Instituto
Politécnico de Setúbal
Escola Superior de
Tecnologia do Barreiro

JOÃO PEDRO FERNANDES LOURENÇO DA FILOMENA

ADAPTATION OF GENOQUAL PIPELINE TO NEW UPSTREAM APPLICATIONS AND TO RUN INDEPENDENTLY FROM GALAXY PORTAL

Relatório de estágio do Mestrado em Engenharia Biológica e Química

ORIENTADOR

Professora Doutora Ana Gabriela Gomes

SUPERVISORES

Doutora Ana Portugal Melo

Doutor Ricardo Leite

Maio de 2021

JOÃO PEDRO FERNANDES LOURENÇO DA FILOMENA

ADAPTATION OF GENOQUAL PIPELINE TO NEW UPSTREAM APPLICATIONS AND TO RUN INDEPENDENTLY FROM GALAXY PORTAL

JÚRI

Presidente: Professora Doutora Maria de Lurdes de Figueiredo Gameiro, ESTBarreiro/IPS

Supervisor: Doutor Ricardo Bastos Leite, Instituto Gulbenkian de Ciência

Vogal: Dr. João Pedro Matos Silveiro Costa, Gene Express, Instituto Gulbenkian de Ciência

Maio de 2021

Acknowledgments

First of all, I would like to express my gratitude to the ESTBarreiro/Polytechnical Institute of Setúbal, for the opportunity to accomplish my master's biological and chemical engineering degree and also to: the course coordinator, Professor Maria de Lurdes Gameiro, my internship's advisor, Professor Ana Gabriela Gomes, and my supervisor, Ricardo Leite, for the internship program as well as the support given me throughout my whole degree.

I'd also like to express my deepest thanks to the BioData.pt staff, namely to Dr. Ana Portugal Melo, Dr. João Costa, Dr. Daniel Faria, João Raimundo, João Rato, and again, to Dr. Ricardo Leite, for their constant support and help through the weekly online meetings we've had throughout the internship.

Resumo

O presente estágio foi realizado no Instituto Gulbenkian de Ciência (IGC) no âmbito do mestrado em engenharia biológica e química.

O estagiário esteve envolvido com termos e ferramentas usadas em bioinformática, metagenómica e *NGS.* A principal tarefa do estagiário focou-se na atualização de uma pipeline de análises genómicas feito pelo IGC designado por “GenoQual”. O principal objetivo do estágio do discente focou-se na atualização de uma pipeline de análises genómicas feito pelo IGC, há vários anos, designado por “GenoQual”. Desde a última atualização do GenoQual, tem havido uma evolução natural das ferramentas usadas em bioinformática, surgindo assim novas alternativas e melhorias nas ferramentas usadas pelo GenoQual. Uma das atualizações mais importantes foi o lançamento do QIIME 2 que trouxe melhorias e novas funcionalidades em relação ao QIIME ainda em utilização no GenoQual. A tarefa principal desta dissertação foi a atualização do código Python do pipeline de modo ser compatível com uma versão mais recente de Python e adicionar novas funcionalidades à pipeline, nomeadamente a compatibilidade com o QIIME 2 e Kraken2.

O projeto foi organizado em duas etapas distintas, a primeira foi a atualização do código do GenoQual de Python 2.7 para o novo Python 3.x. A segunda etapa consistiu na atualização dos softwares utilizados pela versão original do GenoQual de modo garantir que a nova pipeline era compatível com as novas versões desses softwares para aproveitar as novas melhorias e funcionalidades provenientes das novas atualizações.

O código do GenoQual foi sucessivamente atualizado de modo ser compatível com o Python 3.8 e foi proposto a adição da nova plataforma de bioinformáticas microbioma QIIME 2 e o classificador taxonómico Kraken 2 de modo poder realizar analises do tipo 16S e WGS.

PALAVRAS-CHAVE: Bioinformática, Metagenómicas, Pipeline, Python, QIIME 2, NGS, Sequenciamento, FASTQ

Abstract

The following internship was developed at the Instituto Gulbenkian de Ciência (IGC) in the scope of the master's biological and chemical engineering degree.

The intern dealt with bioinformatics, metagenomics and NGS related terms and tools and focused on the task of updating a pipeline of genomic analyses developed by IGC a few years ago designated as "GenoQual. Ever since GenoQual was last updated, there has been a natural evolution of the tools used in the bioinformatics field, appearing newer alternatives and updates to the tools used by GenoQual. One of the main updates that occurred was the release of QIIME 2 which brought newer upgrades and features in relation to QIIME 1 which GenoQual was still using at the time. The main objective of this internship was to update the Python code used by the pipeline so that it would become compatible with a more recent Python version as well as adding newer functionalities to the GenoQual pipeline, namely the compatibility with QIIME 2 and Kraken 2.

The project was organized into two distinct stages; the first was the updating of GenoQual's Python 2.7 code to the newer Python 3.x version. The second stage was the updating of the packages used by the original version of GenoQual to make sure that the pipeline was still compatible with the newer versions of those required packages, so that it could make use of their improvements and newer functionalities.

GenoQual's code was successively updated to be compatible with Python 3.8 and the addition of the new microbiome bioinformatics QIIME 2 platform and Kraken 2 taxonomic classifier were proposed as additions to the GenoQual pipeline so that it would be able to do both 16S and WGS type analyses.

KEYWORDS: Bioinformatics, Metagenomics, Taxonomy, Pipeline, Python, QIIME 2, NGS, Sequencing, FASTQ

INDEX

LIST OF FIGURES

LIST OF TABLES

GLOSSARY AND ABBREVIATIONS

Amplicon sequencing – Genomic analysis approach that targets a specific fragment or locus of DNA from a target organism.

ASV – **Amplicon Sequence Variants** – A group of exact DNA sequences

BLAST – **Basic Local Alignment Search Tool** – Sequence similarity search algorithm.

Bash – **Bourne Again Shell** – Command language interpreter.

bp – **Base pairs**

Bracken – Companion program to Kraken that allows estimation of abundance.

Conda – Package and environment management system.

DADA2 – **Divisive Amplicon Denoising Algorithm version 2** - Denoising algorithm for amplicon sequencing.

Deblur – Denoising algorithm for amplicon sequencing.

Demultiplex – Process of sorting sequences present in a single file into different files.

DNA – **Deoxyribonucleic acid**

FASTQ – Text file format for NGS reads that contains both the DNA sequence and quality information about each version.

FastQC – **Fast Quality Check** – Program that performs several quality checking related analysis.

GenoQual – Genomic analyses pipeline developed by IGC.

GitHub – Web-based hosting service for software development projects.

I1/I2 – **Forward index / Reverse index** – Unique short DNA sequence added to each DNA fragment during library preparation which allows to identify and sort sequences. Also known as a “barcode” or “tag”.

IGC – **Instituto Gulbenkian de Ciência**

ITS – **Internal Transcribed Spacer** – Spacer DNA that’s commonly used as a barcode for the identification of fungi species.

Kraken – Taxonomic classification system that uses exact k-mer matches.

k-mer – Term applied for a substring of length *k* in a genetic string.

Metagenomics – The study of complete microbial populations in environmental and medical samples.

MultiQC – **Multi Quality Check** – Program that compiles bioinformatics analyses results across different samples into a single report.

NCBI – **National Center for Biotechnology Information**

NGS – **Next Generation Sequencing** – Large-scale DNA sequencing technology.

OTU – **Operational Taxonomic Unit** – A group of similar DNA sequences

Pavian – Interactive browser application written in R for analyzing and visualizing metagenomics classification results from classifiers such as Kraken.

PCR – **Polymerase Chain Reaction** – Method used to make millions of copies of a specific DNA segment.

QIIME – **Quantitative Insights Into Microbial Ecology (QIIME 1, QIIME 2)** – Microbiome analysis package.

Reads – Inferred sequence of base pairs corresponding to all or part of a single DNA fragment.

R – Language and environment for statistical computing and graphics

R1 / R2 – **Forward reads / Reverse Reads** – Inferred raw sequence of base pairs produced by a sequencing machine.

rRNA – **Ribosomal Ribonucleic acid**

SSH – **Secure SHell** – Network protocol that gives users a secure way to access a computer over network.

Taxonomy – Classification of living and extinct organisms.

VM – **Virtual Machine** – Emulation of a computer system

WGS – **Whole Genome Sequencing** – Analysis of the entire genomic DNA of a cell

16s/18s rRNA – **16 / 18 Svedberg units Ribosomal Ribonucleic acid**

1. INTRODUCTION

The present document is the result of an internship in the bioinformatics area and was performed at the Instituto Gulbenkian de Ciência (IGC) between September 2020 and February 2021. However, due to the ongoing COVID-19 pandemic, the internship was mostly accomplished remotely with regular online reunions and discussions.

In this postraining, the focus was to apply the acquired biotechnology and bioinformatics knowledge and technics obtained throughout the bachelor and Master degree in order to update IGC's GenoQual pipeline and add newer functionalities to the code.

The GenoQual pipeline is a complex script written in Python that performs several genomic analysis and operations automatically on the provided FASTQ format reads, such as checking their quality through FastQC (Fast Quality Check), demultiplexing, merging reads, BLAST (Basic Local Alignment Search Tool) analysis, taxonomy assignments and provide a final *html* report with all the necessary info produced during the course of the pipeline. This pipeline has the goal of properly analyzing a provided sample in a single machine, assure that the samples are not dependent from each other, that the quality scores of the base calls below 20 are disregarded and that at the end it manages to identify the sequenced organism properly when taking into account the source of the samples.

This pipeline was last updated 4 years ago and was still using some severely outdated packages. The code was also all written on the Python 2 interpreter and it relied on the use of the Galaxy Bioinformatics Portal server, which is not a reliable permanent solution due to its online connectivity dependencies.

There have also been recent major advances in the bioinformatics field and consequently there are now newer alternatives and standards. The most noticeable one, regarding GenoQual's outdated code, was the introduction of ASV (Amplicon Sequence Variants) tables as an alternative to OTU (Operational Taxonomic Unit) tables. ASV tables are a higher resolution version of OTU tables, in which no clustering was made. OTU tables are created by clustering sequences that have around 97% similarity while ASV tables instead focuses on giving each unique sequence its own feature in the ASV table, essentially going for a 100% similarity. As such, procedures that make use of ASV tables will allow the production of higher precision results than when compared to OTU table style processes. Despite all that, the ASV method still holds some disadvantages over the older OTU method. Since ASV tables clusters each unique sequence together, this will lead to a more complicated downstream analysis when compared to the OTU method if using highly diverse and complex samples. The ASV method is also more sensitive to the quality of the data provided, since its denoising step removes PCR or sequencing errors, which can lead to a reduced number of reads when compared to OTU tables. Since IGC works mainly with microbiome samples and requires high precision results, replacing GenoQual's old OTU style method with the newer ASV method is justified.

The DADA2 and Deblur algorithms are two of the most common choices when approaching the ASV table style analysis. These algorithms can also be run in their respective pipelines but they're also available as plugins for the QIIME2 software. Since GenoQual used the original QIIME software in its pipeline to generate OTU tables, the updated code would instead run the newer QIIME2 version and generate ASV tables using the DADA2 denoise plugin.

As for other newer taxonomic classification tools, IGC has shown interest in incorporating Kraken 2 in GenoQual's pipeline due to it being a much faster alternative. Unlike QIIME 2 that uses clustering techniques to classify samples, Kraken2 is based on mapping exact k-mer matches to the lowest common ancestor to achieve high accuracy and classification speeds.

By incorporating both QIIME 2 and Kraken 2 into the GenoQual pipeline, it's possible to do either a 16s rRNA metagenomic analyses approach using QIIME 2's DADA2 plugin and a WGS (Whole Genome Sequencing) approach using Kraken 2 depending on the data that the user inputs provides while also updating the technology used during the pipeline for a newer and better one.

2. OBJECTIVES

This internship had as its primary goal the transition between the existing GenoQual to a new improved one, while updating the functionality of it. The internship had the following goals:

- Familiarize with the Python programing language and update GenoQual's current Python 2.7 code to the newer Python 3.8 version;
- Annotate and revise the pipeline's code;
- Make the code compatible with the newer versions of the packages that it requires to run;
- Introduce new functionalities to the code;
    a) Python 3 compatibility
    b) QIIME 2 and Kraken 2 compatibility
    c) The ability to perform to perform WGS analyses using Kraken2

After further progression and discussion of the work, the new main functionalities that were picked to add to GenoQual's updated code were the replacement of QIIME by its successor QIIME 2 and the introduction of the taxonomy classifier Kraken 2 due to its massive analysis speed and growing relevancy in professional metagenomic studies as well as its ability to perform WGS analyses.

3. PROJECT FRAMEWORK

The present curricular internship was developed at IGC in collaboration with BioData.pt which is ELIXIR's Portuguese distributed infrastructure for biological data.

IGC is a research institute dedicated biological and biomedical research and innovative postgraduate training located in Oeiras. It is part of Fundação Calouse Gulbenkian, a perpetual foundation with charitable, artistic, educational, and scientific stator aims (Figure 1-2) [1].

ELIXIR is an intergovernmental organization that brings together life science resources from across Europe, with the end goal of coordinating those resources so that they form a single infrastructure. Those resources consist of databases, software tools, training materials, cloud storage and supercomputers [2]. The Portuguese ELIXIR node, BioData.pt, works alongside IGC and focuses on the bioinformatics field (Figure 3) [3].

The institute offered the intern a remote Virtual Machine (VM) to access the resources needed to accomplish the several internship's goals and tasks.

Figure 1: Part of the IGC building, reprinted from the official IGC website [4]

INSTITUTO GULBENKIAN DE CIÊNCIA

Figure 2: IGC logo, reprinted from the official IGC website [4]

BioData.pt

Figure 3: BioData.pt logo, reprinted from the BioData.pt website [3]

4. ACTIVITIES CHRONOGRAM

The internship, which took place inbetween September 3 of 2020 and March of 2021, was divided into six main activities:

- Studying and researching the theoritical background needed to propertly understand the project. This part involved the analysis of several Python and bioinformatics documentation related to the work;

- Setting up the VM and the required installations and configurations. This part of the interneship started after IGC created the VM and provided remote access to it;

- Porting GenoQual's code to Python 3. This part of the internship began after establishing remote connection to the VM since the machine contained some files that were required to run the GenoQual script;

- QIIME 2 research and testing which started during the debugging of the new GenoQual script due to the need to update the code that still used QIIME 1;

- Kraken 2 research and testing to allow GenoQual to perform WGS type analysis;

- Writing the dissertation report.

The months in which each activity took place is listed in Table 1.

Table 1:Schedule of the internship that elapsed from September 3 of 2020 to March of 2021

	Months						
Activity	September	October	November	December	January	February	March
Studying and researching the theoretical background	X	X	X				
Setting up the VM and installations			X				
Porting GenoQual's code to Python 3			X	X			
QIIME 2 research and testing				X	X		
Kraken 2 research and testing					X		
Writing the dissertation report				X	X	X	X

5. THEORETICAL BACKGROUND

5.1. METAGENOMIC ANALYSIS PROCESS

Genomics is the study of the entirety of an organism's genetic information, which is encoded in DNA with a sequence consisting of four bases: adenine, thymine, cytosine and guanine. Each cell can contain multiple pieces of DNA and the entire collection of DNAs that an organism contains is called a genome, which can reach a total length of billions of base pairs (bp).

Because of the absurdly high number of bases that is found in the genome of organisms, the genomics field works constantly together with the bioinformatics field. With the use of bioinformatics tools, it's now possible to analyze, study and manage thousands of genomes in shorts amounts of time.

By using software like NCBI's BLAST, it's possible classify and compare a whole genome or even a single protein DNA sequence with millions of other sequences present in its database automatically in a feasible amount of time, allowing the investigators to learn about the origin and evolution of any given organism [5].

In the current days, the field of genomics and bioinformatics are constantly evolving and with that, entire metagenomic studies have been becoming more accessible, fast, accurate and capable of delivering more data to the investigators. Thanks to the constant creation and evolution of bioinformatics software coupled with their integrations in pipelines, it's possible to streamline the entire analysis part of the process.

A metagenomic analysis involves several steps, as shown in Figure 4. The process starts with the gathering of the samples that are going to be analyzed. The DNA of the samples is collected and then amplified through PCR if using an amplicon sequencing method. They are then ran through a sequencer which analyses all genome sequences of all species present in that sample, which then translates it into computer data. That data can then be analyzed in order to check the quality scores of the obtained reads to decide whether the data is suitable for further analysis or not. The data can then be studied through different kinds of software and be compared against reference databases with the possible end goal of discovering the diversity of microbiome species that were present in the collected samples.

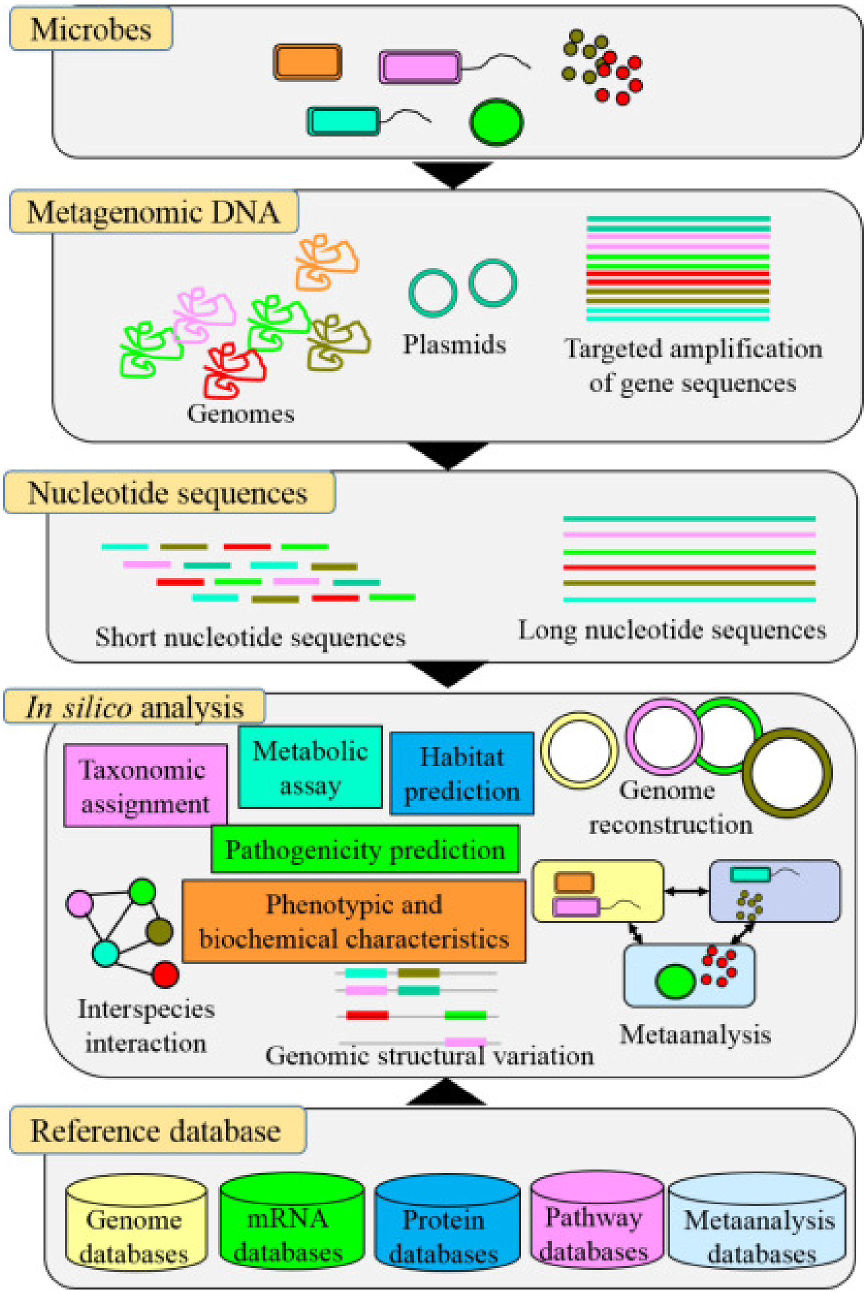


Figure 4: Metagenomic analysis scheme, reprinted from volume 45 of the *Microbiology of Atypical Environments* book series [6]

5.2. ILLUMINA

Illumina is a leading developer, manufacturer and marketer of life science tools and integrated systems for large-scale analysis of genetic variation and function [7]. Currently, Illumina offers solutions for sequencing by synthesis.

The samples used during this internship were amplicons of DNA sequences that were obtained using Illumina's MiSeq sequencer (Figure 5) and the generated data was later converted into FASTQ files using Illumina's bcl2fastq conversion software [8, 9].

Figure 5: MiSeq sequencer, reprinted from the official manufacter's website [8]

5.3 LINUX VIRTUAL MACHINE

The Linux distributions are the best-known and most-used open source systems. Bioinformatics depend heavily on Linux-based computers since most of the useful scientific software is written specifically for Linux.

Virtual Machines are a flexible and cost-effective way to run and share Linux machines in many professional environments. Thanks to the SSH network protocol, it is possible to gain easy access to a specific VM remotely and work on a Linux machine even when using a different operating system. Commands and scripts can be run in the Bash shell terminal, which allow the user to get maximum use of the virtual machine.

To help manage the VM's workflow, Conda version 4.9.2, was used. Conda is an open-source package management system and environment system that can run on Linux. Conda quickly installs, runs, and updates packages and their dependencies. It also allows for the easy creation, saving, loading and switching of environments on the local machine. Thanks to conda, it's possible to create separate environments that run a specific version of python, which is extremely useful for some packages that aren't compatible with certain versions of Python, such as the QIIME 2 pipeline that was used. It also allows for some more sensitive packages to be installed in a clean and untouched environment to avoid any possible problems and to also avoid the eventual creation of packages that might cause some interference in the regular environment [10].

5.4. PYTHON

Python is an interpreted, high-level, and general-purpose programming language and is a standard component in most Linux distributions [11, 12]. It has a massive amount of applications, offering extensive libraries and support of vast third-party modules due to its incredibly active developer community. By using Python, it's possible to automatize, customize and execute long and varied chains of different processes.

As of 2020, the Python 2 version, which was first released during 2000, has been discontinued, being completely replaced by Python 3.6.x and higher versions. Despite Python 3 being first released in 2008, Python 2 was still being developed at the same time as Python 3 for over a decade instead of being completely abandoned straight away. This is due to the fact that numerous machines all across the world were still dependent and reliant on Python 2 to work. Now that Python 2 is finally discontinued, it should be expected to see developers start creating libraries that are only compatible with Python 3.

Due to the jump from Python 2 to Python 3 bringing a major revision of the language and syntax used by the code, this jump caused major issues with backwards-compatibility. Software written in Python 2 is most of the time incompatible with Python 3 interpreters which is the case of GenoQual [12].

GenoQual, which was written in Python 2.7, had its syntax revised to make it compatible with Python 3.8. This revision will allow GenoQual to make use of newer types of packages that rely on Python 3 versions to run which in turn can give the pipeline newer functionalities.

5.5. THE GENOQUAL PIPELINE

The GenoQual pipeline developed by IGC's bioinformatics branch is a script written in Python 2 that has the end goal of analyzing the provided FASTQ files using a sequence of various different packages [13]

The Genoqual pipeline, by default, requires the following folder structure:

- Root folder
    a. Input
        i. Run_X
            1. FASTQ files
            2. Config file (designated as "QCconf.csv")
    b. Reference
    c. Results

The input folder is the directory where the bcl2fastq converted folders containing the FASTQ files are located. The "X" suffix in the folder name can be any natural integer. The user needs to provide the folder's suffix as an argument so that the pipeline starts running analyses on the correct folder. A config file named QCconf.csv, with the structure found in Figure 6, also needs to be present in this folder. In this comma-delimited file, the user provides some additional information for GenoQual to parse. The Sample_ID column of this file contains a list of the FASTQ files that the user wants GenoQual wants to analyze, according to name of the samples, allowing the user to analyze more than a single sample per run. The contaminants column specifies whether blast will be executed on the samples, the reference column lists the reference database that will be used, the demux column lists the samples that will be demultiplexed and the merge_reads column lists the samples that will have their reads merged through FLASH.

```
Sample_ID,Description,Contaminants,Reference,Demux,Merge_reads
Undetermined,Genomes,,,,
Batch40,Genomes,,,,
```

Figure 6: QCconf.csv tweaked to analyse the main bcl2fastq outputs

The reference folder is the directory where the reference genomes for quality checking and taxonomic reference files used by QIIME and are located.

A nucleotide database is also required for BLAST. This database is chosen to check possible contaminants in the samples, check the best hit for each specie and identify the organism.

The results folder is the directory where the results of the pipeline are collected. On the folder, there will be *html* reports generated by the various packages executed during the pipeline alongside a final report unique to GenoQual that summarizes most of the important data obtained during the pipeline.

The pipeline will start compiling several results at the end and generate a final report with a format similar to the one found in Figure 7. This table summarizes all of the relevant info obtained for each samples such as the number of reads above a certain quality threshold calculated by Picardtools (Q20 and Q30), the total number of bases, total number of reads, a direct link to the FastQC report generated, the primary organism present in that sample as well as some quality control information such as the contaminants found, the number of reads that failed to demultiplex and the number of reads that failed to merge. Should BLAST be skipped, the primary organism and contaminants columns will present a "=" symbol.

GenoQual Report v.1.3

Report generated on 2/1/2021, 3:43 for run 1

Sample ID	Filename	Q20	Q30	%Q30	Total bases	Total reads	FastQC	Primary organism	Contaminants
Undetermined	Undetermined_S0_L001_R1_001.fastq						Report	=	=
Undetermined	Undetermined_S0_L001_R2_001.fastq						Report	=	=
Batch40	Batch40_S1_L001_R1_001.fastq						Report	=	=
Batch40	Batch40_S1_L001_R2_001.fastq						Report	=	=
Total							MultiQC report		

Total mapped bases	Mean coverage	Theoretical coverage Q20	Theoretical coverage Q30	Total mapped reads	Total reads (used)	Qualimap	Demux read mean	Demux read stdev	Demux reads failed	Demux	Merged reads	Merged reads failed

Figure 7: GenoQual report HTML file produced at the end of the GenoQual script

The GenoQual pipeline is divided into three different python script modules: genoqual.py, graphs.py and build_db.py.

The main module, genoqual.py contains the python objects found in Table 2. Each one of those objects accomplishes a certain task, raging from pipeline preparations and checks to execution of external packages.

Table 2: List of the classes found in the genoqual.py module

Class	Function
ConfigParser	Reads the config.csv file
Pipeline	Sequentially execute the steps of the pipeline, performing checks first before executing them
DataWrapper	Creates temporary folder where data and temporary files will be created
DataCollector	Compiles the and formats the data obtained for DataFormatter to generate the report
DataFormatter	Used to generate the GenoQual HTML report at the end
BaseTool	Base skeleton for the external tools in the pipeline
QualityStats	Perform calculation of q20 and q30 statistics
FastQC	Runs FastQC on the inputted samples
MultiQC	Runs MultiQC on the FastQC output
TaxonomyClassifier	Runs seqtk and blast to estimate taxonomical distribution of the sample reads
CoverageAndQuality	Use Picard tools to count how many sequences align to a given reference
Demultiplexer	Performs demultiplexing on the files
MergeReads	Uses FLASH to merge paired reads into a single read file

Qiime	Prepares the files to run QIIME scripts
QiimeOnControls	Runs QIIME's beta diversity and taxa plotting scripts and creates GEU metadata file and fna file of the sequences

5.6. FastQC and MultiQC

FastQC is a tool that provides an overview of basic control metrics for sequencing data. It analyses FASTQ files of sequence reads and compiles a HTML report containing various types of data and information regarding the reads. It is useful for letting the user know about the overall quality of the reads and presence of adapters and other contaminants [14, 15].

In Figure 8 it is shown a quality plot generated by FastQC. The y-axis on the graph shows the quality scores, which typically ranges from 2 to 40. These scores are based on the probability of whether the sequencer called the correct base or not. The higher the score the better the base call. The background of the graph is divided into three colors, each color representing a quality range. In these types of analyses, scores above 30 is the preferred range of quality, but depending on the overall quality of the data, quality scores above 20 might also be accepted. Quality scores below 20 on the other hand are simply too unreliable to be taken into consideration and as such, should be discarded and trimmed off from the data. The blue line represents the mean quality calculated from the quartile ranges and its value can be considered as the quality score at that base pair (bp) position [16]. The quality of the reads typically drops towards the end due to the decay of the fluorescence emitted by the Illumina sequencer.

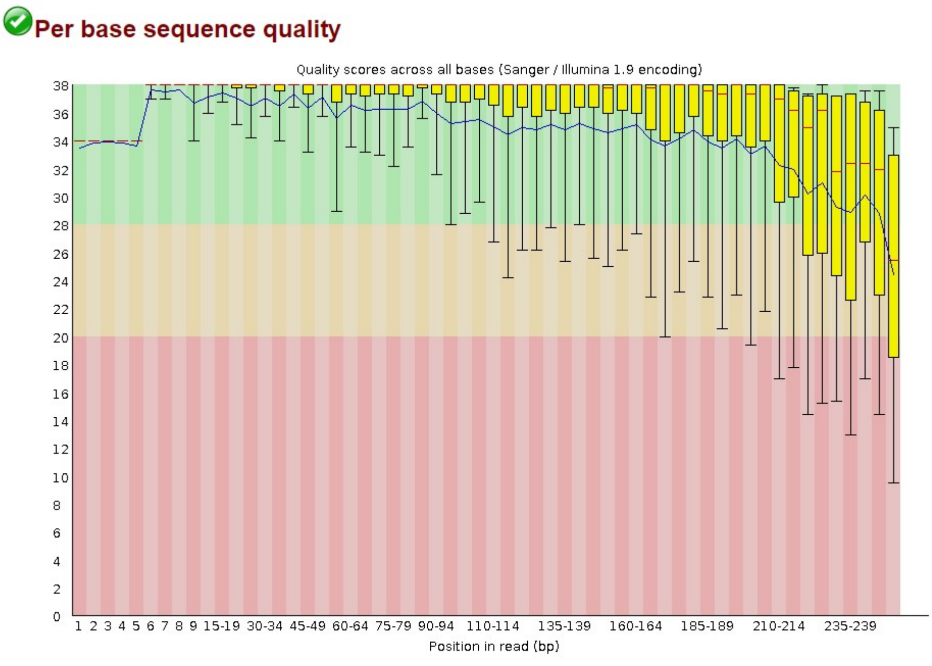


Figure 8: A FastQC quality plot generated while running the GenoQual pipeline on one of the samples

The reports generated by FastQC can then be used by MultiQC to generate a single report containing all different FastQC reports generated for all the samples and read orientations, allowing for easier comparisons and examination of the results obtained [17].

Both packages are incorporated into GenoQual and the final GenoQual results report contains a column that links the user to the MultiQC report generated during the execution of the pipeline.

5.7. 16S/18S/ITS SEQUENCING AND SHOTGUN METAGENOMICS

Metagenomics analyses are performed mainly in two methods: amplicon gene sequencing and shotgun metagenomic sequencing [18] as represented in Figure 9.

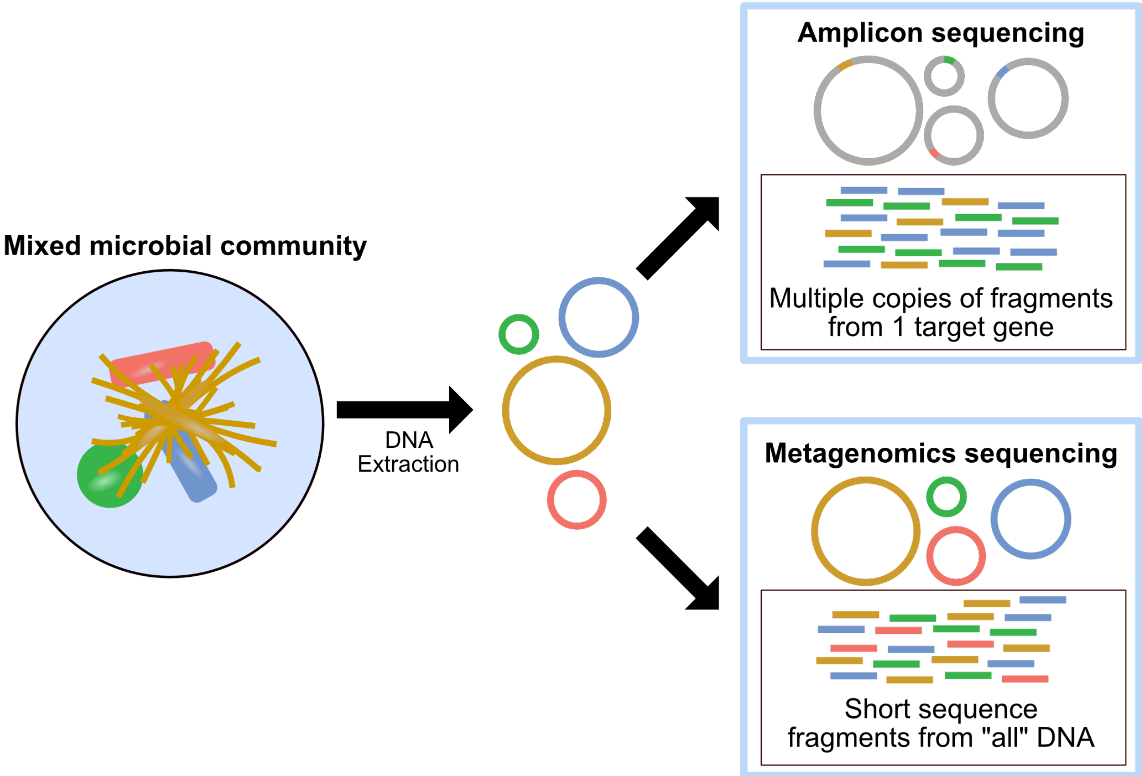


Figure 9: Amplicon sequencing and WGS methods, reprinted from Atrobiomike's *Happy Belly Bioinformatics* github page [18]

The pros and cons of both methods are summarized in Figure 10. The amplicon gene sequencing method revolves around amplifying a specific region of the 16S, 18S or ITS genes with suitable universal primers using PCR and then using Illumina to read the PCR products. The picked regions are regions that contain highly conserved genes but still distinguishable enough to be able to differentiate different species. The 16S rRNA approach is useful to study samples that contain bacteria in them while an ITS study will allow to cover fungi. The amplicon gene sequencing method, while slowly being taken over by newer WGS approaches, is still a viable option due to the lower costs, the high availability and better coverage of reference genomes and the recent appearance of the DADA2 algorithm that allows to obtain less false positive results due to its error correcting model and incorporation of ASV methods [18, 19]

The shotgun metagenomic sequencing method on the other hand focuses on sequencing the whole genome instead of just a specific region. This method is worse for bacteria and fungi, it isn't optimized for samples outside of the human microbiome and it offers a higher amount of false positive results, but unlike the amplicon approach, since the entirety of the DNA is used, it is possible to do further analyses and obtain a deeper taxonomy resolution, allowing to identify the species of the organisms present in the samples instead of just identifying the genus of the species. [18, 20]

	16S/ITS Sequencing	Shotgun Sequencing
Bacteria/Fungi Coverage	High	Limited
Cross-Domain Coverage	No	Yes
False Positives	Low Risk	High Risk
Taxonomy Resolution	Genus-Species	Species-Strains
Host DNA Interference	No	Yes
Minimum DNA Input	10 copies of 16S	1 ng
Functional Profiling	No	Yes
Recommended Sample Type	All	Human Microbiome
Cost per Sample	~ $80	~ $200

Figure 10: 16/ITS Sequencing VS Shotgun Sequencing / WGS, reprinted from Zymo Research's *16S sequencing VS shotgun metagenomic sequencing* blog page[20]

5.8. DADA2

DADA2 is a software package developed that models and corrects Illumina-sequenced amplicon errors, making it the ideal choice for 16S/18S/ITS analyses. It's written in the R programming language and uses a denoising algorithm. Instead of using an OTU clustering approach, DADA2 infers sample sequences exactly due to its ASV style approach [19, 21, 22, 23, 24].

The DADA2 pipeline analysis revolves around running the following bioinformatic tasks [21]:

1. Filter and trim reads based on quality parameters
2. Infers and estimate error rates for all possible transition or transversion point mutations
3. Denoises unique sequences and replicates reads into unique sequences with a 100% identity as opposed to the usual 97% value present in OTU clustering methods
4. Merge denoised forward and reverse reads
5. Construct an ASV table
6. Remove chimera sequences
7. Assign taxonomy by using a reference database like SILVA and GreenGenes [25, 26]

5.9. QIIME

5.9.1 QIIME 1

QIIME 1 is an open-source bioinformatics pipeline software that performs microbial analyses from raw DNA sequencing data and generates graphics and statistics of the study performed [27].

The pipeline revolves around the use of various python scripts to perform various tasks.

The QIIME 1 commands/scripts present in the original version of GenoQual are present in table 3:

Table 3: QIIME scripts used by GenoQual

Script	Function
split_libraries_fastq.py [28]	Demultiplexes the provided FASTQ files
pick_open_reference_otus.py [29]	Does OTU picking by clustering the reads against a reference database, like GreenGenes and SILVA
validate_mapping_file.py [30]	Checks whether the provided metadata.csv file is valid or not
core_diversity_analyses.py [31]	Runs core qiime diversity analyses on closed-reference OTU table to generate a biom table and a phylogeny tree
Beta_diversity_through_plots.py [32]	Generates beta diversity distance matrices for 3D PCoA Plots
Summarize_taxa_through_plots.py [33]	Generates plots of functional categories across various taxonomical levels

The main development team behind the software has dropped further development and support to the QIIME 1 software after January 1, 2018, since all of focus moved towards the development of its successor QIIME 2 [25].
With the software being practically abandoned and the fact that it’s stuck on using the outdated Python 2.7 interpreter, it’s essential that the newer revision of GenoQual swifts towards the use of QIIME 2 or a compatible alternative.

5.9.2 QIIME 2

Unlike its predecessor, QIIME 2 moved away from the use simple python scripts to the use of plugins. While a vast of important plugins are already included in the standard QIIME 2 package, there are also a lot of unofficial ones that allow for additional analyses and customization since QIIME 2 plugins can be developed by anyone [34].

QIIME 2 works mainly with two types of data files [35]:

- QIIME 2 artifacts, which typically have the .qza file extension.
- Visualizations, which typically have the .qzv file extension

Data like the FASTQ files that contain the reads to be analyzed and reference databases need to be converted to the QIIME 2 artifact file format by using importation commands. This artifact file format allows for QIIME 2 to automatically track the type, format, and provenance of data for researchers, allowing to easily trace back to all previous analyses that were run to produce the artifact, including the input data used in every step.
Visualization files on the other hand are essentially files that contain information useful for the user. These files can contain stuff like statistics, images and graphs and are usually produced at the end of a plugin as an output. QIIME 2 artifact files can also be converted to this file format anytime, allowing for the possibility of analyzing nearly every type of output created. These visualization files can be viewed via the QIIME 2 view tool [36] or by converting the .qzv file to a compressed file format like .zip, in which case contains files like .html that allow for viewing even on an offline environment.
Since the original QIIME scripts have no direct QIIME 2 plugin equivalents and since QIIME 2 allows for the use of ASV table style analyses, a new QIIME pipeline needs to be adapted from the original GenoQual QIIME commands.
The workflow used was adapted from the official QIIME 2's “plugin workflow” tutorial and can be summarized in the flowchart present in Figure 11 [37].

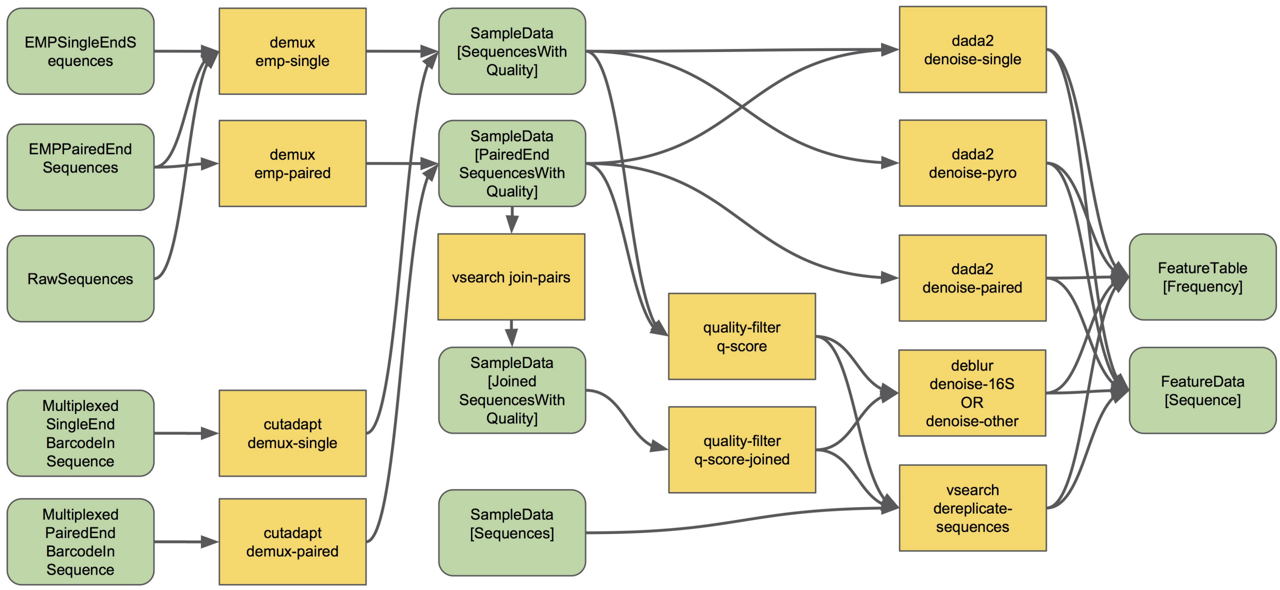


Figure 11: QIIME 2 flowchart, reprinted from QIIME 2's plugin workflows tutorial [37]

The QIIME 2 incorporates the DADA2 pipeline into its own pipeline via the plugin dada2 [38]. This plugin performs all of the necessary DADA2 steps automatically on the bash shell terminal without requiring direct use of the R software environment.

5.10. Kraken 2

Kraken 2 is a taxonomic classification system that uses exact k-mer matches to achieve high accuracy and fast classification reads. Kraken 2 excels at WGS analyses due to its shotgun genome sequencing method [39, 40].

While QIIME 2 is more of a multipurpose tool that allows for various types of operations and analyses, Kraken 2 only focuses on the classification part of a metagenomic study.

A reference sequence database is required to do the classification. The databases can be downloaded and built into Kraken 2 using a command found on the Kraken 2 package. The official website also offers a download link to an unique lightweight database file called MiniKraken2, which is more suitable for low RAM machines due to the heavy size of the main databases [40, 41].

After classifying the taxonomy with Kraken 2, the Bracken package can be optionally used to reevaluate and estimate the abundancy of species within a sample. [42]

6. METHODOLOGY AND RESULTS

6.1. WORKING ENVIRONMENT

Most of the work was performed in a Linux ubuntu virtual machine server. This virtual machine had 12 GB RAM, 6 cores and over 200 GB of storage, which allowed to run properly the software needed. The virtual machine was connected and ran remotely via a SSH server using IGC's VPN service, which gave remote access to the virtual machine's Linux terminal. This virtual machine had no graphical interface and all the work had to be done via the terminal.

PuTTY was the SSH client that was used to establish the remote-control connection since it offered various useful features such as giving a graphical interface for certain applications and the ability to save the IP address of the server [43].

Since genomic analysis can take up a lot of time to conclude, it's necessary to make sure that the processes are still properly running even after disconnecting from the virtual machine. Tmux, version 3.0a, was used to turn SSH sessions persistent even after disconnecting from the server. It also allowed running different operations on the same virtual machine at the same time by creating new sessions [44].

To manage the working environment and package installations in the SSH server, conda 4.9.2 was used. Two different conda environments were created, the first environment was installed with Python 3.8.6 and was the main working environment where the GenoQual script would run and where all the required packages, except for QIIME 2, would be installed. The second environment was created with a Python 3.6 version installation and it was used exclusively for QIIME 2, since that's currently the latest Python version that's supported by QIIME 2 [45]. By creating those two different environments using conda, it will be possible to make use of QIIME 2 even with it being incompatible with the python version GenoQual is currently running on. This was done by adding the QIIME 2 environment into GenoQual's conda environment PATH, which allows for scripts running on the main environment to call for other packages installed in other different environment, thus allowing to essentially build a link between the two environments.

The FTP client FileZilla, version 3.52.2, was also used to allow for easy and fast access and transfer of files in-between the virtual machine and the local computer [46].

The GenoQual code was edited and analyzed using the PyCharm integrated development environment. PyCharm allows for fast and easy conversion of the original Python 2.7 code to the newer 3.8 code since it automatically detects most of the outdated code syntax [47]. The new GenoQual code was uploaded in the IGC's Genomic Unit GitHub account [48].

6.2. GENOQUAL

Genoqual's github page contains a .xml file with the format found in Figure 12 that lists all the required packages and modules that the pipeline requires to run, as well the versions of the packages that were used to originally run the pipeline. Some of those packages also require some specific packages that aren't listed in the .xml file to install properly [49].

```
<tool id="genoqual_1_3" name="genoqual_1_3" version="1.3">
    <description>GenoQual - Quality Control pipeline for Illumina reads</description>
    <requirements>
        <requirement type="package" version="2.2.31">blast</requirement>
        <requirement type="package" version="r75">seqtk</requirement>
        <requirement type="package" version="0.11.4">FastQC</requirement>
        <requirement type="package" version="1.3.1">samtools</requirement>
        <requirement type="package" version="0.7.12">bwa</requirement>
        <requirement type="package" version="0.6.1">htseq</requirement>
        <requirement type="package" version="1.11.2">numpy</requirement>
        <requirement type="package" version="2.2">qualimap</requirement>
        <requirement type="package" version="1.2.11">flash</requirement>
        <requirement type="package" version="1.66">biopython</requirement>
        <requirement type="package" version="1.5.3">matplotlib</requirement>
        <requirement type="package" version="0.7.1">seaborn</requirement>
        <requirement type="package" version="0.19.1">pandas</requirement>
        <requirement type="package" version="0.7">multiqc</requirement>
        <requirement type="package" version="1.9.1">qiime</requirement>
        <requirement type="python-module" version="1.66">Bio</requirement>
    </requirements>
    <command interpreter="python">genoqual.py
```

Figure 12: genoqual.xml file listing the dependencies needed to run the pipeline, adapted from the GenoQual github project page [49]

The most recent version of each required package was installed resulting on the changes present in table 4.

Table 4: Original package versions required by GenoQual and the newer updated packages used

Package	Original version	New version
Blast	2.2.31	2.9
seqtk	r75	1.3
FastQC	0.11.4	0.11.9
samtools	1.3.1	1.11
bwa	0.7.12	0.7.17
htseq	0.6.1	0.12.4
numpy	1.11.2	1.19.4
qualimap	2.2	2.2.2a
flash	1.2.11	1.2.11
biopython	1.66	1.78
matplotlib	1.5.3	3.3.3
seaborn	0.7.1	0.11.0
pandas	0.19.1	0.22.0
multiqc	0.7	1.9
qiime	1.9.1	Replaced with qiime2-2020.11
Bio	1.66	1.78

To better organize GenoQual's code, the genoqual.py module has been split into three modules which are identified in Table 5:

Table 5: New GenoQual modules

Module	Function
genoqual.py	Main module that is used to start and finish the script
dataops.py	Contains the DataWrapper and DataCollector classes
externaltools.py	Module containing the preparation and execution of the different packages

By dividing the genoqual.py module into the three parts, the user gets a better understanding of the entire pipeline far more easily, which also allowed for a smoother conversion of the code from the Python 2 version to Python 3.

The flowchart present in Figure 13 shows a sequence of the steps performed by the GenoQual pipeline after the creation of the new modules. The pipeline requires certain initial arguments which are handled by the parse_args() function. These settings are then used by the ConfigParser class, where alongside the QCconf.csv file, prepares the configurations needed to properly run the script. These configurations are then read by the Pipeline class which is divided into three main functions: "prepare", "run" and "finish". The "prepare" function checks whether or not each one of the required external packages is properly installed, aborting the entire process if it finds any problems. Once it's done checking the packages, the "run" function essentially calls and runs each one of the external packages. The results obtained are then compiled by the "finish" function which then generates the final *html* report.

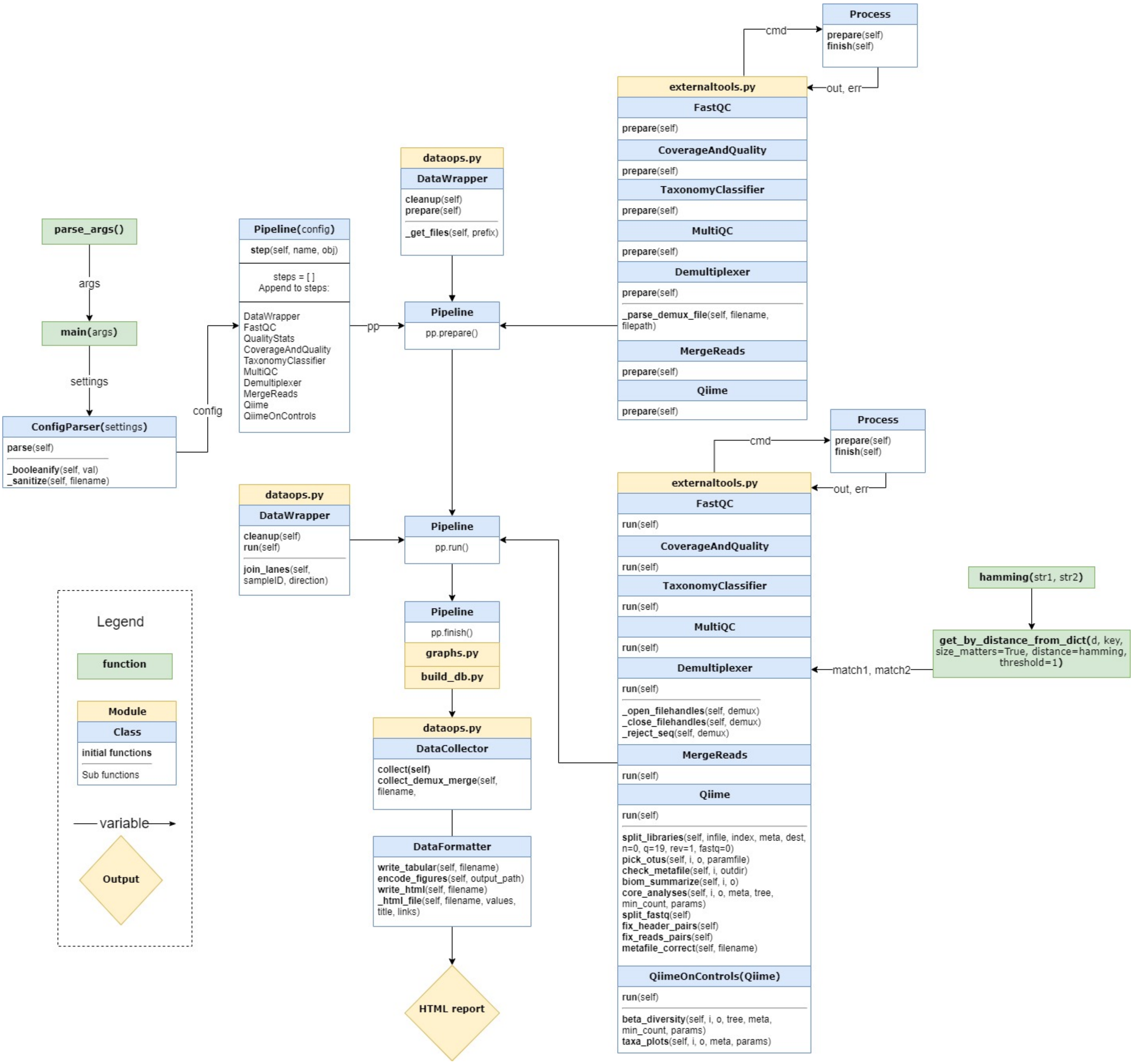


Figure 13: GenoQual pipeline flowchart after the creation of the newer modules and Python 3 conversion

6.3. BCL2FASTQ

Illumina’s bcl2fastq software was used to convert the provided sequencer output files into FASTQ files so that they could be used by the different required software used.

The converter requires a “sample sheet” csv file which lists information regarding the preparation of the samples, the desired length of the reads and a list of indexes for demultiplexing. The sample sheet csv file has a format similar to the one present in Figure 14 and its information is divided into four main parts: “Header”, “Reads”, “Settings” and “Data”.

[Header]
Local Run Manager Analysis Id,23023
Experiment Name,Run#230_B40
Date,22/09/2020
Module,GenerateFASTQ - 2.0.1
Workflow,GenerateFASTQ
Library Prep Kit,Custom
Description,MetaG-Batch40
Chemistry,Default
iemfileversion,4
application,FASTQ Only
[Reads]
251
251
[Settings]
[Data]
Sample_ID,Sample_Name,Description,index,I7_Index_ID,Sample_Project
Batch40,Batch40,,CGCTCATGATCA,CGCTCATGATCA,

Figure 14: Samplesheet.csv

The convertor is then run by simply executing the following command in the terminal:

- bcl2fastq --create-fastq-for-index-reads -o

This command grabs the sequencer’s output folder with all the intensities determined by the sequencer and the samplesheet.csv file and creates a folder, such as the one in Figure 15, that contains paired reads files (R1, R2) and the barcodes indexes (I1) in FASTQ format. Files that do not undergo demultiplexing due to the lack of reliable barcodes are grouped into the “Undertermined” file.

Filename	Filesize
..	
Undetermined_S0_L001_R2_001.fastq.gz	3 931 120 570
Undetermined_S0_L001_R1_001.fastq.gz	3 190 966 832
Undetermined_S0_L001_I1_001.fastq.gz	366 857 395
Batch40_S1_L001_R2_001.fastq.gz	15 323
Batch40_S1_L001_R1_001.fastq.gz	13 984
Batch40_S1_L001_I1_001.fastq.gz	3 413
Stats	
Reports	

Figure 15: bcl2fastq output folder

6.4. QIIME 2

The latest stable QIIME 2 version released at the time of the internship was version 2020.11. Since this version of QIIME 2 isn't compatible with Python versions above 3.6 and since the software is very likely to conflict with other external installed packages, QIIME 2 was installed in its own clean conda environment. This environment was then added to the main GenoQual working environment as a PATH, allowing for GenoQual to run with a Python 3.8 and still be able to run QIIME 2 on its Python 3.6 version.

A metadata table file needs to be provided for QIIME to perform demultiplexing. The metadata file, such as the one in Figure 16, is tab-delimited text file that includes the barcode sequence and primer sequences used for each different sample. QIIME 2 will demultiplex the samples by finding matches between the metadata file and the index FASTQ file.

metadata.csv

```
#SampleID	BarcodeSequence	LinkerPrimerSequence	Description
amsousal.CN32	AGCCTTCGTCGC	AATGATACGGCGACCACCGAGATCTACACGCTAGCCTTCGTCGCTATGGTAATTGTGTGYCAGCMGCCGCGGTAA	515rcbc0
amsousal.CN33	TCCATACCGGAA	AATGATACGGCGACCACCGAGATCTACACGCTTCCATACCGGAATATGGTAATTGTGTGYCAGCMGCCGCGGTAA	515rcbc1
amsousal.CN34	AGCCCTGCTACA	AATGATACGGCGACCACCGAGATCTACACGCTAGCCCTGCTACATATGGTAATTGTGTGYCAGCMGCCGCGGTAA	515rcbc2
amsousal.CN35	CCTAACGGTCCA	AATGATACGGCGACCACCGAGATCTACACGCTCCTAACGGTCCATATGGTAATTGTGTGYCAGCMGCCGCGGTAA	515rcbc3
amsousal.CN36	CGCGCCTTAAAC	AATGATACGGCGACCACCGAGATCTACACGCTCGCGCCTTAAACTATGGTAATTGTGTGYCAGCMGCCGCGGTAA	515rcbc4
amsousal.CN37	TATGGTACCCAG	AATGATACGGCGACCACCGAGATCTACACGCTTATGGTACCCAGTATGGTAATTGTGTGYCAGCMGCCGCGGTAA	515rcbc5
amsousal.CN38	TACAATATCTGT	AATGATACGGCGACCACCGAGATCTACACGCTTACAATATCTGTTATGGTAATTGTGTGYCAGCMGCCGCGGTAA	515rcbc6
amsousal.CN39	AATTTAGGTAGG	AATGATACGGCGACCACCGAGATCTACACGCTAATTTAGGTAGGTATGGTAATTGTGTGYCAGCMGCCGCGGTAA	515rcbc7
amsousal.CN40	GACTCAACCAGT	AATGATACGGCGACCACCGAGATCTACACGCTGACTCAACCAGTTATGGTAATTGTGTGYCAGCMGCCGCGGTAA	515rcbc8
amsousal.CN41	GCCTCTACGTCG	AATGATACGGCGACCACCGAGATCTACACGCTGCCTCTACGTCGTATGGTAATTGTGTGYCAGCMGCCGCGGTAA	515rcbc9
amsousal.907.M1	ACTACTGAGGAT	AATGATACGGCGACCACCGAGATCTACACGCTACTACTGAGGATTATGGTAATTGTGTGYCAGCMGCCGCGGTAA	515rcbc10
amsousal.934.M1	AATTCACCTCCT	AATGATACGGCGACCACCGAGATCTACACGCTAATTCACCTCCTTATGGTAATTGTGTGYCAGCMGCCGCGGTAA	515rcbc11
amsousal.880.M1	CGTATAAATGCG	AATGATACGGCGACCACCGAGATCTACACGCTCGTATAAATGCGTATGGTAATTGTGTGYCAGCMGCCGCGGTAA	515rcbc12
amsousal.837.M1	ATGCTGCAACAC	AATGATACGGCGACCACCGAGATCTACACGCTATGCTGCAACACTATGGTAATTGTGTGYCAGCMGCCGCGGTAA	515rcbc13
amsousal.P145.M1	ACTCGCTCGCTG	AATGATACGGCGACCACCGAGATCTACACGCTACTCGCTCGCTGTATGGTAATTGTGTGYCAGCMGCCGCGGTAA	515rcbc14
amsousal.P147.M1	TTCCTTAGTAGT	AATGATACGGCGACCACCGAGATCTACACGCTTTCCTTAGTAGTTATGGTAATTGTGTGYCAGCMGCCGCGGTAA	515rcbc15
amsousal.P149.M1	CGTCCGTATGAA	AATGATACGGCGACCACCGAGATCTACACGCTCGTCCGTATGAATATGGTAATTGTGTGYCAGCMGCCGCGGTAA	515rcbc16
amsousal.P150.M1	ACGTGAGGAACG	AATGATACGGCGACCACCGAGATCTACACGCTACGTGAGGAACGTATGGTAATTGTGTGYCAGCMGCCGCGGTAA	515rcbc17
```

Figure 16: metadata.csv file

Due to the output of bc2lfastq including an index file that contains the barcode reads associated with each sequence and due to the linker primer sequence being the same one found in the EMP's 16S Illumina Amplicon Protocol [50], the QIIME 2 process used was adapted from QIIME's "Moving Pictures" tutorial [51] since the scripts that are used in that tutorial are meant for data that was sequenced using the EMP protocol. Figure 17 summarizes the steps performed during the QIIME 2 analysis.

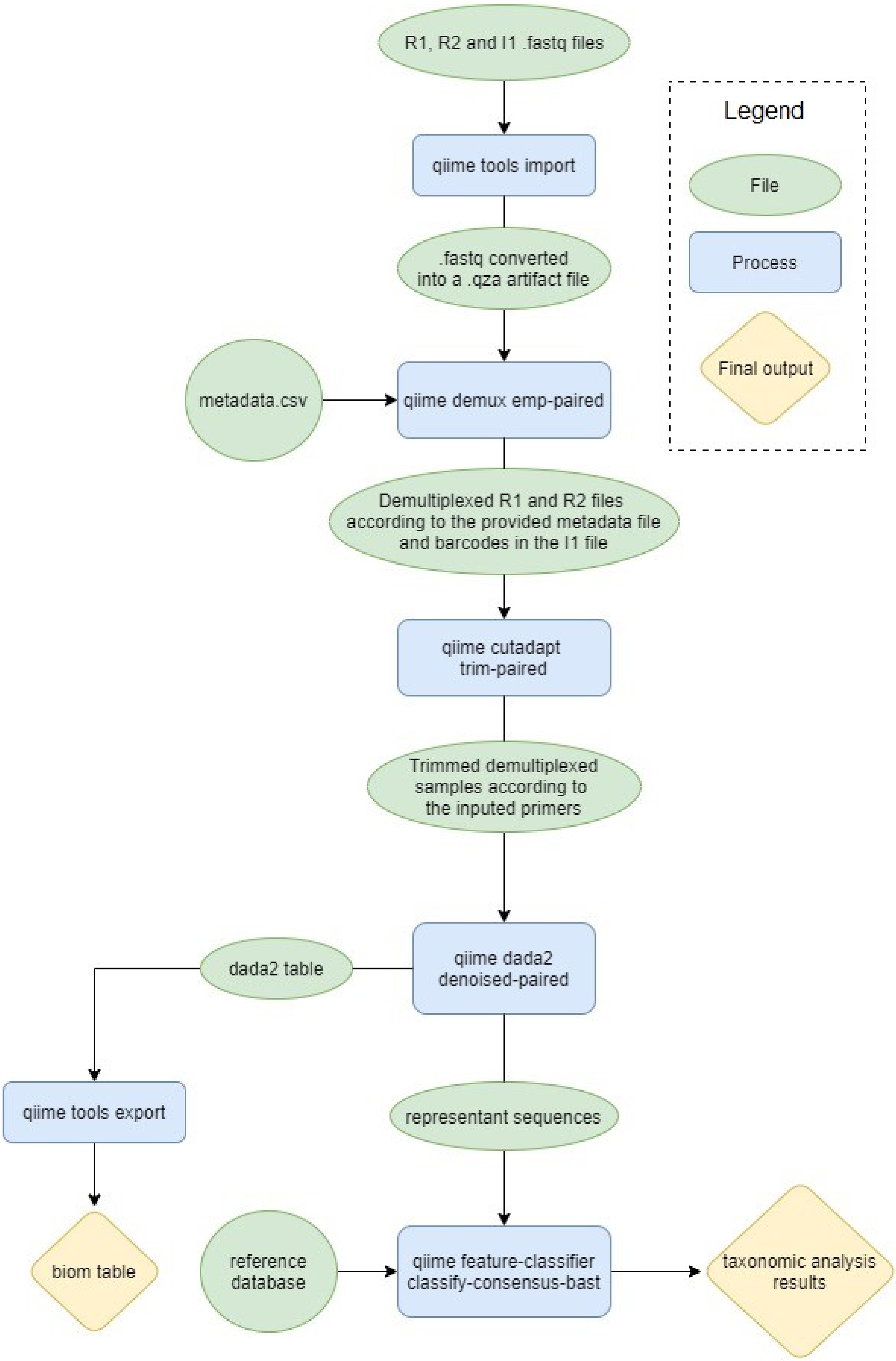


Figure 17: QIIME 2 pipeline flowchart adapted from QIIME’s “Moving Pictures” tutorial

The commands used during the execution of the pipeline were the following:

- qiime tools import \
- --type EMPPairedEndSequences \
- --output-path outputs/emp-paired-end-sequences.qza

Converts the multiplexed FASTQ files and the barcode indexes into a QIIME 2 artifact file [35]

- qiime demux emp-paired \
- --i-seqs outputs/emp-paired-end-sequences.qza \
- --m-barcodes-file inputs/metada.csv \
- --m-barcodes-column BarcodeSequence \
- --o-per-sample-sequences outputs/demux.qza \
- --o-error-correction-details outputs/demux-details.qza \
- --p-golay-error-correction False \
- --verbose

Demultiplexes the reads by using the barcode indexes file and the metadata.csv [52]

- qiime cutadapt trim-paired \
- --i-demultiplexed-sequences outputs/demux.qza \
- --p-cores 6 \
- --p-front-f {ADAPTER1} \
- --p-front-r {ADAPTER2} \
- --o-trimmed-sequences outputs/trimmed.qza \
- --verbose \
- &> logs/trimlog.txt

Uses QIIME 2's cutadapt plugin to trim specific adapter sequences [53]. The --p-front-f and --p-front-r arguments require the R1 adapter sequence and the R2 adapter sequence respectively.

- qiime dada2 denoise-paired \
- --i-demultiplexed-seqs outputs/trimmed.qza \
- --o-table outputs/table_dada2.qza \
- --o-representative-sequences outputs/rep_seqs_dada2.qza \
- --o-denoising-stats outputs/denoising_stats_dada2.qza \
- --p-trunc-len-f {integer} \
- --p-trunc-len-r {integer} \
- --p-n-threads 0 \
- --verbose \
- &> logs/denoiselog.txt

Uses QIIME 2's dada2 plugin to denoise the reads and generate ASV tables [54]. The trunc len length depends on the point where the quality of the reads drops below 30. In the execution of this pipelines, values inbetween 170~210 were used for the R1 reads while values inbetween 150~180 were used for the R2 files since it was around those points where the quality of the reads dropped according to the FastQC quality plots.

- qiime feature-classifier classify-consensus-blast \
- --i-query outputs/rep_seqs_dada2.qza \
- --i-reference-reads taxa/silva-138-99-seqs.qza \
- --i-reference-taxonomy taxa/silva-138-99-tax.qza \
- --o-classification outputs/classifyconsensusBLAST.qza

Assigns taxonomy to query sequences using BLAST+ and a reference database like GreenGenes and SILVA [55].

After executing those commands, two main outputs are produced:

- table_dada2.qza
- classifyconsensusBLAST.qza

The dada2 table file is essentially the ASV table created by the DADA 2 plugin. This table can then be converted into the biom file format by importing it with the following command [56]:

- qiime tools export \
- --input-path outputs/table_dada2.qza \
- --output-path outputs/biomtable

This biom file is extremely important for further downstream analysis. It can also be converted into other types of files by using the “biom convert” QIIME command, resulting on an ASV table with the format found in Table 6:

Table 6: ASV table generated with the "biom convert --to-tsv -i feature-table.biom -o table.tsv" command

```
table.tsv
# Constructed from biom file
#OTU ID amatias.E2.D10.1.1.C    amatias.E2.D10.1.1.F    amatias.E2.D10.1.2.C    amatias.E2.D10.1.2.F
d46e2205f0c6ecf67b51f83d111c509c    0.0 13.0    12.0    0.0 7.0 0.0 0.0 13.0    8.0 6.0 0.0 4.0 0.0 10
e4597f1dce864c29882fa64f616d1a6c    0.0 0.0 25.0    0.0 42743.0 23143.0 43027.0 35800.0 6.0 9.0 41848.
06f825b512d903b9230e1a55d87359ee    12.0    0.0 11.0    8.0 5.0 14.0    0.0 17.0    0.0 8.0 10.0    14
923f521b9cf313f1f95c9367e09bbc1c    9.0 0.0 14.0    0.0 12.0    0.0 7.0 0.0 0.0 0.0 0.0 0.0 0.0 1780.0
4f5efd25dacb5d639316e7291ff6ff8b    14.0    8.0 14.0    0.0 6.0 4.0 0.0 15.0    8.0 7.0 10.0    14.0
daae43be6cf06991f62a085ba8bff3b6    0.0 0.0 0.0 0.0 32.0    0.0 33.0    43.0    0.0 7.0 0.0 27.0    0.
394eda29c886632f514dd94b58381186    0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 5.0 0.0 0.0 1611.0  797.0
32f8fd11d2bee278d609a1d4ab767554    9.0 0.0 0.0 0.0 6.0 0.0 0.0 10.0    0.0 6.0 0.0 0.0 0.0 4292.0  37
0ebb2cf4a2017aeaefadf9ae5062eb62    11.0    0.0 15.0    0.0 27.0    0.0 30.0    0.0 11.0    0.0 20.0
dcba105f35d8ebc9e22269c7491ad3a7    0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 6.0 62.0    14
69a11f927915e2a3ba1f7b9c84486527    0.0 0.0 5.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.
b7b4ecb07e1719860260746a5443092a    0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.
129500150cad877075ed12b0c9424282    0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.
cb8b8a6ee6bcfa115f8b24a45fa1c12a    0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.
04f5947e3f8d4cc43b9a50a6ac7749ab    0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0    0.0 0.0 5.0 0.0 0.
f1860fe716257bd5fd5c4c6a16cf3b95    0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 916.0   982.0
f80166d86a7f15b69b6ac97505299c3a    0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.
e606e6a9dbcbaefd0fe8f53571bbcab0    0.0 0.0 13.0    0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 1570.0  28
cd9401a6bce4a63af516d06d2a843f9d    0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 5.0 0.0 0.0 0.0 787.0   1057.0
948a960b3f069d6eaabc5f61d6e74516    22656.0 18664.0 28026.0 15105.0 0.0 0.0 0.0 0.0 16397.0 22337.0 0.
f9301953e6a6de59f3fb606a562b3cc8    0.0 0.0 9.0 0.0 0.0 0.0 0.0 0.0 0.0 2.0 0.0 0.0 0.0 0.0 0.0 0.0 0.
5648dccee530d68ceb3e4d7d22cf8756    0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 37.0    0.
b626779bd5dfa945ddc31fd5e5070a19    0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.
7b28c20e72c6c95b3e604f0849245770    0.0 0.0 0.0 0.0 0.0 0.0 4.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.
1dde417c45171347b0ec07088aa49839    0.0 0.0 0.0 0.0 12.0    0.0 0.0 0.0 0.0 0.0 0.0 12.0    0.0 0.0 0.
2747bd94a19fbfa82f1e90e326d2f100    0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.
224102d609b83a172b11bd9115107f16    0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 278.0   555.0
9fd2ab2f88f10ef6d386289b4ce9b85d    0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.
a12a086985ef700cc8f5a0bd3c7b45c1    16527.0 14066.0 20117.0 11962.0 2.0 0.0 0.0 0.0 11905.0 17101.0 0.
```

This table shows the number of unique 100% clustered sequences that each one of the samples have.
The classifyconsensusBLAST.qza file on the other hand, contains information regarding the taxonomy. The .qza file was then converted into a .qzv file to allow visualization of the taxonomy table which has the format found in Table 7.

Table 7: Taxonomy table

Feature ID	Taxon	Consensus
#q2:types	categorical	categorical
d46e2205f0c6ecf67b51f83d111c509c	d__Bacteria; p__Proteobacteria; c__Gammaproteobacteria; o__Enterobacterales; f__Enterobacteriaceae; g__Escherichia-Shigella	1.0
e4597f1dce864c29882fa64f616d1a6c	d__Bacteria; p__Bacteroidota; c__Bacteroidia; o__Bacteroidales; f__Muribaculaceae; g__Muribaculum; s__uncultured_bacterium	0.9
06f825b512d903b9230e1a55d87359ee	d__Bacteria; p__Firmicutes; c__Bacilli; o__Lactobacillales; f__Streptococcaceae; g__Streptococcus	1.0
923f521b9cf313f1f95c9367e09bbc1c	d__Bacteria; p__Firmicutes; c__Negativicutes; o__Veillonellales-Selenomonadales; f__Veillonellaceae; g__Veillonella; s__uncultured_bacterium	0.9
4f5efd25dacb5d639316e7291ff6ff8b	d__Bacteria; p__Proteobacteria; c__Gammaproteobacteria; o__Burkholderiales; f__Neisseriaceae; g__Neisseria; s__Neisseria_perflava	0.8
daae43be6cf06991f62a085ba8bff3b6	d__Bacteria; p__Bacteroidota; c__Bacteroidia; o__Bacteroidales; f__Muribaculaceae; g__Muribaculaceae; s__uncultured_bacterium	1.0
394eda29c886632f514dd94b58381186	d__Bacteria; p__Proteobacteria; c__Gammaproteobacteria; o__Pasteurellales; f__Pasteurellaceae; g__Haemophilus; s__uncultured_bacterium	0.9
32f8fd11d2bee278d609a1d4ab767554	d__Bacteria; p__Bacteroidota; c__Bacteroidia; o__Bacteroidales; f__Prevotellaceae; g__Prevotella; s__uncultured_bacterium	1.0
0ebb2cf4a2017aeaefadf9ae5062eb62	d__Bacteria; p__Bacteroidota; c__Bacteroidia; o__Bacteroidales; f__Muribaculaceae; g__Muribaculaceae; s__uncultured_bacterium	1.0
dcba105f35d8ebc9e22269c7491ad3a7	d__Bacteria; p__Proteobacteria; c__Gammaproteobacteria; o__Xanthomonadales; f__Xanthomonadaceae; g__Stenotrophomonas	1.0
69a11f927915e2a3ba1f7b9c84486527	d__Bacteria; p__Proteobacteria; c__Gammaproteobacteria; o__Burkholderiales; f__Sutterellaceae; g__Parasutterella; s__uncultured_organism	1.0
b7b4ecb07e1719860260746a5443092a	d__Bacteria; p__Bacteroidota; c__Bacteroidia; o__Bacteroidales; f__Prevotellaceae; g__Prevotella; s__uncultured_bacterium	0.9
129500150cad877075ed12b0c9424282	d__Bacteria; p__Bacteroidota; c__Bacteroidia; o__Bacteroidales; f__Muribaculaceae; g__Muribaculaceae; s__uncultured_bacterium	1.0
cb8b8a6ee6bcfa115f8b24a45fa1c12a	d__Bacteria; p__Bacteroidota; c__Bacteroidia; o__Bacteroidales; f__Muribaculaceae; g__Muribaculaceae; s__uncultured_bacterium	1.0
04f5947e3f8d4cc43b9a50a6ac7749ab	d__Bacteria; p__Bacteroidota; c__Bacteroidia; o__Bacteroidales; f__Muribaculaceae; g__Muribaculaceae; s__uncultured_bacterium	1.0
f1860fe716257bd5fd5c4c6a16cf3b95	d__Bacteria; p__Proteobacteria; c__Gammaproteobacteria; o__Burkholderiales; f__Neisseriaceae; g__Neisseria; s__uncultured_bacterium	0.8
f80166d86a7f15b69b6ac97505299c3a	d__Bacteria; p__Bacteroidota; c__Bacteroidia; o__Bacteroidales; f__Muribaculaceae; g__Muribaculaceae; s__uncultured_bacterium	1.0

Similar to the ASV table, the taxonomy file table contains the ID of each unique 100% OTU cluster. This time around however, each cluster was classified taxonomy and each cluster was given a consensus score.
Those tables can then be converted into other types of files in order to obtain different types of visual results. A useful bar plot can be generated with the qiime taxa barplot command [57] which has the format found in Figure 18.

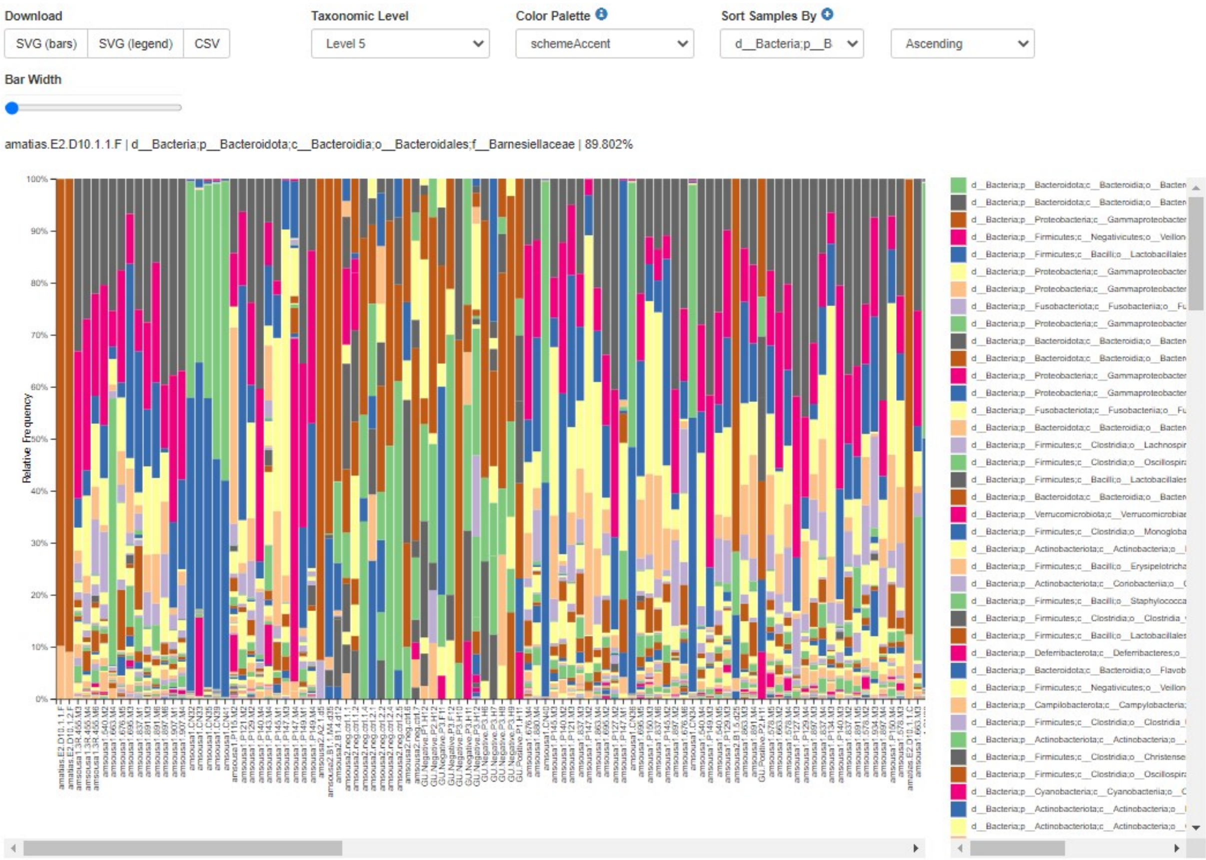


Figure 18: Taxa barplot output

This bar plot HTML file is interactive and allows the user to visualize the taxonomy of each sample through various different taxonomic levels.

6.5. Kraken 2

To run a Kraken 2 classification analysis two things are needed: a database folder and the reads to be classified.

For the Kraken 2 tests, the samples used were the same that were obtained during QIIME 2's trimming steps since the files at that point were already demultiplexed and had their adapters trimmed. However, due to QIIME 2 working with.qza format files, the files had to be converted back to FASTQ files using the "qiime tools extract" command [58].

Minikraken was used as the database due to its small size, low requirements, and general faster speed. However, for more serious analysis, the main Kraken 2 database files should be considered instead.

Kraken 2 uses the following command structure to classify paired read files:

- Kraken2 --db {database folder directory} --threads {number of threads to use} --report {kraken2 report output filename} --paired {R1 FASTQ file} {R2 FASTQ file}

This command creates a kraken2 report file found in Figure 19:

amatias.E2.D10.1.1.C.kreport2

```
 0.62  342 342 U   0   unclassified
99.38  55032   260 R   1   root
98.87  54751   0   R1  131567    cellular organisms
98.87  54751   102 D   2       Bacteria
86.70  48010   0   D1  1783270       FCB group
86.70  48009   0   D2  68336           Bacteroidetes/Chlorobi group
86.70  48008   28  P   976           Bacteroidetes
86.54  47922   73  C   200643              Bacteroidia
86.41  47849   234 O   171549                Bacteroidales
85.25  47207   0   F   2005519                 Barnesiellaceae
85.25  47207   0   G   397864                    Barnesiella
85.25  47207   0   S   397865                      Barnesiella viscericola
85.25  47207   47207   S1  880074                      Barnesiella viscericola DSM 18177
 0.52  290 0   F   2005473             Muribaculaceae
 0.52  290 0   G   1918540               Muribaculum
 0.52  290 290 S   1796646                 Muribaculum intestinale
 0.08  45  0   F   2005525             Tannerellaceae
```

Figure 19: Kraken2 report

To better visualize the results, the Pavian software was used. Pavian is a R package that generates a HTML report with various info.

These kraken report files can be then used by Pavian (via R) to generate a HTML report with summaries and sankey diagrams (Figures 22-24) [59].

Figure 20 shows the sample set summary that lists the number of reads of each sample, the percentage of reads that Kraken was able to classify and the percentage of the type of organisms found.

Sample set summary

Classification summary | Raw read numbers | Sample information

Show 15 rows | CSV | Print | Copy | Column visibility

Search:

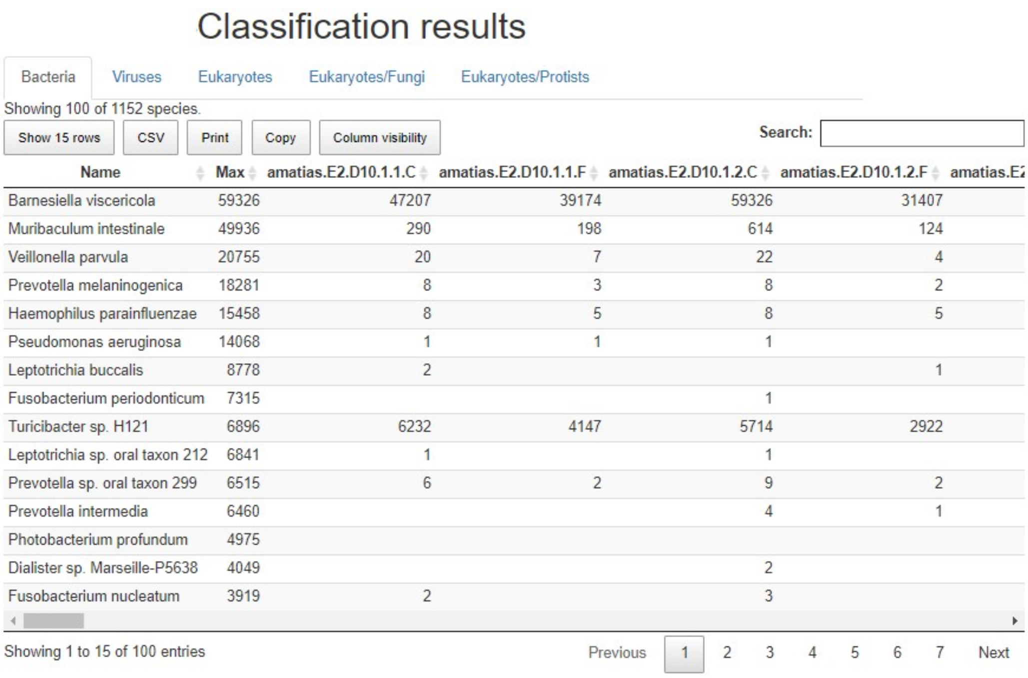
Name	Number of raw reads	Classified reads	Chordate reads	Artificial reads	Unclassified reads	Microbial reads	Bacterial reads	Viral reads	Fungal reads	Protozoan reads
amatias.E2.D10.1.1.C	55,374	99.38%	0%	0%	0.6176%	98.91%	98.87%	0.001806%	0%	0%
amatias.E2.D10.1.1.F	45,044	99.38%	0.00222%	0%	0.6238%	98.54%	98.45%	0%	0%	0%
amatias.E2.D10.1.2.C	68,361	99.31%	0.001463%	0%	0.689%	98.62%	98.57%	0.001463%	0%	0%
amatias.E2.D10.1.2.F	35,155	99.45%	0%	0%	0.5547%	99.2%	99.19%	0%	0%	0%
amatias.E2.D10.2.1.C	55,483	99.44%	0.001802%	0%	0.5587%	98.87%	98.83%	0%	0%	0%
amatias.E2.D10.2.1.F	30,108	99.31%	0%	0%	0.6875%	98.74%	98.66%	0%	0%	0%
amatias.E2.D10.2.2.C	56,951	99.44%	0.001756%	0%	0.5566%	98.82%	98.77%	0.003512%	0%	0%
amatias.E2.D10.2.2.F	48,563	99.08%	0.002059%	0%	0.9205%	97.98%	97.89%	0%	0%	0%
amatias.E2.D2.1.1.F	35,725	99.4%	0.002799%	0%	0.6046%	98.89%	98.85%	0%	0%	0%
amatias.E2.D2.1.2.F	51,428	99.54%	0.001944%	0%	0.4628%	98.87%	98.81%	0.003889%	0%	0%
amatias.E2.D2.2.1.F	56,851	99.37%	0.001759%	0%	0.635%	98.86%	98.81%	0.003518%	0%	0%
amatias.E2.D2.2.2.F	54,814	99.45%	0%	0%	0.5546%	98.11%	98%	0.001824%	0%	0%
amatias.NC.24.07.20	515	88.16%	0%	0%	11.84%	87.77%	87.77%	0%	0%	0%
amsousa1.3R.455.M2	49,275	99.08%	0%	0%	0.9193%	98.64%	98.59%	0%	0%	0%
amsousa1.3R.455.M3	54,534	99.21%	0.007335%	0%	0.7867%	98.62%	98.58%	0.001834%	0%	0%

Showing 1 to 15 of 99 entries

Previous 1 2 3 4 5 6 7 Next

Figure 20: Pavian's sample set summary

Figure 21 shows the classification results which lists the number of times a specific specie is present in every single one of the analyzed samples.


Classification results

Bacteria | Viruses | Eukaryotes | Eukaryotes/Fungi | Eukaryotes/Protists

Showing 100 of 1152 species.

Show 15 rows | CSV | Print | Copy | Column visibility

Search:

Name	Max	amatias.E2.D10.1.1.C	amatias.E2.D10.1.1.F	amatias.E2.D10.1.2.C	amatias.E2.D10.1.2.F	amatias.E2
Barnesiella viscericola	59326	47207	39174	59326	31407	
Muribaculum intestinale	49936	290	198	614	124	
Veillonella parvula	20755	20	7	22	4	
Prevotella melaninogenica	18281	8	3	8	2	
Haemophilus parainfluenzae	15458	8	5	8	5	
Pseudomonas aeruginosa	14068	1	1	1		
Leptotrichia buccalis	8778	2			1	
Fusobacterium periodonticum	7315			1		
Turicibacter sp. H121	6896	6232	4147	5714	2922	
Leptotrichia sp. oral taxon 212	6841	1		1		
Prevotella sp. oral taxon 299	6515	6	2	9	2	
Prevotella intermedia	6460			4	1	
Photobacterium profundum	4975					
Dialister sp. Marseille-P5638	4049			2		
Fusobacterium nucleatum	3919	2		3		

Showing 1 to 15 of 100 entries

Previous 1 2 3 4 5 6 7 Next

Figure 21: Pavian's classification's results

Figure 22 shows the Sankey visualization. This diagram provides the taxonomy of each of the species found in a sample. The x-axis represents the domain, phylum, class, order, family, genus, and type species. The several colored lines each represent a classified specie. The wider the line. the higher is the amount of that specific specie in the sample.

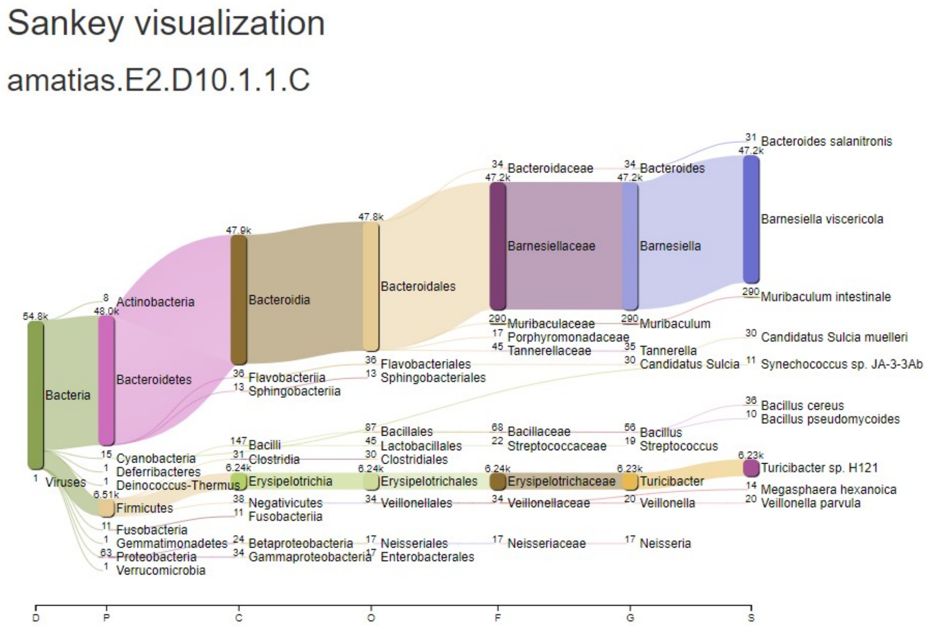


Figure 22: Pavian's sankey visualization of one of the samples

Bracken was also used on the kraken2 report files as a post processing tool to estimate the abundance. The reports generated by Bracken were then also used on Pavian to generate similar results of the previous three figures.

6.6. WRAPPING THINGS UP AND TAKE HOME MESSAGE

With the QIIME 2 and Kraken 2 tests being done and offering suitable results, the pipelines adapted during the work can be now introduced into the GenoQual pipeline, allowing for both 16S/18S/ITS and WGS analysis depending on the data provided by the user.

Figure 23 shows a flowchart that summarizes the entire analysis process used during the execution of these tests. The process starts with the conversion of the Illumina sequenced samples into FASTQ files using bcl2fastq. Afterwards, the samples are converted into QIIME 2 artifacts, where they undergo through demultiplex and trimming. Amplicon sequence samples are then denoised through QIIME 2's dada2 plugin and at the end are classified using the BLAST plugin. For WGS samples, the trimmed artifact files are converted back to FASTQ files. The files are then classified using Kraken 2 with/out Bracken. The generated kraken report files are then compiled into *html* reports using Pavian.

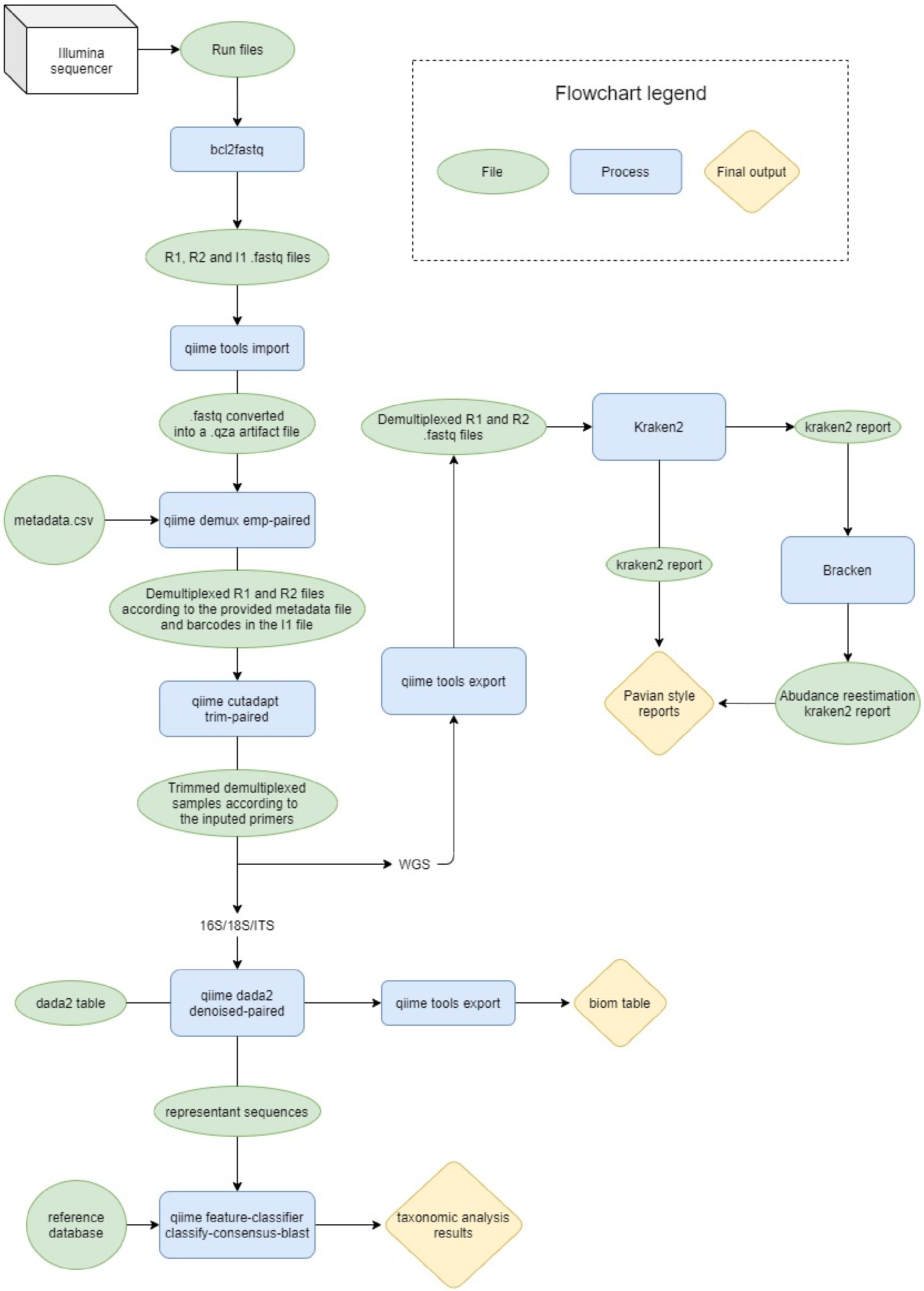


Figure 23: Flowchart of the pipeline

7. CONCLUSÕES

Relativamente a algumas das tarefas planeadas para este estágio, particularmente na atualização da pipeline do GenoQual de modo incluir novas funcionalidades e na adaptação do GenoQual para conseguir correr independentemente do servidor da *Galaxy*, foram impossíveis de concluir como proposto devido a limitações de tempo.

Em relação à parte do trabalho sobre o QIIME 2, houve vários problemas que ocorreram durante o processo, particularmente em relação ao facto que as *reads* das amostras estavam a ser completamente filtradas durante o passo de *denoise* da pipeline o que não se verificava em comparação a outros trabalhos publicados online e à execução da pipeline com outros tipos de samples. Um dos ficheiros produzidos durante o *denoising*, denoising_stats_dada2.qza, foi usado para verificar a existência deste erro. Só após várias tentativas com alterações no fluxo de trabalho usado, foi possível obter resultados apropriados. Graças a esses resultados positivos, o QIIME 2 pode agora ser incluído na pipeline do GenoQual e substituir o código original responsável pelo uso do QIIME 1.

Durante a parte do QIIME 2 do trabalho, os erros que estavam a ocorrer foram reportados e foi proposto a troca do foco do QIIME 2 para o Kraken 2. Ao estudar ambos o QIIME 2 e Kraken 2, o GenoQual poderia incorporar ambos processos no seu código, deixando ultimamente à escolha do utilizador no tipo de analises desejado. O GenoQual pode assim então parar de apenas realizar analises de sequências de *amplicon* e começar também a poder executar analises de WGS. Ao contrário do que o que aconteceu com o QIIME 2, os testes do Kraken 2 correram com menos problemas porque o software funcionou logo corretamente e apresentou resultados razoáveis.

Com ambos os testes a terminarem com resultados satisfatórios semelhantes a outros encontrados *online*, o fluxograma presente na Figura 25 foi apresentado como uma proposta para a implementação do QIIME 2 e do Kraken 2 no pipeline do GenoQual, com um caminho para análises de sequências *amplicon* usando as ferramentas de classificação de taxonomia do QIIME 2 com a apresentação dos conteúdos dos ficheiros .qzv como relatórios *html* e um outro caminho para análises WGS usando o Kraken 2 com/sem o Bracken com os resultados apresentados em relatórios *html* do Pavian.

7. CONCLUSIONS

Relatively to some of the tasks planned for this internship, mainly the updating of the GenoQual pipeline with the newer functionalities and the adaption of GenoQual to run independently of the Galaxy server, they were impossible to wrap up properly due to time constraints, being instead postponed for future work.

For the QIIME 2 part of the work, there were several issues that occurred during the process since most of the reads were getting completely filtered during the denoising step of the pipeline when compared to other works online and execution with other types of samples. One of the files produced during the denoising, denoising_stats_dada2.qza, was used to see the existence of the error. Only after several attempts alongside with a change of the QIIME 2 workflow used, it was finally possible to obtain suitable results. With these positive results, QIIME 2 can now be added into the GenoQual pipeline and replace the original QIIME 1 code that was previously used.

During the QIIME 2 part of the work, the errors were reported, and it was proposed to temporarily switch from QIIME 2 to Kraken 2. By researching both QIIME 2 and Kraken 2, GenoQual could include both workflows into its code, leaving up to the choice of the user the desired type of analysis. GenoQual could move away from doing only amplicon sequence analysis and also start performing WGS type analysis. Unlike what happened with QIIME 2, Kraken 2's tests went far more smoothly since the software worked right away without any problems and delivered reasonable results.

With both tests ending with satisfactory results similar to ones found online, the flowchart present in Figure 25 was presented as a proposal for QIIME 2's and Kraken 2's implementation on the GenoQual pipeline with a branch for amplicon sequence analysis using QIIME 2's taxonomy classifying tools and the presentation of the contents of the .qzv files as HTML reports and WGS analysis using Kraken 2 with/without Bracken with the results presented in Pavian style *html* reports.

8. BIBLIOGRAPHY

[1] https://gulbenkian.pt/ciencia/homepage/igc/missao-visao-e-valores/ Accessed 10th February 2021

[2] https://elixir-europe.org/about-us Accessed 10th February 2021

[3] https://biodata.pt/ Accessed 10th February 2021

[4] http://ce3.igc.gulbenkian.pt/igc/ Accessed 10th February 2021

[5] https://blast.ncbi.nlm.nih.gov/Blast.cgi Accessed 29th December 2020

[6] Hirak Ranjan, Surajit Das, chapter 4 – Molecular methods for studying microorganisms from atypical environments. In *Microbiology of Atypical Environments Volume 45*, 89-122, Volke Gurtler & Jack T. Trevors, 2018

[7] https://www.illumina.com/company/about-us/fact-sheet.html Accessed 29th December 2020

[8] https://www.illumina.com/systems/sequencing-platforms/miseq.html Accessed 29th December 2020

[9] https://support.illumina.com/sequencing/sequencing_software/bcl2fastq-conversion-software.html Accessed 29th December 2020

[10] https://docs.conda.io/projects/conda/en/latest/. Accessed 29th December 2020

[11] https://en.wikipedia.org/wiki/Python_(programming_language) Accessed 29th December 2020

[12] https://www.python.org/about/ Accessed 29th December 2020

[13] https://github.com/igcbioinformatics/genoqual Accessed 29th December 2020

[14] https://github.com/s-andrews/FastQC Accessed 29th December 2020

[15] https://rtsf.natsci.msu.edu/genomics/tech-notes/fastqc-tutorial-and-faq/ Accessed 29th December 2020

[16] https://dnacore.missouri.edu/PDF/FastQC_Manual.pdf Accessed 5th March 2021

[17] https://multiqc.info/ Accessed 29th December 2020

[18] https://astrobiomike.github.io/misc/amplicon_and_metagen Accessed 7th February 2021

[19] https://benjjneb.github.io/dada2/index.html Accessed 7th February 2021

[20] https://www.zymoresearch.com/blogs/blog/16s-sequencing-vs-shotgun-metagenomic-sequencing Accessed 7th February 2021

[21] https://igcbioinformatics.github.io/biomeshinycourse/pages/dada2/Biodata.ptCrashCourses.html Accessed 7th February 2021

[22] https://figshare.com/articles/dataset/Pipeline_for_16S_rRNA_processing_using_DADA2/6997253 Accessed 7th February 2021

[23] Callahan, B. J., McMurdie, P. J., Rosen, M. J., Han, A. W., Johnson, A. J. A., & Holmes, S. P. (2016). *DADA2: High-resolution sample inference from Illumina amplicon data. Nature Methods, 13(7)*, 581–583. https://doi.org/10.1038/nmeth.3869

[24] https://bioconductor.org/packages/devel/bioc/vignettes/dada2/inst/doc/dada2-intro.html Accessed 7th February 2021

[25] https://www.arb-silva.de/ Accessed 29th December 2020

[26] https://greengenes.secondgenome.com/ Accessed 29th December 2020

[27] http://qiime.org/index.html Accessed 29th December 2020

[28] http://qiime.org/scripts/split_libraries_fastq.html Accessed 29th December 2020

[29] http://qiime.org/scripts/pick_open_reference_otus.html Accessed 29th December 2020

[30] http://qiime.org/scripts/validate_mapping_file.html Accessed 29th December 2020

[31] http://qiime.org/scripts/core_diversity_analyses.html Accessed 29th December 2020

[32] http://qiime.org/scripts/beta_diversity_through_plots.html Accessed 29th December 2020

[33] http://qiime.org/scripts/summarize_taxa_through_plots.html Accessed 29th December 2020

[34] https://qiime2.org/ Accessed 29th December 2020

[35] https://docs.qiime2.org/2020.11/tutorials/importing/ Accessed 29th December 2020

[36] https://view.qiime2.org/ Accessed 29th December 2020

[37] https://docs.qiime2.org/2020.11/tutorials/overview/ Accessed 29th December 2020

[38] https://docs.qiime2.org/2020.11/plugins/available/dada2/ Accessed 29th December 2020

[39] https://ccb.jhu.edu/software/kraken2/index.shtml Accessed 30th January 2021

[40] https://github.com/DerrickWood/kraken2/wiki/Manual Accessed 30th January 2021

[41] https://ccb.jhu.edu/software/kraken2/index.shtml?t=downloads Accessed 30th January 2021

[42] https://github.com/jenniferlu717/Bracken Accessed 30th January 2021

[43] https://www.putty.org/ Accessed 29th December 2020

[44] https://github.com/tmux/tmux Accessed 29th December 2020

[45] https://docs.qiime2.org/2020.11/install/native/#install-qiime-2-within-a-conda-environment Accessed 29th December 2020

[46] https://filezilla-project.org/ Accessed 29th December 2020

[47] https://www.jetbrains.com/pycharm/ Accessed 29th December 2020

[48] https://github.com/Genomic-Unit-IGC/genoqual Accessed 17th April 2021

[49] https://github.com/igcbioinformatics/genoqual/blob/master/genoqual.xml Accessed 29th December 2020

[50] https://www.earthmicrobiome.org/protocols-and-standards/16s/ Accessed 7th February 2021

[51] https://docs.qiime2.org/2020.11/tutorials/moving-pictures/ Accessed 30th January 2021

[52] https://docs.qiime2.org/2020.11/plugins/available/demux/emp-paired/ Accessed 30th January 2021

[53] https://docs.qiime2.org/2020.11/plugins/available/cutadapt/trim-paired/ Accessed 30th January 2021

[54] https://docs.qiime2.org/2020.11/plugins/available/dada2/denoise-paired/ Accessed 30th January 2021 Accessed 30th January 2021

[55] https://docs.qiime2.org/2020.11/plugins/available/feature-classifier/classify-consensus-blast/ Accessed 30th January 2021

[56] https://forum.qiime2.org/t/how-does-an-asv-table-looks-like/10552/3 Accessed 30th January 2021

[57] https://docs.qiime2.org/2020.11/plugins/available/taxa/barplot/ Accessed 30th January 2021

[58] https://docs.qiime2.org/2020.11/tutorials/exporting/ Accessed 30th January 2021

[59] https://github.com/fbreitwieser/pavian Accessed 30th January 2021