PIMBA: a PIpeline for MetaBarcoding Analysis

Renato R. M. Oliveira1,2\*[0000-0002-0111-8209], Raíssa Silva1[0000-0002-8539-9780], Gisele L. Nunes1[0000-0003-3994-5135] and Guilherme Oliveira1[0000-0003-0054-3438]

1 Instituto Tecnológico Vale, Belém, Pará, Brazil;

2 Programa de Pós-Graduação em Bioinformática, Instituto de Ciências Biológicas, Universidade Federal de Minas Gerais, Belo Horizonte, Minas Gerais, Brazil

\*To whom correspondence should be addressed [renato.renison@gmail.com](mailto:renato.renison@gmail.com); guilherme.oliveira@itv.org

We chose to benchmark PIMBA, PEMA, QIIME2, Mothur, Obitools, mBRAVE with four datasets: 16S, Fungal ITS, Metazoan COI and Plant ITS.

A few informations regarding the datasets can be seen in the Table below.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Gene | SRA | Total reads | Total bases (Mb) | Read length (bp) |
| 16S | SRR3163904  SRR3163905  SRR3163906 | 895,113 | 471.3 | 2x300 |
| Fungal ITS | SRR5838515  SRR5838516  SRR5838522 | 162,841 | 81.5 | 2x250 |
| Metazoan COI | ERR2181459  ERR2181468  ERR2181466 | 228,019 | 113.8 | 2x300 |

# 16S

We downloaded the dataset from Gohl et al. (2016) (10.1038/nbt.3601).

>fastq-dump --split-3 --skip-technical SRR3163904

For preprocessing, we applied the same quality and length parameters as PEMA:

#Usage: ./pimba\_prepare.sh illumina <rawdata\_dir> <output\_reads> <num\_threads> <adapters.txt> <min\_length> <min\_phred>

#<rawdata\_dir> = path with all the R1 and R2 reads file;

#<output\_reads> = name for the output file;

#<num\_threads> = number of threads;

#<adapters.txt> = txt list file with all adapters and primers used for sequencing;

#<min\_lenght> = The minimum lenght of the read after quality treatment;

#<min\_phred> = Minimum PHRED score of a read after quality treatment;

>pimba\_prepare.sh illumina rawdata\_dir/ AllSamples\_16S\_hqdata 20 adapters.txt 100 20

A total of 810,981 high-quality reads were outputted, with 236.6 Mb

We executed the both OTU clustering and ASV inference, with SILVA (same as PEMA) and NCBI-genbank, using the same clustering similarity and target length as PEMA.

#Usage: ./pimba\_run.sh -i <input\_reads> -o <output\_dir> -w <approach> -s <otu\_similarity> -a <assign\_similarity> -c <coverage> -l <otu\_length> -h <hits\_per\_subject> -g <marker\_gene> -t <num\_threads> -e <E-value> -d <databases.txt>

#-i <input\_reads> = FASTA file with reads output generated by pimba\_prepare.sh

#-o <output\_dir> = Directory where the results will be stored.

#-w <approach> = Analysis strategy to be used. It can be 'otu' or 'asv'. If 'otu', pimba uses vsearch. If 'asv', pimba uses swarm. Default: 'otu'

#-s <otu\_similarity> = Percentage of similarity that will be used in the otu clustering. Default is 0.97

#-a <assign\_similarity> = Percentage of similarity that will be used in the taxonomy assignment. Default is 0.9

#-c <coverage> = minimum converage for the alignment. Default is 0.9

#-l <otu\_length> = Length to trim the reads. If 0, then no reads are trimmed.

#-h <hits\_per\_subject> = if 1, choose the best hit. If > 1, choose by majority. Default is 1

#-g <marker\_gene> = Marker gene and Database of the analisys. It can be: (16S-SILVA, 16S-GREENGENES, 16S-RDP, 16S-NCBI, ITS-FUNGI-NCBI, ITS-FUNGI-UNITE, ITS-PLANTS-NCBI, COI-NCBI, COI-BOLD)

#-t <num\_threads> = Number or threads to use in the blast step. Default is 1.

#-e <E-value> = Expected value used by blast. Dafault is 10.

#-d <databases\_file.txt> = File with the databases path. Default is /bio/pimba\_metabarcoding/databases.txt

#-x <lulu> = if set as 'lulu', PIMBA will discard erroneous OTUs or ASVs with LULU. Default is not to use LULU.

>pimba\_run.sh -i 16S\_hqdata.fasta -o run\_OTU\_SILVA -w otu -s 0.97 -a 0.9 -l 200 -g 16S-SILVA -t 1 -d ../databases.txt

>pimba\_run.sh -i 16S\_hqdata.fasta -o run\_ASV\_SILVA -w asv -s 0.97 -a 0.9 -l 200 -g 16S-SILVA -t 1 -d ../databases.txt

>pimba\_run.sh -i 16S\_hqdata.fasta -o run\_OTU\_NCBI -w otu -s 0.97 -a 0.9 -l 200 -g 16S-NCBI -t 1 -d ../databases.txt

>pimba\_run.sh -i 16S\_hqdata.fasta -o run\_ASV\_NCBI -w asv -s 0.97 -a 0.9 -l 200 -g 16S-NCBI -t 1 -d ../databases.txt

# Fungal ITS

>pimba\_run.sh -i ITS\_hqdata.fasta -o run\_OTU\_FUNGI-UNITE -w otu -s 0.97 -a 0.9 -l 200 -g ITS-FUNGI-UNITE -t 1 -d ../databases.txt

>pimba\_run.sh -i ITS\_hqdata.fasta -o run\_ASV\_FUNGI-UNITE -w asv -s 0.97 -a 0.9 -l 200 -g ITS-FUNGI-UNITE -t 1 -d ../databases.txt

>pimba\_run.sh -i ITS\_hqdata.fasta -o run\_OTU\_FUNGI-NCBI -w otu -s 0.97 -a 0.9 -l 200 -g ITS-FUNGI-NCBI -t 1 -d ../databases.txt

>pimba\_run.sh -i ITS\_hqdata.fasta -o run\_ASV\_FUNGI-NCBI -w asv -s 0.97 -a 0.9 -l 200 -g ITS-FUNGI-NCBI -t 1 -d ../databases.txt

# Metazoan COI

>pimba\_run.sh -i COI\_hqdata.fasta -o run\_OTU\_NCBI -w otu -s 0.97 -a 0.9 -l 200 -g COI-NCBI -t 1 -d ../databases.txt

>pimba\_run.sh -i COI\_hqdata.fasta -o run\_ASV\_NCBI -w asv -s 0.97 -a 0.9 -l 200 -g COI-NCBI -t 1 -d ../databases.txt

# PIMBA’s Manual

## How to install?

To run PIMBA, you just need to have docker (see <https://docs.docker.com>) installed in your operational system.

sudo apt-get install docker.io

And that's all! Now you are able to run PIMBA on your data!

## Prepare your data (pimba\_prepare.sh)

The first step to run PIMBA is to prepare your data. PIMBA can be used with paired-end or single-end reads (the latter being single-index or dual-index). The output will be a fasta file that can be used in the next step.

### paired-end reads:

Please, place all your forward and reverse reads in one directory and make sure that forward reads contain "R1" and reverse reads contain "R2" in the file's name.

./pimba\_prepare.sh illumina <rawdata\_dir> <output\_reads> <num\_threads> <adapters.txt> <min\_length> <min\_phred>

<rawdata\_dir> = path with all the R1 and R2 reads file;  
<output\_reads> = name for the output file;  
<num\_threads> = number of threads;  
<adapters.txt> = tab separated 2-column file with all adapters and primers used for sequencing;  
<min\_lenght> = The minimum lenght of the read after quality treatment;  
<min\_phred> = Minimum PHRED score of a read after quality treatment.\

Example:

./pimba\_prepare.sh ilumina rawdata/ AllSamples 24 adapters.txt 100 20

### single-end reads with dual-index:

In case your single-end reads have been multiplexed with dual-index, use the following command:

./pimba\_prepare.sh iontorrent-dualindex <rawdata.fastq> <barcodes.txt> <barcodes\_reverse.txt> <barcodes.fasta> <barcodes\_for\_dir> <Primer\_forward> <Primer\_reverse> <num\_threads> <output\_name> <min\_length> <min\_phred>

<rawdata.fastq> = single file with all the reads to demultiplex;  
<barcodes.txt> = barcodes used as index in the 3' of the fragment;  
<barcodes\_reverse.txt> = reverse complement of <barcodes.txt>;  
<barcodes.fasta> = fasta file for the <barcodes.txt>;  
<barcodes\_for\_dir> = path to all the barcodes.fasta and barcodes.txt used in the 5' of the fragment. Each 3' barcode must have a fasta and txt file with all the associated 5' barcodes;  
<Primer\_forward> = sequence of the forward primer;  
<Primer\_reverse> = sequence of the reverse primer;  
<num\_threads> = number of threads;  
<output\_name> = name for the output fastq file;  
<min\_lenght> = The minimum lenght of the read after quality treatment;  
<min\_phred> = Minimum PHRED score of a read after quality treatment.\

Example:

./pimba\_prepare.sh iontorrent-dualindex rawdata\_chip-3-4.fastq barcodes.txt barcodes\_reverse.txt barcodes.fasta barcode\_for/ TCCACTAATCACAAAGANATNGGNAC AGAAAATCATAATNAANGCNTGNGC 24 AllSamplesCOI.fastq 100 20

### single-end reads with single-index:

In case your single-end reads have been multiplexed with single-index, use the following command:

./pimba\_prepare.sh iontorrent-singleindex <rawdata.fastq> <prefix> <barcodes.txt> <barcodes.fasta> <primer> <num\_threads> <output\_name> <min\_length> <min\_phred>

<rawdata.fastq> = single file with all the reads to demultiplex;  
= name that will precede the barcodes names;  
<barcodes.txt> = barcodes used as index in the 5' of the fragment;  
<barcodes.fasta> = fasta file for the <barcodes.txt>;  
= primer sequence;  
<num\_threads> = number of threads;  
<output\_name> = name for the output fastq file;  
<min\_lenght> = The minimum lenght of the read after quality treatment;  
<min\_phred> = Minimum PHRED score of a read after quality treatment.\

Example:

./pimba\_prepare.sh iontorrent-singleindex SN1-45.fastq SN1-45-ITS barcodes.txt barcodes.fasta ATGCGATACTTGGTGTGAAT 24 AllSamples

## Run your metabarcoding analysis (pimba\_run.sh)

The output generated by pimba\_prepare.sh is a fasta file that will be used by pimba\_run.sh in the following command:

./pimba\_run.sh -i <input\_reads> -o <output\_dir> -w <approach> -s <otu\_similarity> -a <assign\_similarity> -c <coverage> -l <otu\_length> -h <hits\_per\_subject> -g <marker\_gene> -t <num\_threads> -e <E-value> -d <databases.txt> -x <run\_lulu>

-i <input\_reads> = FASTA file with reads output generated by pimba\_prepare.sh;  
-o <output\_dir> = Directory where the results will be stored;  
-w = Analysis strategy to be used. It can be 'otu' or 'asv'. If 'otu', pimba uses vsearch. If 'asv', pimba uses swarm. Default: 'otu';  
-s <otu\_similarity> = Percentage of similarity that will be used in the otu clustering. Default is 0.97;  
-a <assign\_similarity> = Percentage of similarity that will be used in the taxonomy assignment. Default is 0.9;  
-c = minimum converage for the alignment. Default is 0.9;  
-l <otu\_length> = Length to trim the reads. If 0, then no reads are trimmed;  
-h <hits\_per\_subject> = if 1, choose the best hit. If > 1, choose by majority. Default is 1;  
-g <marker\_gene> = Marker gene and Database of the analisys. It can be: (16S-SILVA, 16S-GREENGENES, 16S-RDP, 16S-NCBI, ITS-FUNGI-NCBI, ITS-FUNGI-UNITE, ITS-PLANTS-NCBI, COI-NCBI). The path for each database must be configured in the <databases\_file.txt>;  
-t <num\_threads> = Number or threads to use in the blast step. Default is 1;  
-e = Expected value used by blast. Dafault is 10;  
-d <databases\_file.txt> = File with the databases path.\

-x <lulu> = if set as 'lulu', PIMBA will discard erroneous OTUs or ASVs with LULU. Default is not to use LULU.

The <databases\_file.txt> must be properly configured. The fixed databases in the <databases\_file.txt> are the following:

#!/bin/bash

SILVA\_DB\_16S=/your/path/to/Silva\_132\_release/SILVA\_132\_QIIME\_release/

GG\_DB\_16S=/your/path/to/gg\_13\_5\_otus/

RDP\_DB\_16S=/your/path/to/RDP/

TAXDUMP=/your/path/to/taxdump/

ITS\_UNITE\_DB=/your/path/to/sh\_refs\_qiime\_ver8.2/

You can download the databases above in the following links: SILVA for 16S: <https://www.arb-silva.de/no_cache/download/archive/qiime/>   
Greengenes for 16S: ftp://greengenes.microbio.me/greengenes\_release/gg\_13\_5/gg\_13\_8\_otus.tar.gz   
RDP for 16S: <https://sourceforge.net/projects/rdp-classifier/files/RDP_Classifier_TrainingData/RDPClassifier_16S_trainsetNo18_QiimeFormat.zip/download>   
UNITE for fungal ITS: <https://plutof.ut.ee/#/doi/10.15156/BIO/786385> \

For NCBI databases, make sure you download the following files to /your/path/to/taxdump/, using your terminal:

wget ftp://ftp.ncbi.nlm.nih.gov/pub/taxonomy/new\_taxdump/new\_taxdump.tar.gz

Then, uncompress:

tar -xzvf new\_taxdump.tar.gz

If you want to use a personalized database,it is only needed a fasta file with the reference sequences and their identification, and a two-column tax.txt file with the sequence ID and the full taxonomy written for every reference sequence in the fasta file. Put them at the same directory, e.g: /path/to/your/database/. Example of FASTA file:

Text

Description automatically generated

Example of Taxonomy file:



Then, install blastn in your computer and run makeblastdb in your fasta file:

sudo apt-get install ncbi-blast+

makeblastdb -in <your\_fasta.fasta> -dbtype nucl -parse\_seqids

After that, all you need is set the /path/to/your/database/ in the -g parameter when running pimba\_run.sh. Example:

./pimba\_run.sh -i AllSamplesCOI\_chip1234\_good.fasta -o AllSamplesCOI\_98clust90assign -w otu -s 0.98 -a 0.9 -c 0.9 -l 130 -h 1 -g /path/to/your/database/ -t 24 -e 0.1 -d databases.txt

## Plot your results (pimba\_plot.sh)

When finished with pimba\_run.sh, you will be able to generate some basic plots for your results, such as PCoA, rarefaction curves, alpha and beta diversity plots. All your need will be two files that pimba\_run.sh will generate and one metadata file that you will have to provide.

Example of OTU table (otu\_table.txt) generated by pimba\_run.sh:

A picture containing graphical user interface

Description automatically generated

Example of OTU Tax assignment (tax\_assignment.txt):

Graphical user interface

Description automatically generated

Example of Metadata file (metadata.csv):

Text

Description automatically generated with medium confidence

The first columns must always be “SampleID” and the second column must always be “SampleName”

Then, you can run the following command:

./pimba\_plot.sh -t <otu\_table> -a <tax\_assignment> -m <metadata> -g <group\_by>

-t <otu\_table> = OTU table generated by pimba\_run;  
-a <tax\_assignment> = Tax assignment file generated by pimba\_run;  
-m = CSV file with columns "SampleID" and "SampleName", and other attributes related to each sample;  
-g = A column from the metadata that will group the results. E.g, Description. If one does not want to group the results, do not specify it.\

Example:

./pimba\_plot.sh -t unionTriplicatas\_otu\_table.txt -a unionTriplicatas\_otus\_tax\_assignments.txt -m mapping\_file.csv -g Description

The list of plots that pimba\_plot.sh will generate:

alpha\_diversity\_dotplot.svg  
rarefaction\_curve2.svg  
cluster\_dendogram.svg  
phylum\_barplots.svg  
class\_barplots.svg  
order\_barplots.svg  
family\_barplots.svg  
genus\_barplots.svg