Environmental Toxicology Laboratory
Toxicology Centre
University of Saskatchewan

STANDARD OPERATING PROCEDURE

UofS-ETL-EDNA-34

Bioinformatics pipeline for metabarcoding

Version 1, July 2019

Yuwei Xie, Ph.D. and John P. Giesy, Ph.D., FRSC, FSETAC

Supported through:
Toxicology Centre and
Department of Veterinary Biomedical Sciences

Correspondence to:
Environmental Toxicology Laboratory
Toxicology Centre
44 Campus Drive,
Saskatoon, Saskatchewan, S7N 5B3
Canada

Phone: (306) 966-5062; 966-2096
Facsimile: (306) 966-4796

APPROVAL PAGE

Revisions to an existing SOP, addition of a SOP change form, or preparation of a new SOP must be reviewed, approved, and signed by the following:

Authored By: Yuwei Xie and John P. Giesy Date: 07/10/2019

Supervisor Review By: John P. Giesy Date: 07/20/2019

Reviewed By: _____ Date: _____
(QA Coordinator)

DEFINITIONS AND ACRONYMS

ETL	Environmental Toxicology Laboratory (University of Saskatchewan)
DQO	Data Quality Objective
QA	Quality Assurance
QAPP	Quality Assurance Project Plan
SOP	Standard Operating Procedure
GWF	Global Water Futures
eDNA	Environmental DNA
PCR	Polymerase chain reaction
NGS	Next-generation sequencing

TABLE OF CONTENTS

1.0 PURPOSE

This SOP is developed to provide a bioinformatics pipeline to analyze the NGS data for eDNA metabarcoding. This SOP is compatible with the pair-end sequencing data.

2.0 SCOPE AND APPLICATION

This SOP applies to the ETL for eDNA metabarcoding from the Global Water Futures (GWF) program titled "Next generation solutions to ensure healthy water resources for future generations" (eDNA project).

3.0 EQUIPMENT AND SOFTWARE

- Personal laptop or working station or server
- Linux operation system
- Minicoda3
- Python 2.7
- Python 3.5
- Biopython
- QIIME version 1
- QIIME version 2
- Trimmomatic
- FastQC
- Fastx-toolkit
- Usearch V11

4.0 METHOD, PROCEDURES, AND REQUIREMENTS

The following procedures describe methods for bioinformatics pipeline of eDNA metabarcoding in a stepwise fashion and explain the reasoning behind the techniques. The computing environment is under Ubuntu 18.04 LTS OS. Linux basic are not included in this SOP. Personnel should get training and familiar with Linux command before data processing. MiniConda3 is used to install and maintain software for bioinformatics. Usearch is needed download from its website (www.drive5.com). Some functions of Usearch can be replaced by Vsearch. Running command line is in gray background. All the commands should be run in the terminal. The output results of bioinformatics will be used for ecological statistics.

4.1 **Setup computing environment**

4.1.1 Install Miniconda3

```
wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh
bash Miniconda3-latest-Linux-x86_64.sh
source ./.bashrc
```

4.1.2 Update Miniconda3

```
conda update conda
```

4.1.3 Install essential packages for bioinformatics

```
conda install wget biopython
conda install -c bioconda trimmomatic fastqc
conda install -c biobuilds fastx-toolkit
```

4.1.4 Install Qiime1

```
# Install Qiime1 through conda
conda create -n qiime1 python=2.7 qiime matplotlib=1.4.3 mock nose -c bioconda
# Activate qiime1 environment
source activate qiime1
# Present qiime configuration
print_qiime_config.py -t
# Deactivate qiime1 environment
source deactivate
```

4.1.5 Install Qiime2

```
# Install Qiime2 2019
wget https://data.qiime2.org/distro/core/qiime2-2019.4-py36-linux-conda.yml
conda env create -n qiime2-2019.4 --file qiime2-2019.4-py36-linux-conda.yml
# OPTIONAL CLEANUP
rm qiime2-2019.4-py36-linux-conda.yml
# Activate qiime2 environment
source activate qiime2-2019.4
# Deactivate qiime2 environment
source deactivate
```

4.2 **Quality control and pre-clean the raw sequencing data**

4.2.1 Quality control

```
# setup your working directory with fastq.gz files
cd /your/working/path/with/your/sequencing/data/
gunzip ./*.fastq.gz
#Quality check of the fastq files
```

```
mkdir -p fastq_info

for fq in *.fastq
do
  usearch11 -fastx_info $fq -output ./fastq_info/$fq
  fastx_quality_stats -i $fq -o ./fastq_info/${fq/.fastq/_QC_summary.txt}
  fastq_quality_boxplot_graph.sh -i ./fastq_info/${fq/.fastq/_QC_summary.txt} \
  -t QC_barplot -o ./fastq_info/${fq/.fastq/_QC.tiff}
  fastx_nucleotide_distribution_graph.sh -i ./fastq_info/${fq/.fastq/_QC_summary.txt} \
  -t ${fq/.fastq/_NA_distribution.tiff} -o ./fastq_info/${fq/.fastq/_NA_distribution.tiff}
  fastqc $fq -t 8 -o fastq_info
done

# get a summary of expected error (EE) distribution, length distribution,
# number of reads...) to check that all of the files are similar
grep "^EE" ./fastq_info/*
```

4.2.2 Trim sequencing adaptors and remove low quality data

```
#Adaptor trimming tool for PE reads from Illumina sequencers
#Trim sequences which are shorter than 80 bp. Caution, change it based on the
sequencing chemistry kit and targets
for R1 in *R1_001.fastq
do
  R2=${R1/R1_001.fastq/R2_001.fastq}
  trimmomatic PE -threads 32 -phred33 -validatePairs $R1 $R2 \
  ${R1/.fastq/_paried.fastq} ${R1/.fastq/_unparied.fastq} \
  ${R2/.fastq/_paried.fastq} ${R2/.fastq/_unparied.fastq} \
  ILLUMINACLIP:$HOME/bin/TruSeq3-PE.fa:2:30:10 LEADING:3 TRAILING:3
SLIDINGWINDOW:4:15 MINLEN:80
done
```

4.2.3 Merge pair-end reads

```
# Assemble pair end reads
for fq in *"_R1_001"*".fastq"
do
  usearch11 -fastq_mergepairs $fq -fastqout ${fq/.fastq/_short_merged.fastq}
done
```

4.2.4 Filtering low quality reads

```
# Discard reads which probably have errors (quality filtering)
# Optional MaxE: 1.0, 2.0, 3.0. The greater of MaxE, the more lossen of quality filtering.
for fq in *merged.fastq
do
```

```
  usearch11 -fastq_filter $fq -fastq_maxee 3.0 -fastqout
${fq/merged.fastq/filtered}"_"$MaxE".fastq"
done
```

4.2.5 Prepare fasta, qual, and mapping files for demultiplexing

```
ln -s
YOUR_TRIMMED_CLEANED_MERGED_FASTQ_FILE_FROM4.2.5 ./raw.fq
#Enter QIIME environment
source activate qiime1
#reverse_complement the filtered QC fastq file
usearch11 -fastx_revcomp raw.fq -label_suffix _RC -fastqout raw_rc.fq
#merge fq files
cat raw.fq raw_rc.fq > QCed.fq
#Convert fastq file to fasta and qual file
convert_fastaqual_fastq.py -c fastq_to_fastaqual -f QCed.fq -o ./
#Prepare Mapping files
#Instruction online: http://qiime.org/tutorials/tutorial.html
#Check the Maping files
for MAP in MAP_*.txt
do
 validate_mapping_file.py -B -s -m $MAP
done
```

4.2.6 Demultiplex reads for each sample – 1st round: based on tag sequences of reverse primer

```
for fa in *.fna
do
 qa=${fa/.fna/.qual}
 output="split_"${fa/.fna/_rev}
 split_libraries.py -m MAP_CyanoB1664R_corrected.txt -f $fa -q $qa -o $output -b
variable_length \
 -H 7 --disable_primers --record_qual_scores
 convert_fastaqual_fastq.py -f $output/seqs.fna -q $output/seqs_filtered.qual -F -m -o
$output
 rm -rf $output/seqs_filtered.qual
 rm -rf $output/seqs.fna
done
```

4.2.7 Demultiplex reads for each sample 2nt round: based on tag sequences of forward primer

```
# Prepare input files for tagged forward primer demultiplexing
cd split_QCed_rev
#USING PYTHON SCRIPT to retrieve the original read name
```

```
# RetrieveNameAfterQiimeSplitLibrary_1.py
# Run this script under python environment

# Prepare fasta and qual files for forward primer demultiplexing
for fq in *.fq
do
  usearch11 -fastx_revcomp $fq -fastqout ${fq/.fq/_RC.fq}
  convert_fastaqual_fastq.py -c fastq_to_fastaqual -o ./ -f ${fq/.fq/_RC.fq}
done

# Split library based on tags of forward primers
for lib in LIST of OUTPUT FILES (NAME of REVERSE PRIMER)
do
  map="MAP_"$lib"_ForPrimer_corrected.txt"
  fa="RetrieveName_"$lib"_RC.fna"
  qual="RetrieveName_"$lib"_RC.qual"
  output="split_"$lib
  split_libraries.py -H 7 --record_qual_scores -b variable_length -z truncate_only -
m ../$map -f $fa -q $qual -o $output
  convert_fastaqual_fastq.py -f $output/seqs.fna -q $output/seqs_filtered.qual -F -m -o
$output
  rm -rf $output/seqs_filtered.qual
  rm -rf $output/seqs.fna
done
# [Opional] Clean intermedia results
rm ./*.qual
rm ./*.fna
rm ./*RC.fq
rm ./RetrieveName_*.fq

#Retrive read ID from second round split_library
mkdir split_fqs
mv split_*/seqs_*.fastq split_fqs/
cd split_fqs/

# RetrieveNameAfterQiimeSplitLibrary_2.py
# Run this script under python environment
# sample.fq and sample_id.txt
#Correct the id.txt removing "_RC"
for id in *_id.txt
do
  sed 's/_RC//g' $id > ${id/.txt/_corrected.txt}
done
```

```
rm ./*_id.txt

# Clean name of Illumina output fastq file
cd ..
# python script: RetrieveNameAfterQiimeSplitLibrary_3.py
# Run this script under python environment

cd split_QCed_rev/split_fqs/
# Retrive reads from PE sequencing output
mkdir dada2

for fq in *.fq
do
  name=${fq/.fq/_id_corrected.txt}
  R1=${fq/.fq/_R1_001.fq}
  R2=${fq/.fq/_R2_001.fq}
  usearch11 -fastx_getseqs ../*_R1_001.fq -labels $name -fastqout ./dada2/$R1
  usearch11 -fastx_getseqs ../*_R2_001.fq -labels $name -fastqout ./dada2/$R2
done
# Clean intermedia data
rm ./*_id_corrected.txt
rm ./seqs_*.fastq

# Prepare fq files for Uparse OTU clustering approach
mkdir uparse
# Discard reads which probably have errors (quality filtering)
for fq in *.fq
do
  #1.0 2.0
  for MaxE in 3.0
  do
    usearch11 -fastq_filter $fq -relabel @ -fastq_maxee $MaxE -
fastqout ./uparse/${fq/.fq/}".fastq"
  done
done

# Prepare fq files for Uparse OTU clustering approach
mkdir Q2
mv ./*.fq ./Q2/

# Clean output folders
cd ..
cat split_*/split_library_log.txt > split_forward_primer_library_log.txt
```

```
rm -rf split_*R*

cd ..
gzip ./*.fastq
gzip ./*.fq
```

4.2.8 OTU clustering approach: Uparse and zout pipeline

```
cd ./uparse/
# Rename fastq files
for fq in *.fastq
do
  usearch11 -fastq_filter $fq -relabel @ -fastq_maxee 3.0 -fastqout
${fq/.fastq/_rename.fastq}
done
#pool targeted fastq files into one fastq file and convert it into fasta formate
cat ./*rename.fastq > combined.fq
# Convert fastq to fasta
usearch11 -fastq_filter combined.fq -fastq_maxee 3.0 -fastaout combined.fna
gzip combined.fq
#Prepare Mapping file for OTU analyses
#Edit "MAP.txt" visually

# otu clustering using uparse pipeline
# Find unique sequences and abundances
usearch11 -fastx_uniques combined.fna -sizeout -relabel Uniq -fastaout uniques.fa
# Create 97% OTUs -minsize 2
usearch11 -cluster_otus uniques.fa -uparseout uparse.txt -relabel Otu -otus otus.fa -
minsize 2
# Create OTU table for 97% OTUs
usearch11 -otutab combined.fna -otus otus.fa -strand plus -otutabout otutab.txt
# Output of uparse table to Qiime input file
biom convert -i otutab.txt -o otus.biom --table-type="OTU table" --to-json
# Assign taxonomy using rdp method
assign_taxonomy.py -m rdp --rdp_max_memory=12000 -i otus.fa \
-r reference_database_fasta_file -t reference_database_taxonomy_file -o otus_rdp_gg
#Add taxonomic annotation to the biom file
biom add-metadata --sc-separated taxonomy --observation-header OTUID,taxonomy \
--observation-metadata-fp PATH/otus_tax_assignments.txt \
-i otus.biom -o otus_tax.biom
biom summarize-table -i otus_tax.biom -o otus_tax_summary.txt
biom convert -i otus_tax.biom -o otus_tax.txt --to-tsv --header-key taxonomy
#Phylogenetic tree construction of OTUs sequences
align_seqs.py -i otus.fa -o otus_align
```

```
# for OTUs with a aligned template
filter_alignment.py -i otus_align/otus_aligned.fasta
# for OTUs without a aligned template
filter_alignment.py -i otus_align/otus_aligned.fasta -s
make_phylogeny.py -i otus_aligned_pfiltered.fasta -o otus_tree.tre
summarize_taxa.py -i otus_tax.biom -o clean_tax_summ_absolute/ -m MAP.txt -L
2,3,4,5,6,7 -a
summarize_taxa.py -i otus_tax.biom -o clean_tax_summ_relative/ -m MAP.txt -L
2,3,4,5,6,7
summarize_taxa_through_plots.py -i otus_tax.biom -o compostion_plot/ -m MAP.txt
```

4.2.9 OTU clustering approach: DADA2 pipeline

```
# import qiime2 environment
source activate qiime2-2019.4
# Set working directory
cd ../Q2
# metadata file creating following instruction online:
https://docs.qiime2.org/2019.4/tutorials/moving-pictures/#sample-metadata
qiime metadata tabulate --m-input-file sample-metadata.tsv --o-visualization sample-
metadata.qzv

# Import trimmed cleaned paired-end fastq files
qiime tools import --type 'SampleData[PairedEndSequencesWithQuality]' --input-path
fq_manifest.csv --input-format PairedEndFastqManifestPhred33 --output-path demux-
paired-end.qza
qiime demux summarize --i-data demux-paired-end.qza --o-visualization demux-paired-
end.qzv
qiime tools view demux-paired-end.qzv

# Process data using dada2 pipeline
mkdir dada2
cd dada2
# denoise reads
# Parameters (--p-trim-left-f, --p-trim-left-r, --p-trunc-len-f, --p-trunc-len-r) should be
optimized based on primer sequences and length of PCR produts
qiime dada2 denoise-paired --i-demultiplexed-seqs ../demux-paired-end.qza --p-trim-left-
f ??? --p-trim-left-r ??? --p-trunc-len-f ??? --p-trunc-len-r ??? --p-n-threads 8 --o-table
table.qza --o-representative-sequences rep-seqs.qza --o-denoising-stats stats-dada2.qza
#Summarize the denoise data
qiime metadata tabulate --m-input-file stats-dada2.qza --o-visualization stats-dada2.qzv
#FeatureTable and FeatureData summaries
qiime feature-table tabulate-seqs --i-data rep-seqs.qza --o-visualization rep-seqs.qzv
```

```
qiime feature-table summarize --i-table table.qza --o-visualization table.qzv --m-sample-metadata-file ../sample-metadata.tsv

#Generate a tree for phylogenetic diversity analyses
qiime phylogeny align-to-tree-mafft-fasttree --i-sequences rep-seqs.qza --o-alignment aligned-rep-seqs.qza \
--o-masked-alignment masked-aligned-rep-seqs.qza --o-tree unrooted-tree.qza --o-rooted-tree rooted-tree.qza

#Alpha and beta diversity analysis
qiime diversity core-metrics-phylogenetic --i-phylogeny rooted-tree.qza --i-table table.qza \
--p-sampling-depth 1109 --m-metadata-file ../sample-metadata.tsv --output-dir core-metrics-results

#alpha-rarefaction
qiime diversity alpha-rarefaction --i-table table.qza --i-phylogeny rooted-tree.qza --p-max-depth 4000 --m-metadata-file ../sample-metadata.tsv --o-visualization alpha-rarefaction.qzv
```

5.0 RECORDS, DOCUMENTATION, AND QC REQUIREMENTS

Raw data, codes and processed results should be maintained following the Data Management Plan of eDNA project.

6.0 RESPONSIBILITIES

Project Director — Will oversee and approve all project activities.

Project Manager — Will oversee and approve all project activities; review QA reports; approve final project QA needs; authorize necessary actions and adjustments to accomplish program QA objectives; and act as liaison between agencies, field staff.

Quality Assurance (QA) Manager — Will oversee all QA activities to ensure compliance with contract specifications; initiate audits on work completed by project personnel and subcontractors, including analytical laboratories and independent data validation contractors; review program QA activities, quality problems, and quality-related requests. In response to field and analytical findings, this person will approve the corrective actions. This person will report quality non-conformances to the Project Manager and review all pertinent portions of the deliverables before they are transmitted to ensure conformance with QA/QC procedures and quality work product.

Data Manager — Will oversee data management for this project. This person is responsible for the structure, organization, format, implementation, and operation of the project database.

Field Team Leader — Will oversee field activities and supervise the field crews. This person will ensure that proper sample collection, preservation, storage, transport, and COC QC procedures are followed. This person will inform the Project QA Manager when field problems occur, and will communicate and document corrective actions taken. The Field Team Leader will discuss field activities with the Project Manager.

Laboratory Project Manager — The laboratory project manager for this project is the person responsible for assuring that the analysis of all samples is performed in accordance with the QAPP and the laboratory's quality assurance manual. In addition, the Laboratory Project Manager performs the final laboratory review of project data packages for completeness and compliance with project requirements.

7.0 REFERENCES

Bolyen E, et al.. 2018. QIIME 2: Reproducible, interactive, scalable, and extensible microbiome data science. PeerJ Preprints 6:e27295v2 https://doi.org/10.7287/peerj.preprints.27295v2.
J Gregory Caporaso, et al.. Nature Methods, 2010; doi:10.1038/nmeth.f.303.
Edgar, R.C. (2013) UPARSE: Highly accurate OTU sequences from microbial amplicon reads, Nature Methods [Pubmed:23955772, dx.doi.org/10.1038/nmeth.2604].
Benjamin J Callahan, et al.. DADA2: High-resolution sample inference from Illumina amplicon data. Nature Methods volume 13, pages 581–583 (2016).
Bolger, A. M., Lohse, M., & Usadel, B. (2014). Trimmomatic: A flexible trimmer for Illumina Sequence Data. Bioinformatics, btu170.
Andrews S. (2010). FastQC: a quality control tool for high throughput sequence data. Available online at: http://www.bioinformatics.babraham.ac.uk/projects/fastqc.