**QWRAP v1**

**Updated : 19 July 2014. Tested for QIIME v. 1.7 (we are working to make it compatible for QIIME 1.8)**

**For any questions email at - rkumar@uab.edu**

**Installing QWRAP and other dependencies**

Minimum requirements: User should have a basic idea of how to install and access softwares on Linux operating system. These tools must be installed for proper functioning of the tool QWRAP. All these tools can be either installed on a Linux machine or a [QIIME virtual Box](http://qiime.org/install/virtual_box.html).

* 1. QIIME: There are several ways to install QIIME (<http://qiime.org/install/index.html>). Virtual box based installation (<http://qiime.org/install/virtual_box.html>) seems to be the easiest which can also be installed on a windows machine. The QIIME v 1.7 virtual box image can be downloaded frsom this link (<ftp://thebeast.colorado.edu/pub/qiime-release-VMs/QIIME-1.7.0-amd64.vdi.gz>)
  2. FASTQC (<http://www.bioinformatics.babraham.ac.uk/projects/fastqc/>)
  3. FASTX (<http://hannonlab.cshl.edu/fastx_toolkit/>)
  4. Usearch (<http://www.drive5.com/usearch/>). After downloading the 32bit linux binary, rename the binary file to usearch61
  5. R (<http://www.r-project.org/>) . This is usually available on Linux machines.
  6. QWRAP is available for download at <https://github.com/QWRAP/QWRAP>. It can be installed using the following command on a Linux terminal as

git clone git://github.com/QWRAP/QWRAP.git

This command will download all the QWRAP scripts and the example dataset to the current directory in a folder named QWRAP. Now you need to add the location of this folder to the system PATH variable (so that you can run the program from any location on computer). For example, if absolute path for the QWRAP folder is “/home/username/QWRAP” then you can add the following line to file “.bashrc” (/home/username/.bashrc) which is present in your home directory

export PATH=${PATH}:/home/username/QWRAP/

and run the following command to update environment

source /home/username/.bashrc

**Checking other tools and dependencies**

To check if all the necessary dependencies for QWRAP are installed, a script “check\_qwrap\_plus.sh” is provided. Run the script as

check\_qwrap\_plus.sh

It will check for all dependencies and tells whether it is a success or failure. In case of any failure please re-install the necessary program.

**RAW DATA**

FASTQ files are expected as raw data. QWRAP is designed for de-multiplexed dataset (each sample is single fastq file without any barcodes present). So if your reads are multiplexed, please demultiplex them before using QWRAP.

**SINGLE END READS ANALYSIS**

Make sure all the raw data files (fastq/fastq.gz format) are present in one folder. Usually, each fastq file represents one sample. Please rename the files so that it reflects a proper sample name. The QWRAP program works on all the fastq files present in a given folder.

Let’s take an example. The QWRAP program folder containing a sample dataset called as “example\_rawdata.tar.gz” which has 8 samples from four different groups. We can copy the file and uncompress it. It looks like

cp home/username/QWRAP/example\_rawdata.tar.gz .

tar xvf example\_rawdata.tar.gz

ls example\_rawdata/

shows the list of files

A1.fastq.gz A2.fastq.gz B1.fastq.gz B2.fastq.gz C1.fastq.gz C2.fastq.gz D1.fastq.gz D2.fastq.gz

Let us create a new directory for analysis and storing the results and name it as “ANALYSIS”. Open the directory “ANALYSIS”, we are going to run all the commands inside this directory to create a well-defined structure of different results, which is later useful for generating HTML report:

mkdir ANALYSIS

cd ANALYSIS

**Quality checking of raw data**

Run the data quality check on all files present in the raw data folder (compressed fastq files). If the raw data is located in folder “../example\_rawdata”, then run the program as

quality\_check\_rawdata.sh ../example\_rawdata

The following command calls a program FASTQC and runs it over all the files present in the “example\_rawdata” directory. This script creates a folders fastqc\_rawdata with FASTQ report for all samples. Inside the folder, a HTML file “FASTQC\_overview.html” is created which provides a combined overview of the quality statistics for all samples and also provide more detailed report for individual samples. Users are suggested to look at the report carefully and decide the parameter which might be useful for quality filtering of raw data. Please note the length of sequence reads present in the dataset as this parameter can be used later for trimming bad quality reads.

**Quality Filtering and quality checking for processed data**

Here quality filtering is performed in two steps.

1. All the reads are trimmed to a user defined sequence length (TRIM\_LENGTH).

2. Only high quality reads are kept: for example reads which has more than 80% bases (QC\_PERCENT) with a QScore >20 (QC\_SCORE). The values can be changed when running the commands.

The program “quality\_filter\_single.sh” requires the location of the folder containing raw data (FASTQ files) and the user defined length to trim the reads and the two quality filtering parameters. If the user doesn’t want to trim the reads, they can provide the full length of the reads (therefore no trimming is performed). The script requires 4 parameters as can be used as

quality\_filter\_single.sh INPUT\_FOLDER TRIM\_LENGTH QC\_PERCENT QC\_SCORE

For example, if a user want to trim the length of reads to 200 bases and run the QC where he wants to select reads where >= 80% bases have a QScore >20. He can use the following commands

quality\_filter\_single.sh ../example\_rawdata 200 80 20

The program does the quality filtering and produces the filtered fastq files in a directory called “filtered\_fastq”.

The program also creates the fasta file for all the samples in the current directory “ANALYSIS” which are used for subsequent analysis.

The program “quality\_check\_filterdata.sh” uses the directory filtered\_fastq to generate the quality report for the filtered fastq files.

quality\_check\_filterdata.sh filtered\_fastq

This script creates a folders fastqc\_filterdata with FASTQC report for all files of the filtered dataset. Inside the folder, the HTML file “FASTQC\_overview.html” is created which provides an combined overview of the quality statistics for all samples and also provide more detailed report for individual samples.

**Generating QIIME input files and running QIIME scripts**

Based on the quality report of filtered dataset if the user wants to remove certain samples (sometime because of low sampling depth), they can just delete the corresponding fasta file from the “ANALYSIS” directory. All the fasta files present in the current directory (ANALYSIS) will be used/included in the analysis.

The data analysis workflow is going to generate two set of results.

First set is the original OTU tables and other files on which no rarefaction (sample size normalization) and no rare OTU filtering was applied. All the files and folder generated in this step has suffix "\_org" (original). This was used to generate taxonomy charts. This step is performed by script "script.sh".

Second set might appeal to most users. Here the OTU table is rarified to user supplied sample depth (or minimum sample depth present across the samples). The OTUs are then filtered which had average abundance <0.0005%. This generated a new filtered OTU table. All files and folder generated at this step has suffix "\_fil". This was used to generate taxonomy bar charts, OTU Multiple Sequence Alignment, phylogenetic tree, alpha diversity, beta diversity etc. This step is achieved by script "script\_adv.sh".

The program “microbiome-workflow1.sh” runs inside the ANALYSIS directory and creates several files which are required for any QIIME based analysis. You can run it as

microbiome-workflow1.sh

The program generates the following files:

a) seqs.fna: Here all the fasta files present in a current folder are merged into a single fasta file called “seqs.fna”. The sample name becomes the header for the reads to distinguish different samples. A sample snapshot is given below. Since read headers are not very informative, they are renamed as 1,2… to save disk space.

>A1\_1

AACGTAGGTCACAAGCGTTGTCCGGAATTACTGGGTGTAAAGGGAGCGCA

>A1\_2

TACGTAGGGGGCAAGCGTTATCCGGATTTACTGGGTGTAAAGGGAGCGTA

…

>B1\_1

TACGGAGGATCCGAGCGTTATCCGGATTTATTGGGTTTAAAGGGAGCGTA

>B1\_2

AACGTAGGGTGCAAGCGTTGTCCGGAATTACTGGGTGTAAAGGGCGCGCA

…

b) mapping.txt: A default mapping file is created with a single column containing a header and sample names as shown below.

#SampleID

#Mapping file for the QIIME analysis

A1

A2

B1

B2

C1

C2

D1

D2

The mapping file can/should be edited to add group information about the sample (like control vs treatment). For example, since these 8 samples belong to four groups (A, B, C, D) a new column called “Group” can be added as shown below. This is a tab delimited file so all columns are separated by tab. In a similar fashion more attributed of samples can also be added. This will help in PCoA plots to distinguish different groups. It will also help in other statistical analysis.

#SampleID Group

#Mapping file for the QIIME analysis

A1 A

A2 A

B1 B

B2 B

C1 C

C2 C

D1 D

D2 D

c) sample\_order.txt: This file is used to determines how the samples are ordered in the OTU table. This is especially important as the same order is used for generating taxonomy bar charts. You can change the order of samples manually here to reorder them in OTU table.

A1

A2

B1

B2

C1

C2

D1

D2

**d) script.sh :** This file has all the commands required to generate the OTU table and taxonomy bar chart. Most of the commands are from the QIIME package and are arranged in a certain order. More details about the command can be found at <http://qiime.org/scripts/>.

Users can look into the file script.sh for exact command and their parameters. An example of “script.sh” file can be viewed [here](https://github.com/QWRAP/QWRAP/blob/master/sample_script.sh). Briefly, the script is going to perform the following functions in order

* Chimera Filtering using program "Usearch".
* OTU Clustering at 97% sequence similarity using program "uclust".
* Picking representative of OTUs based on abundance.
* Assigning taxonomy to OTUs using RDP classifier (threshold 0.8) using Greengenes database.
* Sorting OTU table based on file "sample\_order.txt".
* Summarizing OTUs into taxonomic groups.
* Creating Normalized OTU table i.e. converting raw numbers from OTU table into proportion and also merging the taxa information in a single file.
* A filtered list of top 10, top 25 and top 100 OTUs and taxa are generated.

Execute the file “script.sh” as

sh script.sh

This will execute all the commands present in the file and generate unfiltered/unrarified OTU table and the taxonomic charts. Please note here that the OTU table is not normalized for sample size differences and there is no filtering done at this step to remove any rare taxa. We call this as original (all files include “\_org” in their name). The taxa charts (taxa\_summary\_org), OUT table (otu\_table\_org.biom / otu\_table\_org.txt) are generated.

To further advance the analysis with sample size normalization, rare OTU filtering, alpha and beta diversity calculations, another program “microbiome-workflow2.sh” is provided. Further advanced analysis is performed in which the following steps are performed

* The OTU table is rarified to user specified (or minimum) sample size.
* Rare OTUs are filtered at abundance level < 0.0005%.
* Summarizing OTUs into taxonomic groups.
* Creating Normalized OTU table i.e. converting raw numbers from OTU table into proportion and also merging the taxa information in a single file.
* A filtered list of top 10, top 25 and top 100 OTUs and taxa are generated.
* Generates a multiple sequence alignment ofOTUs using the program "PYNAST" and creates a phylogenetic tree using the program "FASTTREE"
* Calculates alpha diversity (using chao1, observed species, PD whole tree, shannon, and simpson diversity indices) and generates plots.
* Calculates beta diversity (using bray curtis, unweighted unifrac, and weighted unifrac distances) and generates plots.
* Generates a UPGMA tree of all samples.

An example of “script\_adv.sh” file can be viewed [here](https://github.com/QWRAP/QWRAP/blob/master/sample_script_adv.sh). The script requires one user defined parameter “sampling depth” (read depth). Since different samples may have different read depth, the read depth should be normalized across all samples. When a read depth is provided, a random sampling event is used to rarify the OTU table. If the sampling depth is 22986 (in our example dataset), you can run the script as

microbiome-workflow2.sh 22986

If no sampling depth is provided, the program will automatically calculate the minimum sampling depth from the file “otu\_table\_org.stats.txt”

The file “script\_adv.sh” can be executed as

sh script\_adv.sh

**Generating HTML report**

Although most of the results are generated in ANALYSIS folder and can be explored manually, we have developed a script which can put together most of the results into a user friendly HTML report. The HTML report is a static so the whole ANALYSIS folder can be moved or shared with another person or the results can also be shared using a webserver.

Run the following program to generate the HTML report.

report\_microbiome.sh

This creates an html file “microbiome\_report.html” in the ANALYSIS directory and can be opened using any web browser.

The most updated version of the protocol ([QWRAP-Readme.docx](https://github.com/QWRAP/QWRAP/blob/master/QWRAP-Readme.docx)) is also available at

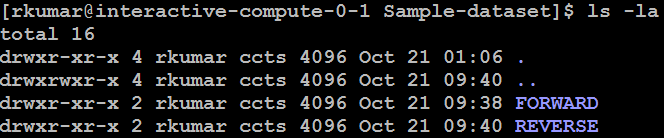
<https://github.com/QWRAP/QWRAP>. On the github page click the button “RAW” to view this file. This readme file also has instructions to perform paired end microbiome data analysis.

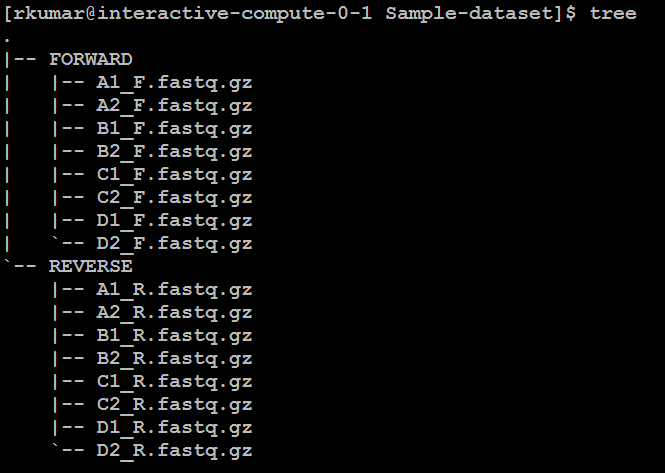
**PAIRED END READS ANALYSIS**

Locate the folders with forward and reverse reads.

Here in this example they are stored in separate folders FORWARD & REVERSE for forward and reverse end reads.

Just check the files.





**Step A: Create directory for analysis**: Create a directory to store analysis results and get (cd) into it.

mkdir ANALYSIS

cd ANALYSIS

**Step B: For Quality check before merging:**  run the program “quality\_check\_before.sh” with location of FWD reads and REV reads.

quality\_check\_before.sh ../FORWARD ../REVERSE

This creates two folders fastqc\_beforef and fastqc\_beforer with FASTQC results for the forward and reverse files and stats.

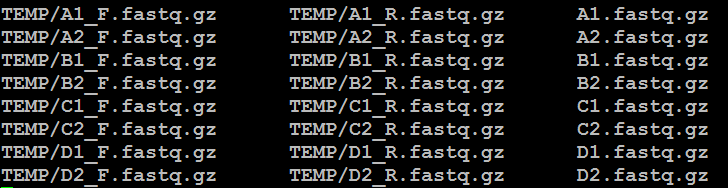
**Step C**: **Merge the forward and reverse files**: Run program: merge\_reads\_F\_R.sh with parameters containing the location of forward and reverse folder. This creates a TEMP folder which has all the reads for the analysis.

merge\_reads\_F\_R.sh ../FORWARD ../REVERSE

**Step D:** **Prepare for merging reads:** The program “prepare\_merge\_fastq.sh” requires the location of TEMP folder (containing both forward and reverse files) as a command line argument. Run the program as

prepare\_merge\_fastq.sh TEMP

This program creates a mapping file "Paired\_Filelist.txt" with each line containing the location of forward read and reverse reads and the predicted name of merged read.



You can edit the column 3 of the merged file in a text editor if required (especially if you want to rename the files. Make sure that the names in 3rd column should not include underscore (\_).

**Step E: Merge the reads:** This is done using program merge\_fastq.sh. This script uses program USEARCH for merging reads. The program needs 5 parameters which includes the quality control parameters for merging.

1. Name of mapping file (Paired\_Filelist.txt)
2. Length for trimming forward reads. Provide reads full length if no trimming is required.
3. Length for trimming reverse reads. Provide reads full length if no trimming is required.
4. Max mismatch allowed between forward and reverse reads when aligned.
5. Minimum overlap required between forward and reverse reads when aligned.

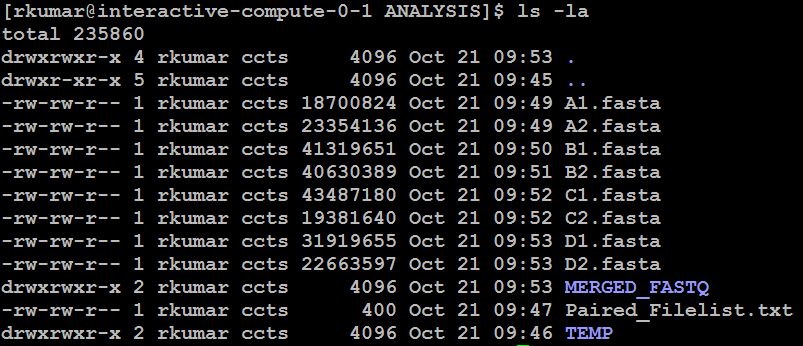
In this examples since we had reads of length 251 and decided not to trim them, we used following parameters.

merge\_fastq.sh Paired\_Filelist FWD\_TRIM REV\_TRIM USEARCH\_MAXDIFF USEARCH\_MINOVERLAP

merge\_fastq.sh Paired\_Filelist.txt 200 200 15 50

This script after merging creates all fasta files in the current directory. It also stores the FASTQ files in a folder MERGED\_FASTQ.

So the current ANALYIS directory will look something like this



**After this step, these seuqences are treated as single end reads and the analysis workflow used for processing single end reads follows. For ex**

**Step F: Quality filtering after merging:**  run the program “quality\_filter\_single.sh” with location of merged FASTQ reads. The command line arguments are described above.

quality\_filter\_single.sh MERGED\_FASTQ 200 80 20

The program does the quality filtering and produces the filtered fastq files in a directory called “filtered\_fastq”.

The program also creates the fasta file for all the samples in the current directory “ANALYSIS” which are used for subsequent analysis.

The program “quality\_check\_filterdata.sh” uses the directory filtered\_fastq to generate the quality report for the filtered fastq files.

quality\_check\_filterdata.sh filtered\_fastq

This script creates a folders fastqc\_filterdata with FASTQC report for all files of the filtered dataset. Inside the folder, the HTML file “FASTQC\_overview.html” is created which provides an combined overview of the quality statistics for all samples and also provide more detailed report for individual samples.

Please follow the rest of steps described for single end analysis.