat for R2.

mothur > fastq.info(fastq=soil.R2.fastq)

mothur > summary.seqs(fasta=soil.R2.fasta, processors=4)

Using 4 processors.

	Start	End	NBases	Ambigs	Polymer	NumSeqs
Minimum:	1	250	250	0	3	1
2.5%-tile:	1	250	250	0	4	9374

25%-tile:	1	250	250	0	4	93740f
Median:	1	250	250	0	5	187480
75%-tile:	1	250	250	0	5	281219
97.5%-tile:	1	250	250	0	6	365585
Maximum:	1	250	250	8	250	374958
Mean: 1	250	250	3.20036e-05	4.62882		
# of Seqs:	374958					

Now, we will pair the forward and reverse reads with the "make.contigs" command, which requires the "stability.files" as input. You should format the stability.files based on the form that you get your data back from your sequencing facility. In this tutorial, we only need a three-column format, where the first column corresponds to the sample, and the second and third columns to the R1 and R2 fastq files of this sample, respectively.

mothur > make.contigs(file=soil_stability.files, processors=4)

Now, let's make a fastq file out of the paired fasta and the quality file and check it with FastQC.

mothur > make.fastq(fasta=soil_stability.trim.contigs.fasta, qfile=soil_stability.contigs.qual)

mothur > fastq.info(fastq=soil_stability.trim.contigs.fastq)
10000
…….
374958

Output File Names:
soil_stability.trim.contigs.fasta
soil_stability.trim.contigs.qual

mothur > summary.seqs(fasta=soil_stability.trim.contigs.fasta, processors=4)

	Start	End	NBases	Ambigs	Polymer	NumSeqs
Minimum:	1	247	247	0	3	1
2.5%-tile:	1	253	253	0	4	9374
25%-tile:	1	253	253	0	4	93740
Median:	1	253	253	0	4	187480
75%-tile:	1	253	253	0	5	281219
97.5%-tile:	1	254	254	12	6	365585
Maximum:	1	500	500	92	250	374958
Mean: 1	253.178		253.178		1.16067	4.55027
# of Seqs:	374958					

It took 4 secs to summarize 374958 sequences.

Let's return back to the analysis. There are some troublesome observations, such as the 500 bp long sequences, the 12 and 92 ambiguous bases in some of them etc. The next command enables us to keep sequences that fulfill certain user-defined criteria.

mothur > screen.seqs(fasta=soil_stability.trim.contigs.fasta, group=soil_stability.contigs.groups, maxambig=0, maxlength=280)

Output File Names:
soil_stability.trim.contigs.good.fasta
soil_stability.trim.contigs.bad.accnos
soil_stability.contigs.good.groups

It took 9 secs to screen 374958 sequences.

mothur > summary.seqs(fasta=soil_stability.trim.contigs.good.fasta, processors=4)

Using 4 processors.

	Start	End	NBases	Ambigs	Polymer	NumSeqs
Minimum:	1	250	250	0	3	1
2.5%-tile:	1	253	253	0	4	7086
25%-tile:	1	253	253	0	4	70852
Median:	1	253	253	0	5	141704
75%-tile:	1	253	253	0	5	212555
97.5%-tile:	1	254	254	0	6	276321
Maximum:	1	280	280	0	11	283406
Mean: 1	253.12	253.12	0	4.55898		
# of Seqs:	283406					

Output File Names:
soil_stability.trim.contigs.good.summary

It took 2 secs to summarize 283406 sequences.

We anticipate that many of our sequences are duplicates of each other. Let's de-replicate our dataset.

mothur > unique.seqs(fasta=soil_stability.trim.contigs.good.fasta)
283406 73327

Output File Names:
soil_stability.trim.contigs.good.names
soil_stability.trim.contigs.good.unique.fasta

mothur > summary.seqs(fasta=soil_stability.trim.contigs.good.unique.fasta, name=soil_stability.trim.contigs.good.names, processors=4)

	Start	End	NBases	Ambigs	Polymer	NumSeqs
Minimum:	1	250	250	0	3	1
2.5%-tile:	1	253	253	0	4	7086
25%-tile:	1	253	253	0	4	70852
Median:	1	253	253	0	5	141704
75%-tile:	1	253	253	0	5	212555
97.5%-tile:	1	254	254	0	6	276321

Maximum:	1	280	280	0	11	283406
Mean:	1	253.12	253.12	0	4.55898	
# of unique seqs:		**73327**				
total # of seqs:		**283406**				

Now, let's create a count table that makes everything more affordable computationally.

mothur > count.seqs(name=soil_stability.trim.contigs.good.names, group=soil_stability.contigs.good.groups)

Using 4 processors.
It took 1 secs to create a table for 283406 sequences.

Total number of sequences: 283406

Output File Names:
soil_stability.trim.contigs.good.count_table

mothur > summary.seqs(count=soil_stability.trim.contigs.good.count_table)
Using soil_stability.trim.contigs.good.unique.fasta as input file for the fasta parameter.

We will need to align our sequences to a reference database (fasta & taxonomy). This is how mothur works! I downloaded silva_v119 (http://www.mothur.org/wiki/Silva_reference_files) and trimmed it to the region of interest (V4) of the 16S rRNA gene with the "pcr.seqs" command. This is not necessary, but it will make your computer's life easier.

mothur > summary.seqs(fasta=silva.v4.fasta, processors=4)

Using 4 processors.

	Start	End	NBases	Ambigs	Polymer	NumSeqs
Minimum:	1	20730	501	0	4	1
2.5%-tile:	2	21228	522	0	4	374
25%-tile:	2	21228	528	0	4	3740
Median:	2	21228	545	0	5	7479
75%-tile:	2	21228	549	0	5	11218
97.5%-tile:	2	21228	551	2	6	14583
Maximum:	10	21228	587	5	9	14956
Mean:	2.00053	21228	539.276		0.13192	4.97727
# of Seqs:	14956					

It took 4 secs to summarize 14956 sequences.

mothur > align.seqs(fasta=soil_stability.trim.contigs.good.unique.fasta, reference=silva.v4.fasta)

Output File Names:
soil_stability.trim.contigs.good.unique.align
soil_stability.trim.contigs.good.unique.align.report
soil_stability.trim.contigs.good.unique.flip.accnos

mothur > summary.seqs(fasta=soil_stability.trim.contigs.good.unique.align, count=soil_stability.trim.contigs.good.count_table, processors=4)

	Start	End	NBases	Ambigs	Polymer	NumSeqs
Minimum:	2	4	1	0	1	1
2.5%-tile:	7436	17018	253	0	4	7086
25%-tile:	7436	17018	253	0	4	70852
Median:	7436	17018	253	0	5	141704
75%-tile:	7436	17018	253	0	5	212555
97.5%-tile:	7436	17018	254	0	6	276321
Maximum:	21228	21228	280	0	11	283406
Mean:	7439.76	17022.5		253.037	0	4.55763
# of unique seqs:		**73327**				
total # of seqs:		**283406**				

We will re-run screen.seqs to make sure that everything overlaps the same region. We will get sequences to start at or before position 7436 and let mothur optimize the end (or optimize it at 17018). We will also set the maximum homopolymer length to 8 (this could have been done in the first execution of screen.seqs above).

mothur > screen.seqs(fasta=soil_stability.trim.contigs.good.unique.align, count=soil_stability.trim.contigs.good.count_table, summary=soil_stability.trim.contigs.good.unique.summary, start=7436, optimize=end, maxhomop=8)

Using 4 processors.
Optimizing end to 17018.

Output File Names:
soil_stability.trim.contigs.good.unique.good.summary
soil_stability.trim.contigs.good.unique.good.align
soil_stability.trim.contigs.good.unique.bad.accnos
soil_stability.trim.contigs.good.good.count_table

mothur > summary.seqs(fasta=soil_stability.trim.contigs.good.unique.good.align, count=soil_stability.trim.contigs.good.good.count_table, processors=4)

	Start	End	NBases	Ambigs	Polymer	NumSeqs
Minimum:	6701	17018	250	0	3	1
2.5%-tile:	7436	17018	253	0	4	7058
25%-tile:	7436	17018	253	0	4	70573
Median:	7436	17018	253	0	5	141146
75%-tile:	7436	17018	253	0	5	211719
97.5%-tile:	7436	17018	254	0	6	275234
Maximum:	7436	19021	280	0	8	282291
Mean:	7435.96	17019.2		253.082	0	4.55521
# of unique seqs:		**72628**				
total # of seqs:		**282291**				

It took 20 secs to summarize 282291 sequences.

Now that we know our sequences overlap the same alignment coordinates, we want to make sure they only overlap that region. So we will filter the sequences to remove the overhangs at both ends.

mothur > filter.seqs(fasta=soil_stability.trim.contigs.good.unique.good.align, vertical=T, trump=.)

If you did something wrong in the "screen.seqs" you will end up with nothing here.

mothur > summary.seqs(fasta=current, count=current)
Using soil_stability.trim.contigs.good.good.count_table as input file for the count parameter.
Using soil_stability.trim.contigs.good.unique.good.filter.fasta as input file for the fasta parameter.

	Start	End	NBases	Ambigs	Polymer	NumSeqs
Minimum:	1	525	234	0	3	1
2.5%-tile:	1	525	253	0	4	7058
25%-tile:	1	525	253	0	4	70573
Median:	1	525	253	0	5	141146
75%-tile:	1	525	253	0	5	211719
97.5%-tile:	1	525	254	0	6	275234
Maximum:	2	525	274	0	8	282291
Mean: 1.00006		525	253.07	0	4.5552	
# of unique seqs:		72628				
total # of seqs: 282291						

We will remove any redundancy that we created by running "unique.seqs" again.

mothur > unique.seqs(fasta=soil_stability.trim.contigs.good.unique.good.filter.fasta, count=soil_stability.trim.contigs.good.good.count_table)
72628 72621

Output File Names:
soil_stability.trim.contigs.good.unique.good.filter.count_table
soil_stability.trim.contigs.good.unique.good.filter.unique.fasta

mothur > summary.seqs(fasta=current, count=current)

	Start	End	NBases	Ambigs	Polymer	NumSeqs
Minimum:	1	525	234	0	3	1
2.5%-tile:	1	525	253	0	4	7058
25%-tile:	1	525	253	0	4	70573
Median:	1	525	253	0	5	141146
75%-tile:	1	525	253	0	5	211719
97.5%-tile:	1	525	254	0	6	275234
Maximum:	2	525	274	0	8	282291
Mean: 1.00006		525	253.07	0	4.5552	
# of unique seqs:		**72621**				
total # of seqs:		**282291**				

It took 1 secs to summarize 282291 sequences.

We will take care of more erroneous reads by running the "pre.cluster" command.

mothur > pre.cluster(fasta=soil_stability.trim.contigs.good.unique.good.filter.unique.fasta, count=soil_stability.trim.contigs.good.unique.good.filter.count_table, diffs=2)

……..

It took 62 secs to cluster 22113 sequences.
24879 14254 10625
Total number of sequences before pre.cluster was 24879.
pre.cluster removed 10625 sequences.

It took 81 secs to cluster 24879 sequences.
It took 84 secs to run pre.cluster.

mothur > summary.seqs(fasta=current, count=current)

	Start	End	NBases	Ambigs	Polymer	NumSeqs
Minimum:	1	525	234	0	3	1
2.5%-tile:	1	525	253	0	4	7058
25%-tile:	1	525	253	0	4	70573
Median:	1	525	253	0	5	141146
75%-tile:	1	525	253	0	5	211719
97.5%-tile:	1	525	254	0	6	275234
Maximum:	2	525	274	0	8	282291
Mean:	1.00006	525	253.07	0	4.55684	
# of unique seqs:		**37219**				
total # of seqs:		**282291**				

And now, let's identify any chimeric sequences.

mothur > chimera.uchime(fasta=soil_stability.trim.contigs.good.unique.good.filter.unique.precluster.fasta, count=soil_stability.trim.contigs.good.unique.good.filter.unique.precluster.count_table, dereplicate=t)

Using 4 processors.

uchime by Robert C. Edgar
http://drive5.com/uchime
This code is donated to the public domain.

Checking sequences from soil_stability.trim.contigs.good.unique.good.filter.unique.precluster.fasta ...

It took 407 secs to check 6882 sequences from group 2A.

It took 459 secs to check 7415 sequences from group 3A.

It took 758 secs to check 11631 sequences from group 4A.

It took 981 secs to check 14254 sequences from group 61A.

Now, let's remove these chimeras.

mothur >
remove.seqs(fasta=soil_stability.trim.contigs.good.unique.good.filter.unique.precluster.fasta, accnos=soil_stability.trim.contigs.good.unique.good.filter.unique.precluster.uchime.accnos)

mothur > summary.seqs(fasta=current, count=current)

	Start	End	NBases	Ambigs	Polymer	NumSeqs
Minimum:	1	525	234	0	3	1
2.5%-tile:	1	525	253	0	4	6174
25%-tile:	1	525	253	0	4	61737
Median:	1	525	253	0	5	123474
75%-tile:	1	525	253	0	5	185210
97.5%-tile:	1	525	254	0	6	240773
Maximum:	2	525	272	0	8	246946
Mean: 1.00002		525	253.07	0	4.5668	
# of unique seqs:		**18179**				
total # of seqs:		**246946**				

It is time we classified our sequences. I like doing this so as to remove unspecific sequences within mothur. Others do this further down in R. I do not see the reason of not further reducing my dataset as soon as possible.

mothur >
classify.seqs(fasta=soil_stability.trim.contigs.good.unique.good.filter.unique.precluster.pick.fasta, count=soil_stability.trim.contigs.good.unique.good.filter.unique.precluster.uchime.pick.count_table, reference=silva.nr_v119.ng.fasta, taxonomy=silva.nr_v119.tax, cutoff=65)

Using 4 processors.
Generating search database... DONE.
It took 129 seconds generate search database.

Reading in the silva.nr_v119.tax taxonomy... DONE.
Calculating template taxonomy tree... DONE.
Calculating template probabilities... DONE.
It took 222 seconds get probabilities.
Classifying sequences from soil_stability.trim.contigs.good.unique.good.filter.unique.precluster.pick.fasta ...
[WARNING]: M02248_9_000000000-AB7RF_1_1105_10669_16474 could not be classified. You can use the remove.lineage command with taxon=unknown; to remove such sequences.
…….
It took 292 secs to classify 18179 sequences.
It took 0 secs to create the summary file for 18179 sequences.

The primers we use are only supposed to amplify members of the Bacteria and if they are hitting Eukaryota or Archaea, it is a mistake. Also, chloroplasts and mitochondria have no functional role in a microbial community. There's also just the random stuff that we want to get rid of ("unknown" -> no domain). Now, we will remove these sequences.

mothur > remove.lineage(fasta=soil_stability.trim.contigs.good.unique.good.filter.unique.precluster.pick.fasta, count=soil_stability.trim.contigs.good.unique.good.filter.unique.precluster.uchime.pick.count_table, taxonomy=soil_stability.trim.contigs.good.unique.good.filter.unique.precluster.pick.nr_v119.wang.taxonomy, taxon=Archaea-Chloroplast-mitochondria-Eukaryota-unknown)

mothur > summary.seqs(fasta=current, count=current, processors=4)

	Start	End	NBases	Ambigs	Polymer	NumSeqs
Minimum:	1	525	234	0	3	1
2.5%-tile:	1	525	253	0	4	6108
25%-tile:	1	525	253	0	4	61078
Median:	1	525	253	0	5	122156
75%-tile:	1	525	253	0	5	183233
97.5%-tile:	1	525	254	0	6	238203
Maximum:	2	525	272	0	8	244310
Mean: 1.00002		525	253.071		0	4.56208
# of unique seqs:		**17967**				
total # of seqs:		**244310**				

Our unique sequences are 7% of our dataset. I think this is a high percentage of unique sequences and if you run the following command, you will be able to spit the abundance into abundant sequences and rare sequences. I will use a cutoff of one read to define a rare sequence.

mothur > split.abund(fasta=soil_stability.trim.contigs.good.unique.good.filter.unique.precluster.pick.pick.fasta, count=soil_stability.trim.contigs.good.unique.good.filter.unique.precluster.uchime.pick.pick.count_table, cutoff=1)

Summarize the abundant sequences.

mothur > summary.seqs(fasta=soil_stability.trim.contigs.good.unique.good.filter.unique.precluster.pick.pick.abund.fasta, count=soil_stability.trim.contigs.good.unique.good.filter.unique.precluster.uchime.pick.pick.abund.count_table)

	Start	End	NBases	Ambigs	Polymer	NumSeqs
Minimum:	1	525	234	0	3	1
2.5%-tile:	1	525	253	0	4	5891
25%-tile:	1	525	253	0	4	58903
Median:	1	525	253	0	5	117805
75%-tile:	1	525	253	0	5	176707
97.5%-tile:	1	525	254	0	6	229718
Maximum:	2	525	272	0	8	235608
Mean: 1.00001		525	253.069		0	4.56239
# of unique seqs:		**9265**				

total # of seqs: 235608

Interestingly, almost half of our unique sequences were singleton reads, and while this produced a great reduction in our unique sequences (which will make things easier for clustering and probably more meaningful ecologically), the total reduction of the whole dataset was only 3.5%.

Let's continue with traditional analysis and proceed to clustering the sequences into OTUs. There are two ways to do this in mothur: 1) the traditional way that we will do consists of calculating uncorrected pairwise distances between aligned DNA sequences and then clustering into OTUs based on your favorite clustering method implemented in mothur (furthest, average, and nearest neighbor), 2) the faster and memory saving cluster.split way, where taxonomic information is used to split the sequences into bins and then cluster within each bin.

mothur >
dist.seqs(fasta=soil_stability.trim.contigs.good.unique.good.filter.unique.precluster.pick.pick.fasta, cutoff=0.20)

Output File Names:
soil_stability.trim.contigs.good.unique.good.filter.unique.precluster.pick.pick.dist

It took 485 seconds to calculate the distances for 17967 sequences.

mothur >
cluster(column=soil_stability.trim.contigs.good.unique.good.filter.unique.precluster.pick.pick.dist, count=soil_stability.trim.contigs.good.unique.good.filter.unique.precluster.uchime.pick.pick.count_table)

```
******************#****#****#****#****#****#****#****#****#****#****#
Reading matrix:     ||||||||||||||||||||||||||||||||||||||||||||||||||||
***********************************************************************
changed cutoff to 0.0922386
```

Output File Names:
soil_stability.trim.contigs.good.unique.good.filter.unique.precluster.pick.pick.an.unique_list.list

It took 694 seconds to cluster

Now, we will create a table with the number of sequences per OTU and per sample.

mothur >
make.shared(list=soil_stability.trim.contigs.good.unique.good.filter.unique.precluster.pick.pick.an.unique_list.list, count=soil_stability.trim.contigs.good.unique.good.filter.unique.precluster.uchime.pick.pick.count_table, label=0.03)

And we will assign taxonomy.

mothur >
classify.otu(list=soil_stability.trim.contigs.good.unique.good.filter.unique.precluster.pick.pick.an.unique_list.list, count=soil_stability.trim.contigs.good.unique.good.filter.unique.precluster.uchime.pick.pick.count_

table,
taxonomy=soil_stability.trim.contigs.good.unique.good.filter.unique.precluster.pick.nr_v119.wang.pick.taxonomy, label=0.03)
reftaxonomy is not required, but if given will keep the rankIDs in the summary file static.
0.03 5314

Let's give some easier names to our final files. If you use windows, write "copy" instead of "cp".

mothur > system(cp
soil_stability.trim.contigs.good.unique.good.filter.unique.precluster.pick.pick.an.unique_list.shared
soil_final.shared)

mothur > system(cp
soil_stability.trim.contigs.good.unique.good.filter.unique.precluster.pick.pick.an.unique_list.0.03.cons.taxonomy soil_final.cons.taxonomy)

We need to figure out how many sequences we have per sample…

mothur > count.groups(shared=soil_final.shared)
2A contains 68535.
3A contains 64649.
4A contains 55745.
61A contains 55381.

Total seqs: 244310.

…and then normalize the sampling effort by sub-sampling with the lowest number of sequences observed.

mothur > sub.sample(shared=soil_final.shared, size=55381)

Output File Names:
soil_final.0.03.subsample.shared

We can evaluate the sequencing depth in our samples and plot it as shown below in R.

mothur > rarefaction.single(shared=soil_final.shared, label=0.03, freq=1000)

Indices are important, not only to calculate and plot them, but also to understand the information they provide and the limitations they might have.

mothur > summary.single(shared=soil_final.shared, calc=nseqs-coverage-sobs-shannon-invsimpson, subsample=55381)

mothur > quit

Mothur provides several other options for β-diversity analyses (e.g., calculation of ordination matrices, venn diagrams, heatmaps etc), as well as other ecological approaches such as the Indicator Species or the LEfSE (Linear discriminant analysis Effect Size) analyses. Plotting and other statistics can be done in R.

mothur > dist.shared(shared= soil_final.shared, calc=braycurtis, subsample=55381)
nmds(phylip=soil_final. braycurtis.0.03.lt.ave.dist)

*Particularly useful commands might be the "**get.oturep**", which generates a fasta-formatted sequence file containing only a representative sequence for each OTU, and the "**bin.seqs**", which can provide the full list of sequences for each OTU and a fasta file, if you provide a fasta file in the optional parameters.*

Open R – Basic R Package - This is not a R workshop, so we will explore basic ways of plotting data from mothur with R. Thing can get more sophisticated with packages such as vegan, phyloseq etc.

1) Plot a rarefaction curve

```
data<-read.table(file="soil_final.groups.rarefaction", header=T)
plot(x=data$numsampled, y=data$X0.03.2A, xlab="Number of Sequences Sampled",ylab="OTUs", type="l",
col="red", font.lab=3)
lines(x=data$numsampled, y=data$X0.03.3A, type="l", col="green", font.lab=3)
lines(x=data$numsampled, y=data$X0.03.4A, type="l", col="blue", font.lab=3)
lines(x=data$numsampled, y=data$X0.03.61A, type="l", col="black", font.lab=3)
legend(x=60000, y=600, c("2A", "3A", "4A", "61A"), c("red", "green", "blue", "black"))
```

There are more efficient ways to do this, e.g., in the vegan package.

2) Plot a nMDS ordination plot using a mothur-outputted matrix.

```
Nmds <- read.table("……axes", header=T)
nmds.col<-c(rep("green", 2), rep("blue", 2))

plot(nmds$axis2~nmds$axis1, col=nmds.col, xlab="Axis 1", ylab="Axis 2")
legend(x="x", y="y", legend=c("Altitude 1", " Altitude 2"), pch=1, col=c("green", "blue"))

plot(nmds$axis2~nmds$axis1, col=nmds.col, xlab="Axis 1", ylab="Axis 2", pch=18,
        cex=2)
legend(x=="x", y="y", legend=c("Altitude 1", " Altitude 2"), pch=18, cex=1, col=c("pink",
        "blue"))

plot(nmds$axis2~nmds$axis1, col=nmds.col, xlab="Axis 1", ylab="Axis 2", pch=18,
        cex=2, type="b")
legend(x=0.3, y=0.6, legend=c("Altitude 1", " Altitude "), pch=18, cex=1, lty=1,
        col=c("green", "blue"))
```

*You can also overlay two graphs on top of each other using the points command. Do this for the taxa that drive this grouping by using the output of the "**corr.axes**" command in mothur.*

3) Use the "phyloseq: package, make a "*phyloseq object*" and graph relative abundances bars.

```
sharedFile = read.table("soil_final.0.03.subsample.shared")
sharedFile = t(sharedFile)
rownames(sharedFile) = sharedFile[,1]
colnames(sharedFile) = sharedFile[2,]
sharedFile = sharedFile[,2:5]
```

```
sharedFile = sharedFile[4:5241,]
class(sharedFile) <- "numeric"
taxFile = read.table('soil_final.taxonomy.txt', header=T, sep='\t')
rownames(taxFile) = taxFile[,1]
taxFile = taxFile[,3:8]
taxFile = as.matrix(taxFile)
metaFile = read.table('soil_stability.meta.txt', header=T, sep='\t')
rownames(metaFile) = metaFile[,1]
metaFile = metaFile[,2:3]

OTU = otu_table(sharedFile, taxa_are_rows = TRUE)
TAX = tax_table(taxFile)
META = sample_data(metaFile)
Soil_physeq = phyloseq(OTU, TAX, META)

Soil_physeq <- prune_taxa(taxa_sums(Soil_physeq) > 0, Soil_physeq)
sum(taxa_sums(Soil_physeq) == 0) #check if you have OTUs with "0" abundance

#Transform to proportions
Soil_physeqP = transform_sample_counts(Soil_physeq, function(x) 100 * x/sum(x))

#Plot the top20 OTUs
top20otus = names(sort(taxa_sums(Soil_physeqP), TRUE)[1:20])
taxtab20 = cbind(tax_table(Soil_physeqP), genus20 = NA)
taxtab20[top20otus, "genus20"] <- as(tax_table(Soil_physeqP)[top20otus, "Genus"], "character")
tax_table(Soil_physeqP) <- tax_table(taxtab20)

# plot with panelling by CompEvent for an easier PMA-wise comparison
Soil_physeq_P20 = prune_taxa(top20otus, Soil_physeqP)
title = "Relative Abundance of Top20 OTUs"
p1 = plot_bar(Soil_physeq_P20, fill = "genus20", title = title) + labs(colour = "genus")

p1 + facet_wrap(~Method)
```