# PCAMMS

Pipeline for Classifying and Assembling Microbes from Metagenomic Sequencing

\*\*\*THIS PIPELINE IS STILL IN DEVELOPMENT\*\*\*

Table of Contents

[Description: 2](#_Toc534703666)

[Brief: 2](#_Toc534703667)

[In-depth: 2](#_Toc534703668)

[Metagenomic\_qc.py 2](#_Toc534703669)

[metagenomic.py 3](#_Toc534703670)

[Installation 4](#_Toc534703671)

[Usage: 5](#_Toc534703672)

[metagenomic\_qc.py 5](#_Toc534703673)

[metagenomic.py 5](#_Toc534703674)

[Basic 6](#_Toc534703675)

[metagenomic\_qc.py 6](#_Toc534703676)

[metagenomic.py 6](#_Toc534703677)

[Example: 7](#_Toc534703678)

[With Negative control 7](#_Toc534703679)

[Multiple negative controls 7](#_Toc534703680)

[Using a local contamination knowledge 8](#_Toc534703681)

[Local contamination library 8](#_Toc534703682)

[List of taxon to ignore 9](#_Toc534703683)

[Customising host read removal 9](#_Toc534703684)

[Not removing host 9](#_Toc534703685)

[Customizing host 9](#_Toc534703686)

[Running the pipeline from a sample list 9](#_Toc534703687)

# Description:

## Brief:

This pipeline is for initial analysis of metagenomic samples sequenced on Illumina platforms. It was originally designed for detection of pathogens in sterile site infections sequenced on the Miseq. The pipeline consists of two scripts, the first (metagenomic\_qc.py) prepares the reads for analysis (including merging reads, removing host and performing basic QC). The second (metagenomic.py) performs classification on the reads using CLARK in light mode, then downloads the representative reference file for each species taxon ID estimated to be significant, and performs reference assembly on the unpaired reads against this reference.

This pipeline is intended to be a first pass analysis to narrow the pool of possible microbes in a sample. It has been designed to be high sensitivity but low sensitivity and results should be reviewed and edited using local knowledge.

Input:

Raw paired-end reads from Illumina in a single directory

A directory for processing and storing results

Name of run (no spaces), this will be used to name files within the results directory.

## 

## In-depth:

### Metagenomic\_qc.py

1) Script will check the input files and directory, if the output directory does not exist it will create it. All files will be written to this directory

\*Note processes are paralleled using Nextflow. This will create a directory called 'work' in your current directory where intermediate files will be kept, this directory will be deleted at the end of the script. If there is already a directory called 'work' in your current directory this will be overwritten, so rename this directory before starting the pipeline.

2) Next the script will copy the raw files from the input directory to output directory and rename based either on the sample number and run name provided (default), or rename based on a given sample list. Option available to not rename the files.

Raw files remain unchanged.

3) Host reads will be removed by mapping against the host reference, default is human. These fastq files replace the raw files in the output directory. Host can be changed see below. Option to not map against host.

4) Paired-end reads will be merged using fastq-join. The paired-end reads will be stored in a directory within the project folder

5) Basic Quality control will be performed using Fastqc on merged reads, then produce a combined report using Mulitqc.

6) If negatvie controls are given these will be classified using CLARK-L (optional)

7) A directory called qc\_results will be created which will contain:

\* a html report containing the multiqc results

\* a csv file containing information on the host mapping process (input reads, number of reads mapping to host, % reads mapped to host, and the output reads)

\* a csv containing read information (number of parie-end reads, number of merged reads and the % of reads which were sucessfully merged)

\* html(s) of a Krona plot of the results of classifying the negative (if performed)

### metagenomic.py

1) Optional: Map against the negative control(s) and/or a local contamination library to remove any contaminating reads, mapped reads will be removed.

2) The reads will be classified using CLARK in light mode.

3) The significance of classified hits will then be assessed for each species identified in a sample. Using the average read length, size of the genome and the number of reads classified.

(av\_readlength x number of classified hits)/genome size) if this is greater than 0.0025 (0.25%) then the hit is considered significant.

This balances not missing hits, but also not performing assembly on very low quality hits.

This cut off can be altered in the metagenomic.py if required (line 423)

4) For those hits considered to be potentially significant the representative genome for the taxon ID will be downloaded from RefSeq and stored in a directory PCAMMS/refs.

5) The paired-end reads will then be assembled against this reference, and the number of mapped reads and genome coverage will be calculated.

6) The taxon abundance across the whole run will be calculated, which includes how many samples each taxon was identified in, how many reads were identified and which samples have the highest number of reads. This is designed to help indicate contamination within the run.

7) A directory called 'results' will be created in the output directory and will contain:

* runame.csv:
  + csv file containing basic run information for each sample in the run(number of raw reads, number of reads mapping to host, percentage of reads mapping to host, number of reads not mapping to host , number of reads after paired-end merging, % of reads that were merged , number of reads which were classified, % of reads which were classified)
* runame\_genome\_cov.csv:
  + csv containing the estimated genome coverage based on calculation described above
* runame\_referene\_mapping.csv:
  + csv file with the mapping information for each sample and each microbe identified (organism name, organism genome size, bases in the genome covered in the reference assembly, % of genome covered, number of reads mapped, % of the total reads mapped, taxon ID of the organism, how many reads were classified as that taxon ID, the reference used in the assembly)
* runname\_sample\_ref.csv:
  + contains full path to the reference used
* sample.html :
  + Krona plot of results of classification.

# Installation

Please note building the database requires 90GB of storage. Once built most files are removed and the total required space is 2.2GB

In the directory where you want to store the pipeline:

$ git clone https://github.com/Catanscombe/PCAMMS.git

$ cd PCAMMS

$ nohup ./setup

\*This will take several hours\*

This will install the following prerequisites:

* java version 8
* fastq-join (https://github.com/brwnj/fastq-join)
* CLARK CLAssifier based on Reduced K-mers (http://clark.cs.ucr.edu)
  + \*including building database
* Krona tools (https://github.com/marbl/Krona/wiki/KronaTools)
* Nextflow (https://www.nextflow.io)
* Fastqc (https://www.bioinformatics.babraham.ac.uk/projects/fastqc/)
* Multiqc (https://multiqc.info)
* seqtk (https://github.com/lh3/seqtk)
* bwa (http://bio-bwa.sourceforge.net)

Examples have been included to allow for installation checking

# Usage:

## metagenomic\_qc.py

usage: metagenomic\_qc.py [-h] [-nl NEG\_LIST] [-r {yes,no}] [-id SAMPLE\_ID]

[-s SAMPLE\_LIST] [-n1 NEG\_SAMPLE\_R1]

[-n2 NEG\_SAMPLE\_R2] [-t THREADS] [-rh {yes,no}]

input\_dir output\_dir run

positional arguments:

input\_dir directory containing input files

output\_dir output directory

run run description

optional arguments:

-h, --help show this help message and exit

-nl NEG\_LIST, --neg\_list NEG\_LIST

path to csv containing information on negative control

samples, see example for format

-r {yes,no}, --rename {yes,no}

option to rename files when in results directory,

based on sample number and run name, default is yes

-id SAMPLE\_ID, --sample\_ID SAMPLE\_ID

path to csv containing sample number and sample ID,

will rename samples based on sample number and sample

ID, see example for format. If the run contains one or

more negative control, please submit separately using --neg\_list command

-s SAMPLE\_LIST, --sample\_list SAMPLE\_LIST

provide a list of samples to process, default is to

process all files in input directory

-n1 NEG\_SAMPLE\_R1, --neg\_sample\_R1 NEG\_SAMPLE\_R1

path to negative control file R1

-n2 NEG\_SAMPLE\_R2, --neg\_sample\_R2 NEG\_SAMPLE\_R2

path to negative control file R2

-t THREADS, --threads THREADS

number of threads available, default is 4

-rh {yes,no}, --remove\_host {yes,no}

option remove host reads at the beginning of analysis,

default yes. Default host is Human, if another host is

needed, place the reference fasta in host directory.

See README for more detail

## metagenomic.py

usage: metagenomic.py [-h] [-c CONTAMINATION]

[-nc NEG\_CONTROL [NEG\_CONTROL ...]]

[-tc TAXON\_CONTAMINANTS] [-t THREADS]

output\_dir run

positional arguments:

output\_dir output directory

run run description

optional arguments:

-h, --help show this help message and exit

-c CONTAMINATION, --contamination CONTAMINATION

path to fasta containing local contamination library

-nc NEG\_CONTROL [NEG\_CONTROL ...], --neg\_control NEG\_CONTROL [NEG\_CONTROL ...]

list the merged negative control fastq files from

metagenomic.py you wish to be used as contamination

libraries

-tc TAXON\_CONTAMINANTS, --taxon\_contaminants TAXON\_CONTAMINANTS

provide a list of taxon ID you wish to be ignored in

the process

-t THREADS, --threads THREADS

number of threads availble, default is 4

## Basic

### metagenomic\_qc.py

$ python <Path/to/PCAMMS/metagenomic\_qc.py> <directory/containing/raw-illumina/fastqs> <output/directory> <runname>

The script will process all samples in the input directory. To only process some samples please use the –sample\_list option

### metagenomic.py

$ python Path/to/PCAMMS/metagenomic.py output/directory runname

Example:

$ python PCAMMS/metagenomic\_qc.py PCAMMS/example/ example\_output example\_run

then

$ python PCAMMS/metagenomic.py example\_output example\_run

Download the directories example\_output/qc\_results and example\_output/results to view output files

## With Negative control

$ python <Path/to/PCAMMS/metagenomic\_qc.py> <directory/containing/raw-illumina/fastqs> <output/directory> <runname> -n1 <Path/to/neg\_control\_R1.fastq> -n2 <Path/to/neg\_control\_R2.fastq>

This will classify the negative control and produce a Krona plot for you to inspect. If you then wish to use this as a contamination file use the following. This will remove any reads that map to the negative control from the classification analysis of samples.

Then

$ python Path/to/PCAMMS/metagenomic\_qc.py output/directory runname -nc output/directory/neg\_control/neg\_control.fastq

Example:

$ python PCAMMS/metagenomic\_qc.py PCAMMS/example/ example\_negative\_control example\_neg -n1 PCAMMS/example/neg\_meta\_R1.fastq -n2 PCAMMS/example/neg\_meta\_R2.fastq

Then

$ python PCAMMS/metagenomic.py example\_negative\_control/ example\_neg -nc example\_negative\_control/neg\_control/neg\_example\_neg.fastq

Multiple negative controls

The negative controls need to be in the raw data file with the other samples. Provide the negative controls on a csv using the following format (with headings):

control type,sample ID

PCR-neg,1

water-neg,19

eg if the PCR control was sample one on your Illumina run (1\_S1\_L001\_R1\_001.fastq) the ID will be 1.

The sample will be renamed based on the information in the csv eg 1\_S1\_L001\_R1\_001.fastq will become PCR-neg\_R1.fastq

$ python <Path/to/PCAMMS/metagenomic\_qc.py> <directory/containing/raw-illumina/fastqs> <output/directory> <runname> -nl Path/to/neg\_list.csv

Then if you want to use all/some of these samples as contamination libraries u se the -nc command followed by a list of the fastqs, from the neg\_control directory.

$ python Path/to/PCAMMS/metagenomic\_qc.py output/directory runname

-nc output/directory/neg\_controls/neg\_1.fastq output/directory/neg\_controls/neg\_2.fastq

Example

$ python PCAMMS/metagenomic\_qc.py PCAMMS/example/example\_neg\_list/ example\_neg\_list neg\_list -nl PCAMMS/example/example\_neg\_list/neg\_list.csv

If you then want to use both of these as contaminant libraries:

aired/

$ python PCAMMS/metagenomic.py example\_neg\_list/ neg\_list -nc example\_neg\_list/neg\_control/water-neg\_neg\_list.fastq example\_neg\_list/neg\_control/PCR-neg\_neg\_list.fastq

## Using a local contamination knowledge

These options allow you to include knowledge of contamination that you frequenctly encounter eg kit contaminants. This can be included in two ways, either in the form of a local contamination library in fasta format, or by proving a list of taxon ID you wish the pipeline to ignore.

### Local contamination library

In the metageneomic.py script use the -c (--contamination) command to give the path to the fasta file containing the sequences you with to be removed.

### List of taxon to ignore

You can provide a list of taxon ID which you wish to ignore in the analysis eg if you identify *Guillardia theta* and *Xanthomonas citri* in multiple runs and believe it is contamination you can provide the pipeline with their taxon ID and the pipeline will ignore them. The taxon ID need to be provided in a csv file with one taxon ID in each row.

55529

346

These will still appear in the Krona plot and other CLARK-L output files, but will not be reference assembled.

Example:

$ python PCAMMS/metagenomic\_qc.py PCAMMS/example/ taxon\_contaminaation taxon\_contam

$ python PCAMMS/metagenomic.py taxon\_contaminaation taxon\_contam -tc PCAMMS/example/taxon\_contamination.csv

## Customising host read removal

The default is to remove reads mapping to the human from downstream analysis. This option can be turned off, or the host genome could be changes (eg to a chicken genome)

### Not removing host

When running the metagenomic\_qc.py script use -rh no (--remove\_host no)

$python <Path/to/PCAMMS/metagenomic\_qc.py> <directory/containing/raw-illumina/fastqs> <output/directory> <runname> -rh no

### Customizing host

To change the reference used as the host, put a single fasta file containing the host reference in the directory PCAMS/host. Remove all the files associated with the human.gasta.gz. Run bwa index on the new reference. The pipeline will now automatically use this fasta as the host reference.

## Running the pipeline from a sample list

The default of the pipeline is to process all samples in a directory. However using the -s (--sample\_list) option you can process a subset of the files in the directory by providing a list of samples to process. The samples need to be given in a csv with each sample on each line, you must list the forward and reverse files for each sample

e.g

sample1\_meta\_R1.fastq

sample1\_meta\_R2.fastq

sample2\_meta\_R1.fastq.gz

sample2\_meta\_R2.fastq.gz

Example

$ python PCAMMS/metagenomic\_qc.py PCAMMS/example/ example\_list example\_list -s PCAMMS/example/sample\_list.csv

## Renaming samples based on a list

If you provide the pipeline with a list of samples and sample names the pipeline will process those samples and rename based on information provided. Provide the information in csv format, with two columns, the first column should contain the sample ID you ish to use for the sample, this should not contain spaces. The second column should have the sample number from the raw Illumina run eg 1\_S1\_L001\_R1\_001.fastq will be 1. See example below and in PCAMMS/example/example\_neg\_list/sample\_ID.csv. Keep headings. Use the -id (--sample\_id) option to give the pipeline the list.

sample\_ID, sample\_number

samplea,1

sampleb,2

samplec,3

this will rename 1\_S1\_L001\_R1\_001.fastq as samplea\_runname\_R1.fastq

Example

$ python PCAMMS/metagenomic\_qc.py PCAMMS/example/example\_neg\_list/ sample\_ID example\_sample\_ID -id PCAMMS/example/example\_neg\_list/sample\_ID.csv