Detecting 16S rRNA Gene Fragments from a Metagenome to Assemble Full-Length 16S Sequences

Benjamin Tully

Abstract

Goal: Identify 16S rRNA gene metagenomic fragments and create assembled full-length 16S rRNA sequences from fragments.
Note 1 - As is this case with most bioinformatic processes, this is one of many possible methods, but has had successful results in the past
Note 2 - This method is best applied to sequences that have been subjected to quality control as in https://www.protocols.io/view/Basic-Illumina-Sequence-Quality-Control-d4e8td

Citation: Benjamin Tully Detecting 16S rRNA Gene Fragments from a Metagenome to Assemble Full-Length 16S Sequences. **protocols.io**
dx.doi.org/10.17504/protocols.io.d7u9nv
Published: 23 Nov 2015

Before start

There are many tools available for this process - this is one example that has been used effectively
Required software:
IDBA-UD v. 1.1.1 - http://i.cs.hku.hk/~alse/hkubrg/projects/idba_ud/
Meta-RNA - http://weizhong-lab.ucsd.edu/meta_rna/
HMMER v3.1b2 - http://hmmer.janelia.org/
MetaRNA_to_FastQ.py - https://github.com/bjtully/BioData/tree/master/Various_Tools
Biopython - http://biopython.org/DIST/docs/tutorial/Tutorial.html#htoc4
EMIRGE - https://github.com/csmiller/EMIRGE
Cython
pysam
scipy/numpy
usearch (www.drive5.com/usearch/ -- tested with usearch version 6.0.203)
samtools (http://samtools.sourceforge.net/ -- tested with verison 0.1.18)
bowtie (http://bowtie-bio.sourceforge.net/index.shtml -- tested with version 0.12.7 and 0.12.8)

Protocol

Step 1.

Convert quality trimmed FASTQ metagenomic sequences to a FASTA format
Assumes that sequences are paired-end
Perform conversion to FASTA.
fq2fa is available as part of IDBA-UD

SOFTWARE PACKAGE (Unix/Linux)

IDBA-UD, 1.1.1

Peng Yu

cmd COMMAND (Unix/Linux)

```
fq2fa --merge --
filter INFILE_NAME.R1.001_paired.fastq INFILE_NAME.R2.001_paired.fastq INFILE_NAME.001.merg
ed.fasta
```

--merge = interweave paired-end sequences in two separate FASTQ files, new file will have the format of R1, followed by R2, etc. --filter = remove sequences containing Ns

Step 2.

Detect small subunit (SSU) rRNA gene fragments using command-line based Meta-RNA

There have been several updates to this program - this example utilizes the version accessible following the link associated with the file 'readme_H3.txt (old file)' and 'Download here (old file)'

SOFTWARE PACKAGE (Unix/Linux)

Meta-RNA, HMMER3.0b3

Ying Huang

http://weizhong-lab.ucsd.edu/meta_rna/rRNA_hmm3.tar.gz

cmd COMMAND (Unix/Linux)

```
rna_hmm3.py -i INFILE_NAME.001.merged.fasta -
L /directory/location/of/HMM/files/rna_hmm3/HMM3/ -o OUTFILE_NAME.001_predictedRNAs -
m ssu -e .0000000001 -p 36
```

-L = provide the directory address of the downloaded, pre-computed HMM alignment models for both SSU and large subunit rRNA genes (part of the Meta-RNA tarball) -o = set name of output file -m = sets target as SSU fragments only -e = E-value cutoff used by HMMER (1 x 10^-10) -p = number of available CPUs

Step 3.

Use the script MetaRNA_to_FastQ.py to use the output table created by Meta-RNA as a guide for trimming the original quality trimmed FASTQ files.

MetaRNA_to_FastQ.py is only confirmed to work with the version of Meta-RNA described above Requires Biopython

SOFTWARE PACKAGE (Linux/Unix)

MetaRNA_to_FastQ.py

Benjamin Tully

cmd COMMAND (Linux/Unix)

```
MetaRNA_to_FastQ.py -r OUTFILE_NAME.001_predictedRNAs -q INFILE_NAME.R1.001_paired.fastq -
o OUTFILE_PREFIX1.001
```

-r = Meta-RNA table output -q = original FASTQ file searched by Meta-RNA -o = a prefix for the outfile, the final file will have '.metagenome16S.fastq' as the suffix

Step 4.

Repeat Step 3 for R2 reads (and any other sets of reads from the same sample)

SOFTWARE PACKAGE (Linux/Unix)

MetaRNA_to_FastQ.py

Benjamin Tully

cmd COMMAND

```
MetaRNA_to_FastQ.py -r OUTFILE_NAME.001_predictedRNAs -q INFILE_NAME.R2.001_paired.fastq -
o OUTFILE_PREFIX2.001
```

Step 5.

Combine output 'metagenome16S.fastq' files as desired

Options: (1) Pairs of 16S rRNA fragments, or (2) All 16S rRNA fragments from a single sample

cmd COMMAND (Linux/Unix)

```
cat OUTFILE_PREFIX1.001.metagenome16S.fastq OUTFILE_PREFIX2.001.metagenome16S.fastq > OUTFILE.001.metagenome16S.fastq
```

Step 6.

If using EMIRGE for the first time

Set-up EMIRGE dependencies:
Cython
pysam
scipy/numpy
usearch
samtools
bowtie

SOFTWARE PACKAGE (Linux/Unix)

EMIRGE

Chris Miller

Step 7.

If using EMIRGE for the first time

EMIRGE requires a reference database. SILVA (Ref111) can be downloaded using the script provided with EMIRGE:
emirge_download_candidate_db.py

COMMAND (Linux/Unix)

```
emirge_download_candidate_db.py
```

Step 8.

If using EMIRGE for the first time

Reference database must be 'corrected' using the script provided with EMIRGE:
fix_nonstandard_chars.py

COMMAND (Linux/Unix)

```
fix_nonstandard_chars.py < input.fasta > output.fasta
```

Step 9.

If using EMIRGE for the first time

The Reference database must be indexed using bowtie

COMMAND (Linux/Unix)

```
bowtie-build SSU_candidate_db.fasta SSU_candidate_db_btindex
```

Step 10.

Assemble full-length 16S rRNA sequences using EMIRGE, using emirge_amplicon.py, as the input file is exclusively 16S rRNA fragments, not a full metagenome

SOFTWARE PACKAGE (Linux/Unix)

EMIRGE

Chris Miller

COMMAND (Linux/Unix)

```
emirge_amplicon.py ./emirgeWorkingDir -1 OUTFILE_NAME_all.metagenome16S.fastq -f /directory/location/of/EMIRGE/SSURef_111_candidate_db.fasta -b /directoy/location/of/EMIRGE/SSURef_111_candidate_db_btindex -l 260 -i 246 -s 88 -a 32 --phred33
```

./emirgeWorkingDir = creates a working directory for EMIRGE command, if command needs to be re-run, must delete this directory OR change the directory name -1 = input FASTQ file containing all metagenome16S.fastq sequences. It is possible to input paired-end reads files, requires addition of -2 option -f = location of reference FASTA file -b = location of reference FASTA bowtie index file -l = max length of sequences -i = insert size between sequences (required even if only using -1 option) -s = standard deviation of insert sequence sizes -a = number of available CPUs --phred33

= utilizes 33 based Phred quality, standard output on current Illumina sequences

Step 11.

Move to the 'emirgeWorkingDir' and create the final FASTA file using the script provided with EMIRGE: emirge_rename_fasta.py

cmd COMMAND (Linux/Unix)

```
emirge_rename_fasta.py iter.40 > FINAL_16S.fasta
```

Step 12.

Optional step: Assign taxonomy using mothur

Requires: Reference alignment for align.seqs and reference FASTA + taxonomy files for classify.seqs

SOFTWARE PACKAGE (All)

mothur, 1.36.1

Pat Schloss

cmd COMMAND (All)

```
align.seqs(fasta=INPUT.16S.fasta, reference=reference.alignment.fasta, flip=T, processors=6
4)
remove.seqs(accnos=INPUT.16S.flip.accnos, fasta=INPUT.16S.align)
classify.seqs(fasta=INPUT.16S.pick.align, template=reference.database.fasta, taxonomy=refer
ence.database.tax, cutoff=80, iters=1000, processors=64)
```

Run as 3 separate commands Align to reference alignment Remove sequences without alignments Classify sequences using the reference database flip = T - looks for matches in both forward and reverse directions processors = available number of CPUs cutoff = level cutoff for a taxonomic assignment iters = number of comparisons performed to ensure correct assignment