Introduction of QIIME

Lab meeting

Jan 2013

Greg Gu

Metagenomics

- Kevin Chen and Lior Pachter (UC, Berkeley) "...the study of communities of microbial organisms directly in their natural environments, bypassing the need for isolation and lab cultivation of individual species" 2005
  - 1980s
  - 16S rRNA, Bacteria, Shotgun Sequencing
  - 2005 high-throughput sequencing (454 Illumina)

Challenges in metagenomics

- DNA directly from environmental samples
  - “Dirty”: huge data set size, 10K species mixtures, other noises, etc
  - Typically Short (very hard to get a few thousand base pair)
  - PCR, Primer design
- High-throughput sequencing
  - Cost
  - Sequence length

Bioinformatics

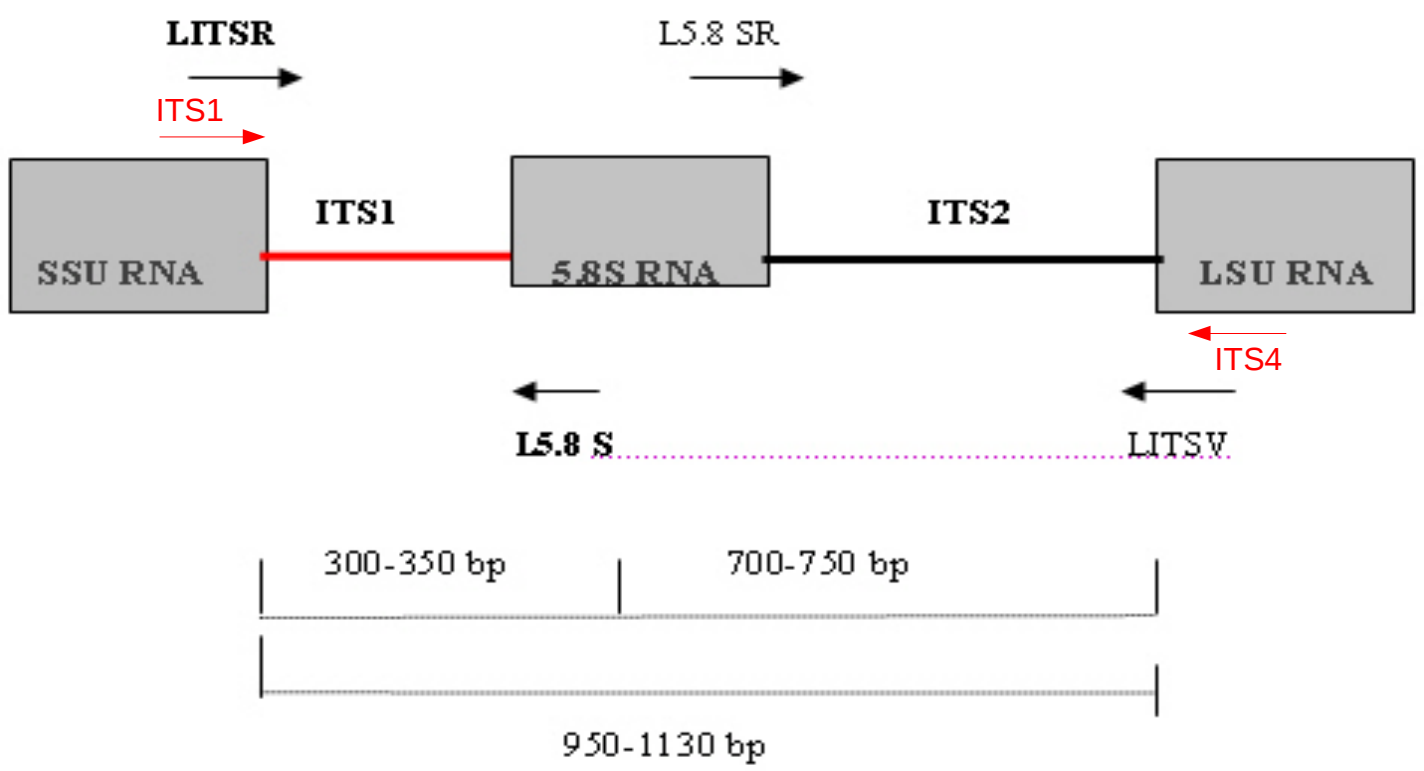
- Limited by tools
- Limited by database
  - Matadata database
  - Taxonomy database
- binding seq to species

Bar Code for fungi

- Ideally, a DNA barcode region should be chosen which are likely to include universality of primers, feasibility and species-level resolution.

- Large subunit (LSU) D1/D2 for yeasts; internal transcribed spacer(ITS) for ectomycorrhizal fungi, etc

ITS


LITSR
L5.8 SR
ITS1
ITS1
ITS2
SSU RNA
5.8S RNA
LSU RNA
ITS4
L5.8 S
LITSV
300-350 bp
700-750 bp
950-1130 bp

QIIME (Quantitative Insights Into Microbial Ecology) Pipeline

- Written in Python and free to use
- Wrap a lot third party tools, (PyNAST, RDP)
- Run locally or remotely
- Kind of easy to use
- UNITE/QIIME ITS ref database
- GreenGenes

QIIME working flow

- Clean sequence reads for quality and assign multiplexed reads to starting samples (check_id_map.py, split_libraries.py)
- Pick Operational Taxonomic Units (OTUs) based on sequence similarity within the reads, and pick a representative sequence from each OTU. (pick_reference_otus_through_otu_table.py)
- Assign the OTU to a taxonomic identity using reference databases.
- Align the OTU sequences and create a phylogenetic tree.
- Calculate diversity metrics for each sample and compare the types of communities, using the taxonomic and phylogenetic assignments.
- Generate plots and figures to visually depict the differences between the samples

```
#!/bin/bash
module load qiime
module load stajichlab
module load stajichlab-python
module load java/1.6.0_29
module load AmpliconNoise/1.25
module load cd-hit/3.1.1
module load cdbfasta/0.99
module load mothur/1.25.0
module load microbiomeutil/r20110519
module load clearcut/1.0.9
module load infernal/1.0.2
module load RAxML/7.3.2
module load pplacer/1.1.alpha13
module load ParsInsert
module load rtax/0.982
module load muscle/3.8.31
module load uclust
module load usearch/5.2.32
module load FastTree/2.1.3
module load ncbi-blast/2.2.22
cd '/rhome/daigu/stajichlab/projects/Built_Env/Kinney_ITS_2012/Split Library/split_library_output_site24f'
#Split libraries
split_libraries.py -m seqs_map.txt -f seqs.fna -q seqs.qual -o split_library_output -b 8
#OTUs by its database
pick_reference_otus_through_otu_table.py -i split_library_output/seqs.fna -o otus/ -r 97_otus_ln.fasta -t 97_tax_ln.txt
#OTU Heatmap
make_otu_heatmap_html.py -i otus/uclust_ref_picked_otus/otu_table.biom -o otus/OTU_Heatmap/
#OTU Network
make_otu_network.py -m seqs_maps.txt -i otus/uclust_ref_picked_otus/otu_table.biom -o otus/OTU_Network
#Make Taxa Summary Charts
summarize_taxa_through_plots.py -i otus/uclust_ref_picked_otus/otu_table.biom -o wf_taxa_summary -m seq_map_corrected.txt
#making a phylogenetic tree
#assign taxonomy
assign_taxonomy.py -i split_library_output/seqs.fna -o otus/
#align the seqs by muscle/PyNAST
align_seqs.py -i split_library_output/seqs.fna [-m muscle] -o otus/
#make the tree
make_phylogeny.py -i otus/seqs_rep_set_aligned.fasta -t  fasttree -o otus/rep_set.tre -l otus_tree.log
#Alpha rarefaction"
echo "alpha_diversity:metrics shannon,PD_whole_tree,chao1,observed_species" > alpha_params.txt
alpha_rarefaction.py -i otus/uclust_ref_picked_otus/otu_table.biom -m seqs_map.txt -o wf_arare/
#not work due to "phylogenetic metric supplied, but no phylogenetic tree supplied"
#Beta diversity
beta_diversity_through_plots.py -i otus/uclust_ref_picked_otus/otu_table.biom -m seqs_map.txt -o wf_bdiv_even146/
#Jackknifed beta diversity
jackknifed_beta_diversity.py -i otus/otu_table.biom -t otus/rep_set.tre -m seqs_map.txt -o wf_jack -e 110
#Make Bi-Plots
make_3d_plots.py -i wf_bdiv_even146/unweighted_unifrac_pc.txt -m seqs_map.txt -t wf_taxa_summary/otu_table_L3.txt --n_taxa_keep 5 -o 3d_biplot
```

Pre-processing

check_id_map.py

- Pass

split_libraries.py -m seqs_map.txt -f seqs.fna -q seqs.qual -o split_library_output -b 8

- Raw input seqs: 50,819
- Barcode mismatch: 9,069
- Primer mismatch: 34,459
- Other proble: 314
- Valid seqs: 6977(~1/7)

Create OTU table

pick_reference_otus_through_otu_table.py -i split_library_output/seqs.fna -o otus/ -r 97_otus_ln.fasta -t 97_tax_ln.txt

- 104 OTUs
- 2,978 failures (out of 6,977, ~40%)

Visualize OTU

- Heatmap

  html file create

- OTU networks

  Cytoscape network

Taxonomic summary

summarize_taxa_through_plots.py -i otus/uclust_ref_picked_otus/otu_table.biom -o wf_taxa_summary -m seq_map_corrected.txt

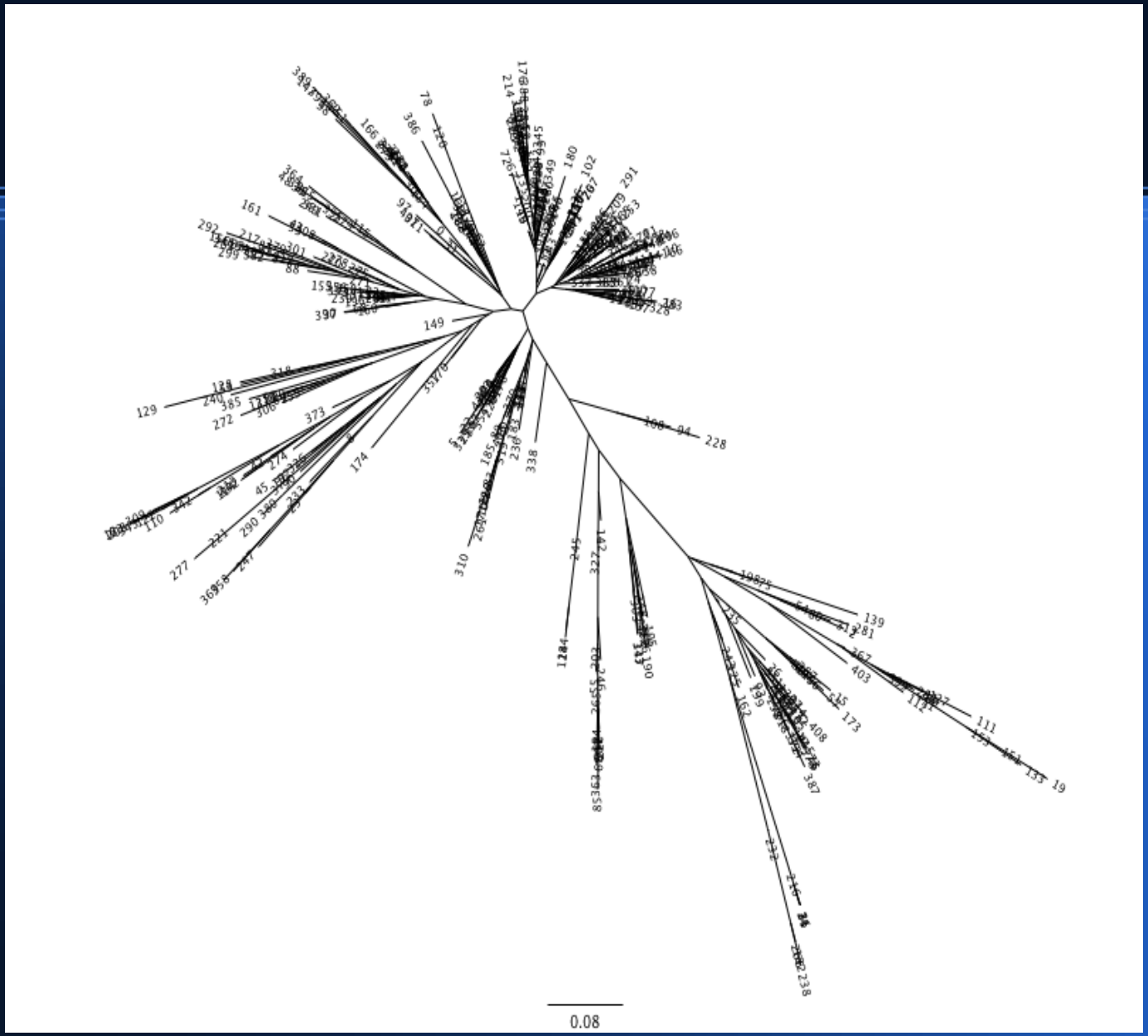
html file create

Phylogenetic tree

align_seqs.py -i split_library_output/seqs.fna [-m muscle] -o otus/

no error but only empty fasta file create

make_phylogeny.py -i otus/[pynast_aligned_seqs]/seqs_rep_set_aligned.fasta -t fasttree -o otus/rep_set.tre -l otus_tree.log


0.08

alpha rarefaction
beta diversity
3D Bi-plot

KiNG plot

Next

- Find out a solution for tree generation
- Recheck filtered out seqs
- Compare the results with the output from other tools, VAMPS
- Detail understand the meaning of each plot
- Compare the results by using different database, GreenGenes

```
#!/usr/bin/perl
use strict;
use warnings;


my $counter = 1;
my $format = "freq";
my $format2 = "frq";
my @buffer =();
my @fileList = ();
print "#####################################\n";
print "\n";
print "This program will clean up all your freq file in current folder.\n";
print "Any sequence less than 50bp long will be ignored. \n";
print "A new fasta format file for each input freq file. \n";
print "\n";
print "#####################################\n";
print "Created by Greg Gu from Jason Stajich's lab in UC Riverside.\n";
print "Email Greg using daigu\@ucr.edu for any more help.\n";
print "#####################################\n";


my $dir = ".";
opendir(DIR,$dir);
@fileList = sort(grep {/\.$format$/} readdir(DIR));
closedir(DIR);
opendir(DIR,$dir);
my @temp = sort(grep {/\.$format2$/} readdir(DIR));
push (@fileList,@temp);
closedir(DIR);
my $fileNumbers = @fileList;
if ($fileNumbers >0){
foreach my $fileName (@fileList){
#$fileName =~ s/[\n\r\t\f]//g;
#while ($fileName ne "stop")
open inFile, $fileName or die $!; #only good for same directory}
$fileName =~ s/\.[^.]+$//;
while (<inFile>){
my @s=split();
my @s=split();
if (length($s[1])>50){
my $newLine = ">Query"."$counter"."\n";
push(@buffer, $newLine);
push(@buffer, ($s[1],"\n"));
$counter++;
}
else {$counter++;}
}
close inFile;
open outFile, ">$fileName.fasta" or die $!;
print outFile @buffer;
close outFile;
print "Done with "."$fileName.fasta"."!\n";
@buffer = ();
$counter = 1;
#$fileName = <STDIN>;
#$fileName =~ s/[\n\r\f\t]//g;
}
print "ALL DONE. \n";
}
else{
print "There are no freq files in current directory.\n";
print "Did nothing, quit without error.\n";
}
```

```
#!/usr/bin/perl
use strict;
use warnings;


#use File::Basename;
#use Bio::Seq;


sub writeBuffer{
# @buffer2, $subCounter, $fileName
        my @arr = @{ $_[0] };
        foreach my $l (@arr){
        my $outName = "$_[2]"."_"."$_[1]";
        open outFile, ">$outName.fasta" or die $!;
        print outFile @arr;
        close outFile;
}
}
my $format = "fasta";
my @fileList = ();
my $counter = 0;
my $subCounter = 1;
#my @buffer1 = ();
my @buffer2 = ();
print "#####################################\n";
print "\n";
print "This program will clean up your ITS file in this folder for any sequence less than 25bp long. \n";
print "The Program will alse creat new files for each input fasta file. \n";
print "Each new file will NOT longer than 500 records.\n";
print "\n";
print "#####################################\n";
print "Created by Greg Gu from Jason Stajich's lab in UC Riverside.\n";
print "Email Greg using daigu\@ucr.edu for any more help.\n";
print "#####################################\n";
#my $fileName = <STDIN>;
my $dir = ".";
opendir(DIR,$dir);
@fileList = sort(grep {/\.$format$/} readdir(DIR));
closedir(DIR);
my $fileNumbers = @fileList;
my %Hbuffer1 = ();
my $key = "";


    print "There are no freq files in current directory.\n";
    print "Did nothing, quit without error.\n";
}


if ($fileNumbers > 0){
                foreach my $fileName (@fileList){
                open inFile, $fileName or die $!; #only good for same
directory}
                $fileName =~ s/\.[^.]+$//; #get only name without
extention
                foreach my $l (<inFile>){
                        if ($counter % 2 == 1 and length($l) >24)
{$Hbuffer1 {$key} = $l;} #fillup the hash buffer
                        else {if ($counter % 2 == 0) {$key = substr $l,
6;
                                $key = int ($key);}}   #creat a new key
                        #else {push (@buffer2,$l);}
                        $counter++;
                }
                close inFile;
                #foreach my $seq (@buffer1){
                #foreach $key (keys %Hbuffer1){
                foreach (sort {$a <=> $b} keys(%Hbuffer1)) {
                        #if (length($seq)>24){
                        #push into buffer2;
                        #my $newLine = $key;
                        my $newLine = ">Query".$_."\n";
                        push(@buffer2, $newLine);
                        push(@buffer2, $Hbuffer1{$_});
                        #$counter++;
                        #all required data ready
                #if buffer2 is full then write buffer;
                if (scalar(@buffer2)>999){
                        #write buffer into a file
                        writeBuffer(\@buffer2,$subCounter,
$fileName);
                        #reset all related counters;
                        $subCounter += 1;
                        @buffer2 =();
                        #$counter = 1; #if the new file should
be always start at query1
                        }
                }
                writeBuffer(\@buffer2,$subCounter,$fileName); #save
whatever left into the last file
                $subCounter = 1;
                @buffer2= () ;
                print "Done with "."$fileName.fasta"."!\n";
        }
        print "ALL DONE. \n";
}
else{print "Did nothing, quit without error.\n";}
```