**Deblurred bog dataset**

**Alex Linz, 1-7-15**

**Why deblur?**

In order to reduce the number of OTUs unclassified at the phylum level, the raw sequencing reads from the original bog Illumina run were processed using QIIME’s deblurring algorithm. Although this method is not yet published, here is the description provided in personal communication:

“The idea of deblurring is trying to get rid of the read errors so only real sequences are left.

deblurring uses a prediction of the read error profile of the sequences in order to throw sequences suspected as read errors. Therefore, there is no clustering and no similarity threshold. Ideally, it can tell apart 2 sequences which differ only in 1 bp.

HOWEVER

due to the inaccuracy of the read error model, if there 2 bacteria in the mixture that differ in 1 bp and their frequencies are very different (i.e one bacteria is > 10 times more frequent than the other bacteria), then deblurring may throw away all the reads of the lower freq. bacteria (think they are all read errors).

in another way:

1. Hopefully, most of the OTUs (sequences) identified by deblurring should be actually present in the sample (so no read errors...). So each OTU sequence is exact.

2. All bacterial sequences present in the sample will be identified as OTUs AS LONG AS they are not 1bp different from another sequence which is present in much higher frequency in the sample. so the OTU table should cover almost all bacterial sequences in the sample”

**Files you will need:**

An alignment database – I used Greengenes’ core\_set\_aligned.fasta.imputed from <http://greengenes.lbl.gov/Download/Sequence_Data/Fasta_data_files/core_set_aligned.fasta.imputed>

Our Freshwater Classification database, in the form of a fasta file and a taxonomy file. Available at <https://github.com/mcmahon-uw/FWMFG>

The latest Greengenes databases – I used gg\_13\_5.fasta and gg\_13\_5.taxonomy from <http://greengenes.secondgenome.com/downloads/database/13_5>. However, this must be processed before being used in mothur. Trina has a script to do this. All sequences already in our freshwater database must be removed. Semicolons at the end of classifications should be removed, add spaces between seqIDs and fasta sequences, and put a new line character after each fasta sequence.

The scripts indexfasta.pl and fetchseqs.pl from the perl-cmdtools folder on Zissou. Either add these to your path or copy them into your working directory.

**QIIME to mothur**

The output of deblurring included 3 files:

bogs.clean.min25.biom

bogs.clean.min25.seq.fa

bogs.mapping.txt

The biom file can be opened in mothur using the command:

make.shared(biom=bogs.clean.min25.biom)

This produces a file named bogs.clean.min25.shared. It looks like an OTU table, with samples as the rows and sequences as the columns. The data is a matrix of relative abundance data. Since no group file is provided, mothur creates a column of group names all called “dummy”. A .rabund file is produced for each sample and contains relative abundance data for each individual sample.

The fasta file contains representative sequences, not the raw sequencing reads – identical reads are lumped together under a single ID. The IDs correspond to the sequence IDs in the .shared file. The mapping file contains metadata about each sample – where it was collected, the sequencing method used, and much more. However, since it was all identical or coded in sample names, I didn’t use this file downstream.

mothur requires a .names, .groups, and .fasta file in its workflow. To make the QIIME files compatible with mothur, I wrote a series of R scripts that change file formats or skip over steps requiring those files:

*names\_from\_shared.R* – Creates a .names file from the .shared file created from the .biom file. Since it has only representative sequences, it’s not that informative, but I used it to keep track of which sequences were removed in various quality control steps

*remove\_seqs\_from\_shared.R* – Keeps only sequences in the .shared file that remain in the current .names file. This is necessary when poor alignments, chimeras, or chloroplasts are removed from the dataset.

*shared\_to\_count.R* – A .count\_table is typically generated from a .names and a .groups file and looks a lot like the .shared, but transposed and with less information. It can be used in certain steps in place of a .names and .groups combination.

All scripts will prompt the user to type in one or more input file names, as well as the name of the output file. The code for them is at the bottom of this workflow.

**Workflow:**

This workflow uses mothur v.1.34.4, R, and Unix. Make sure to perform it on a system that can do all three and has the processing power to handle the clustering steps (such as Zissou). Also keep in mind that mothur will use the last file of a given type generated if one is not provided.

########################

*#Rename the bogs.clean.min25.seq.fa to a .fasta file to make mothur happy*

mv bogs.clean.min25.seq.fa bogs.clean.min25.fasta

*#Convert the biom file into a shared file*

mothur

make.shared(biom=bogs.clean.min25.biom)

quit()

*#This produced a lot of .rabund files besides the shared that are not used. I'll use a wildcard to remove all .rabunds*

rm \*.rabund

*#Make the names file using the R script*

R

source("names\_from\_shared.R")

Input file name:1: bogs.clean.min25.shared

Output file name:1: bogs.clean.min25.names

q()

####################################

The following steps perform sequence quality control in mothur using bogs.clean.min25.fasta and bogs.clean.min25.names.

####################################

mothur

align.seqs(candidate=bogs.clean.min25.fasta, template=../Classification databases/core\_set\_aligned.fasta.imputed)

*#Remove the heads and tails of alignments that are all periods in 90% of sequences, as well as sequences with more than 6 homopolymers in a row and sequences that are all periods*

screen.seqs(optimize=start-end, criteria=90, maxhomop=6, name=bogs.clean.min25.names) filter.seqs(vertical=T, trump=.)

*#Since the deblurring kept only unique sequences, the next command may be redundant*

unique.seqs()

*#Pre-cluster into rough OTUs for input into the uchime chimera finder.*

pre.cluster(diffs=2)

chimera.uchime(name=bogs.clean.min25.good.filter.unique.precluster.names)

*#Remove chimeras from the unclustered names and fasta files instead of the currently clustered files, otherwise you will miss many sequences*

remove.seqs(name=bogs.clean.min25.good.filter.names, fasta=bogs.clean.min25.good.filter.unique.fasta)

*#Do a classification to identify and remove chloroplasts*

classify.seqs(template=gg\_13\_5\_97\_otus\_noFW.fasta, taxonomy=gg\_13\_5\_97\_otus\_noFW.taxonomy, cutoff=60)

remove.lineage(taxon=k\_\_Bacteria;p\_\_Cyanobacteria;c\_\_Chloroplast;, name=bogs.clean.min25.good.filter.pick.names, fasta=bogs.clean.min25.good.filter.unique.pick.fasta)

quit()

*#Rename the last names and fasta files something more reasonable*

mv bogs.clean.min25.good.filter.unique.precluster.pick.pick.names qc.bogs.clean.min25.names

mv bogs.clean.min25.good.filter.unique.precluster.pick.pick.fasta qc.bogs.clean.min25.fasta

*#Keep only sequences from the current names file in the .shared file*

R

source("remove\_seqs\_from\_shared.R")

Input shared file:1: bogs.clean.min25.shared

Input names file:1: qc.bogs.clean.min25.names

Output shared file:1: qc.bogs.clean.min25.shared

source(“count\_from\_shared.R”)

Input shared file:1: qc.bogs.clean.min25.shared

Output count file:1: qc.bogs.clean.min25.count\_table

q()

*#The current files are qc.bogs.clean.min25.shared, qc.bogs.clean.min25.names, qc.bogs.clean.min25.fasta, and qc.bogs.clean.min25.count\_table*

################################

The following steps make an OTU table with serial taxonomic assignments. I chose to cluster OTUs before subsampling to avoid errors about sequences missing in the subsampled .shared file that are present in the .fasta file.

################################

dist.seqs(fasta=qc.bogs.clean.min25.fasta, output=lt)

cluster(method=average)

*#Set label=0.02 for 98% similar OTUs*

make.shared(list=qc.bogs.clean.min25.phylip.an.list, count=qc.bogs.clean.min25.count\_table, label=0.02)

system(rm \*.rabund)

*#Samples with no reads were removed in the previous step.*

####################################

I chose to sample to 2500 as a compromise between keeping enough reads for good quality and keeping the threshold low enough to retain as many samples as possible. I ran some simulations that showed a) quality breaks down at sample size < 500 as seen in banding patterns and b) the diversity of the bog communities is not captured even in the raw numbers of reads, so they are conceptually already a subsample. Samples with fewer than 2500 reads are removed in this step.

####################################

sub.sample(persample=T, size=2500, shared=qc.bogs.clean.min25.phylip.an.shared)

*#Classify the OTUs. We do this first in our freshwater database. Anything that is identified at less than 70% confidence is passed to the Greengenes database.* classify.seqs(fasta=qc.bogs.clean.min25.fasta, template=FW\_trainingset\_MMBR\_strict\_12July12.fasta, taxonomy=FW\_trainingset\_MMBR\_strict\_12July12.taxonomy, cutoff=70)

system(grep -v "^[^;]\*;[^;]\*;[^;]\*;[^;]\*;unclassified" qc.bogs.clean.min25.FW\_trainingset\_MMBR\_strict\_12July12.wang.taxonomy > FW.70.taxonomy)

system(grep "^[^;]\*;[^;]\*;[^;]\*;[^;]\*;unclassified" qc.bogs.clean.min25.FW\_trainingset\_MMBR\_strict\_12July12.wang.taxonomy > unclassified\_toGG.taxonomy)

system(cut -f 1 unclassified\_toGG.taxonomy > unclassified\_toGG.list.txt)

system(indexfasta.pl qc.bogs.clean.min25.index qc.bogs.clean.min25.fasta)

system(fetchseq.pl -i unclassified\_toGG.list.txt -x qc.bogs.clean.min25.index -o unclassified\_toGG.fasta)

classify.seqs(fasta=unclassified\_toGG.fasta, template=gg\_13\_5\_97\_otus\_noFW.fasta, taxonomy=gg\_13\_5\_97\_otus\_noFW.taxonomy, cutoff=60)

system(mv unclassified\_toGG.gg\_13\_5\_97\_otus\_noFW.wang.taxonomy GG.60.taxonomy)

system(cat FW.70.taxonomy GG.60.taxonomy > all.FWGG.taxonomy)

*#Output summary files for the taxonomy. .taxonomy contains the phylogenetic assignment for each OTU. The sum.tax file is less useful since there is no .groups file – it contains the totals broken down by successive phylogenetic groups.*

summary.tax(taxonomy=all.FWGG.taxonomy)

classify.otu(list=qc.bogs.clean.min25.phylip.an.list, taxonomy=all.FWGG.taxonomy, label=0.02, cutoff=60, basis=sequence)

**Final Product**

The OTU table is contained in the file qc.bogs.clean.min25.phylip.an.0.02.subsample.shared. I will do some further processing to make it compatible with my R scripts – removing the label = 0.02 column, transposing so that samples are columns and OTUs are rows, and shortening/standardizing sample names.

The phylogenetic assignments are in the file qc.bogs.clean.min25.phylip.an.0.02.cons.taxonomy. This contains all of the OTUs and their phylogenies. There may be OTUs that do not exist in the OTU table because the subsampling occurred after OTU clustering. These can be removed if necessary.

Finally, mothur also typically produces a .sum.tax file that contains each phylogenetic assignment separate from the OTU ID. I did not make this, since I do not have a .groups file. However, since it contains all of the information in the .taxonomy file and the .shared file, I do not think it is necessary. The sum.tax is more user-friendly, but the .taxonomy is more R-friendly.