Microbial eukaryotic 18S tag-sequence processing/QC - V4 region (automating) Version 2

Sarah Hu

Abstract

This protocol provides step-by-step instructions for implementing a perl script to easily automate quality checking of raw tag sequences. This was specifically constructed for the numerous 18S diversity projects (which target the V4 hypervariable specific for evaluting protistan diversity).

It is highly recommended that you follow along this Github repo: https://github.com/shu251/QC_steps_V4_tagsequencing

Citation: Sarah Hu Microbial eukaryotic 18S tag-sequence processing/QC - V4 region (automating). **protocols.io** dx.doi.org/10.17504/protocols.io.j9bcr2n
Published: 11 Oct 2017

Guidelines

Protocol describes quality checking process for raw fastq sequences. Includes some discussion for alternate approaches. End product is ready for OTU clustering. I've also included suggestions for OTU clustering at the end (specific for 18S data). Much of this pipeline can be applied for any tag sequencing analysis, but keep in mind this was specifically constructed to analyze single-celled microbial eukaryotic communities.

This protocol (and my preferred method) is to perform as much quality checking on each individual sample before combining for downstream OTU clustering (or other analysis). This way, I can track any samples that may be not ideal for downstream analysis.

Full github repo: https://github.com/shu251/QC_steps_V4_tagsequencing

Before start

Required programs:

- Database (preferrably with associated taxa list) to align sequences to. A version of the PR2 database formatted for QIIME can be found here
- QIIME - v.1.9.1 or higher. Note as of Jan 2018 QIIME1 will not be supported. Future iterations of this protocol will use QIIME2.
- fastqjoin - this should install with QIIME. Alternatively you can use PEAR to merge PE sequences.
- Trimmomatic - v.0.32
- vsearch - v1.11.1
- Make sure you know your primer sequences and generate a fasta file: 'trimPE_V4primer.fasta'. In the below example I'm using V4 Stoeck et al. 2010 primers

Protocol

Pre-processing

Step 1.

Input raw fastq files that are already demultiplexed. This tutorial uses V4 primers (Stoeck et al. 2010) for microbial eukaryotes

First create a text file with a list of file prefixes (prefix.txt).

Example:

Test01_L001_R1_001.fastq

Test01_L001_R2_001.fastq

Test02_L001_R1_001.fastq

Test02_L001_R2_001.fastq

Associated prefixes:

Test01

Test02

Programs required:

- Database (preferrably with associated taxa list) to align sequences to. A version of the PR2 database formatted for QIIME can be found here
- QIIME - v.1.9.1 or higher. Note as of Jan 2018 QIIME1 will not be supported. Future iterations of this protocol will use QIIME2.
- fastqjoin - this should install with QIIME. Alternatively you can use PEAR to merge PE sequences.
- Trimmomatic - v.0.32

- vsearch - v1.11.1
- Make sure you know your primer sequences and generate a fasta file: 'trimPE_V4primer.fasta'. In the below example I'm using V4 Stoeck et al. 2010 primers
- Create a 'prefix.txt' file to index sample fastq files.
- Run 'create.map.pl'

LINK:
https://github.com/shu251/QC_steps_V4_tagsequencing

cmd COMMAND
```
git clone https://github.com/shu251/QC_steps_V4_tagsequencing
```
Clone github repo

Create map file (for use in QIIME)

Step 2.

Map files in QIIME serve to de-multiplex reads (based on barcodes and indices) and are used in 'split_library' QIIME step. Here, I've split up the V4 forward primer to fulfill the necessary barcode / index space required in the QIIME mapping file. However, we aren't using this feature of the mapping file.

Use this QIIME command to check mapping file:

http://qiime.org/1.6.0/scripts/check_id_map.html

cmd COMMAND
```
#Required the prefix.txt file
./create.map.pl
```

Perl script to QC each fastq file

Step 3.

Steps of script noted here:

(1) Import prefix file, named 'prefix.txt' and starting loop

(2) Remove primers using Trimmomatic

(3) Merge paired end (R1 and R2) reads with a 20 bp overlap

(4) Perform split library command in QIIME, filter sequences with Q score 30 (-q 29)

(5) Length filter: seqlength_cutoff.pl [input.fasta] [min] [max] [output.fasta]

(6) Chimera check with vsearch (uchime) using a reference database

- You will need to acquire reference database that is best for your sample type. I am using the PR2 database. You can download a QIIME formatted version here.

- Alternatively, you can change the command to run vsearch chimera checking de novo. This will take longer.

(7) Repeat and combine QCed reads together

Required components:

- Forward and reverse primer sequences, I'm using V4 Stoeck et al. 2010 primers

- reference database

- seqlength_cutoff.pl (see script abo)

cmd COMMAND

```
#! /usr/bin/perl -w
open INFILE, "prefix.txt";
@prefix = ();
while (<INFILE>)
{ chomp;
  push(@prefix, $_);
}
close INFILE;


#Perl script to automate V4 tag sequence QC process
#last updated 09-29-2017, Sarah K. Hu


for $i(@prefix)
{
#(1) Trim primers with trimmomatic
print "java -
jar /usr/local/bioinf/Trimmomatic-0.32/trimmomatic-0.32.jar PE ",$i,"_L001_R1_001.fastq ",$
i,"_L001_R2_001.fastq ",$i,"_trim_R1_PE.fastq ",$i,"_trim_R1_orphan.fastq ",$i,"_trim_R2_PE
.fastq ",$i,"_trim_R2_orphan.fastq LEADING:10 TRAILING:10 SLIDINGWINDOW:10:30 MINLEN:50 ILL
UMINACLIP:trimPE_V4primer.fasta:2:30:10\n";

#(2) Merge paired end (R1 and R2) reads with a 20 bp overlap
print "join_paired_ends.py -f ",$i,"_trim_R1_PE.fastq -r ",$i,"_trim_R2_PE.fastq -
o out_",$i," -j 20\n";
print "mv out_",$i,"/fastqjoin.join.fastq ",$i,"_trim_merged.fastq\n";

#(3) Perform split library command in QIIME, filter sequences with Q score 30 (-q 29)
print "split_libraries_fastq.py -i ",$i,"_trim_merged.fastq -m ",$i,"_map.txt --
barcode_type 'not-barcoded' --sample_ids ",$i," -q 29 -n 0 -o split_",$i,"\n";
print "mv split_",$i,"/seqs.fna ",$i,"_trim_merged_Q30.fasta\n";
print "mv split_",$i,"/ out_",$i,"/ \n";

#(4) Length filter: seqlength_cutoff.pl [input.fasta] [min] [max] [output.fasta]
print "./seqlength_cutoff.pl ",$i,"_trim_merged_Q30.fasta 150 500 ",$i,"_trim_merged_Q30_le
n.fasta\n";

#(5) Chimera check with vsearch (uchime) using a reference database
print "vsearch --uchime_ref ",$i,"_trim_merged_Q30_len.fasta --
db /galadriel/sarah/PR2/pr2.qiime.fasta --uchimeout ",$i,".uchimeinfo_ref --
chimeras ",$i,".chimeras_ref.fasta --strand plus --
nonchimeras ",$i,"_trim_merged_Q30_len_nc.fasta \n";

#(9) Move excess files (chimeras) to split directories
print "mv ",$i,".chimeras_ref.fasta out_",$i,"/ \n";
print "mv ",$i,".uchimeinfo_ref out_",$i,"/ \n";
print "mv ",$i,"*orphan.fastq out_",$i,"/ \n";

#(10) Combine all reads together
print "cat ",$i,"_trim_merged_Q30_len_nc.fasta >> allseqs_test.fasta\n";

}
```

Perl script to automate running QC steps on paired end sequences. See step 4 directions for required components and programs.

Check all steps

Step 4.

- Run command below.
- Output will generate a fasta file for each step of the QC process. This protocol (and my preferred method) is to perform as much quality checking on each individual sample before combining for downstream OTU clustering (or other analysis). This way, I can track any samples that may be not ideal for downstream analysis.
- **Note:** for our lab, I see between 20-50% sequences lost per sample (from raw reads to post-chimera check). As long as the final sequence count is > 10,000 I keep the sample. Our threshold of at least 10,000 sequences (typically more) is very conservative.
- Once this is done, you can clean up directory by removing any of these intermediate files.
- **Explanation of output**: After running the count_seqs command (below), the output will show you the number of sequences in each step. We started with 800 sequences, initial quality trimming and primer removal/QC removed almost 200 sequences. Merging and additional QC removed 224 sequences, while length filtration removed no additional sequences. Finally, the chimera checking step only removed 69 sequences. We only kept around 40% of the total sequences.

cmd COMMAND

```
count_seqs.py -
i Test01_L001_R1_001.fastq,Test01_L001_R2_001.fastq,Test01_trim_R1_PE.fastq,Test01_trim_R2_
PE.fastq,Test01_trim_merged.fastq,Test01_trim_merged_Q30.fasta,Test01_trim_merged_Q30_len.f
asta,Test01_trim_merged_Q30_len_nc.fasta
```

Example QIIME command to check number of sequences.

OTU-clustering

Step 5.

Depending on the questions you want to ask your data, decide about what type of OTU clustering you want to do (both algorithm and percent similarity).

QIIME is a great resource for tutorials on OTU clustering - http://qiime.org/tutorials/otu_picking.html#running-the-otu-picking-workflows

cmd COMMAND

```
#OTU clustering with uclust, will make a new directory (pick_open)
pick_open_reference_otus.py -i allseqs_test.fasta -o pick_open -m uclust -
r /galadriel/sarah/PR2/pr2.qiime.fasta --suppress_step4 --suppress_taxonomy_assignment
cd pick_open
#assign taxonomy using PR2 database, creates a new directory (uclust_taxonomy)
assign_taxonomy.py -i rep_set.fna -t /galadriel/db/PR2/ids.names.2.txt -
r /galadriel/db/PR2/pr2.qiime.fasta -m uclust -o uclust_taxonomy
#make an OTU table, and convert to txt
make_otu_table.py -i final_otu_map.txt -o V4_tagseq_test.biom -
t uclust_taxonomy/rep_set_tax_assignments.txt
biom convert -i V4_tagseq_test.biom -o V4_tagseq_test.txt --to-tsv --header-key=taxonomy
```

Analysis

Step 6.

For examples of preliminary analyses and figure generation in R see:
https://github.com/shu251/PreliminaryFigures_V4_tagseq