East African Community One People, One Destiny BNITM Bernhard Nocht Institute for Tropical Medicine	**EAC Mobile Lab**	**STANDARD OPERATING PROCEDURE**	**Effective Date**	**SOP-Number : SOP_EACML_ AMR_Bioinf _047_V1**
			Page 1 **of** 34	**Date: 16th May 2023**
Title: NGS-driven Antimicrobial Resistance Analysis				

Dr. Julien A. Nguinkal
Department of Infectious Disease Epidemiology
Bernhard Nocht Institute for Tropical Medicine
Hamburg, Germany

Date: May 2023

Reviewed/Approved Date: May 2023

This SOP has been read and understood by:

Name	Signature	Date
1.		
2.		
3.		
4.		
5.		
6.		
7.		
8.		
9.		
10.		
11.		
12.		

INTRODUCTION

This guidance document defines the standard operating procedures (SOPs) for bacterial next-generation sequencing (NGS) analysis, including antimicrobial resistance (AMR) genotyping, as well as data processing and reporting. It provides a framework and a comprehensive guide for the analysis of AMR characterisation using bacterial NGS data and bioinformatics. Ultimately, this workflow will use a Nextflow DSL2 pipeline to automate the analysis of fastq files, genome assembly, annotation, and identification of antimicrobial resistance genes (ARG) in bacterial samples. The pipeline is optimized for AMR surveillance data of the Global Antimicrobial Surveillance System (GLASS) priority pathogens.

The EAC-RNPHRL for AMR Surveillance Network for Communicable Diseases and AMR prioritizes the following pathogens and specimens for AMR surveillance:

- *Enterococcus species*
- *Staphylococcus aureus*
- *Escherichia coli*
- *Klebsiella species*
- *Acinetobacter species*
- *Pseudomonas species*
- *Streptococcus species*
- *Salmonella enterica*

The sequencing data to be analyzed will primarily come from Illumina's iSeq100 and NextSeq1000 platforms. Both are popular Illumina sequencing platforms that are commonly used for short read sequencing applications. Short read sequencing is the process of creating a large number of highly accurate short DNA sequences, typically ranging from 50 to 300 base pairs in length.

The iSeq100 is a compact and affordable desktop (benchtop) sequencer that is well-suited for small to medium-sized sequencing projects, such as bacterial strain identification or AMR surveillance. It uses Illumina's proven sequencing-by-synthesis (SBS) technology to generate up to 1.5 billion reads per run with a maximum read length of 150 bp. This platform is therefore ideal for targeted resequencing, metagenomics, amplicon sequencing, and small genome sequencing projects. Furthermore, it is compatible with quick library preparation workflows, making it an efficient choice for bacterial genome sequencing projects with limited time and resources.

The NextSeq1000/2000 is a more powerful and flexible sequencer with higher throughput that is capable of producing up to 1.2 terabytes of data per run with a maximum output of 400 million reads and a read length of up to 150 bp. This platform is ideal for a broad range of applications, including whole-genome sequencing, transcriptome profiling, etc.

Figure 1: portfolio of *Illumina* sequencing platforms

This SOP is a support document for the Bioinformatics AMR Training for the EAC Regional Network of Public Health Reference Laboratories (EAC-RNPHRL) for Communicable Diseases and Antimicrobial Resistance network. The SOPs cover various aspects of NGS-based AMR surveillance, such as library preparation, sequencing, data analysis, and reporting standards. The SOPs detail each step, including the materials and equipment needed, the procedures to be followed, and the quality control measures to be implemented. The document emphasises the importance of maintaining high quality standards throughout the AMR surveillance process based on NGS. It outlines how to ensure the accuracy and reliability of results, such as thorough regular quality control checks, the use of appropriate bioinformatics tools, and strict data analysis protocols.

This document provides standard guidance in the following:

1. Library Preparation:

- Preparing the DNA library using a suitable library preparation kit according to the manufacturer's instructions.
- Quality standards for bacterial DNA sequencing

2. Sequencing:

- Setting up the sequencing run according to the desired parameters, such as read length and sequencing depth.
- Expected quality metrics.

3. Quality Control:

- Assessing the quality of the sequencing reads using quality control software such as FastQC and AQUAMIS
- Trimming/Filtering the reads to remove low-quality bases and adapter sequences using software such as fastp.

4. Genome Assembly:

- Assemble the reads into contigs using a suitable genome assembly program such as SPAdes or MEGAHIT.
- Evaluate the quality of the assembled genome using software such as QUAST and CheckM.

5. Annotation:

- Annotate the assembled genome using a suitable annotation program such as Prokka or Bakta.
- In addition, evaluate the quality of the annotation using software such as BUSCO or CheckM.

6. Analysis of Antimicrobial Resistance:

- Identify potential antimicrobial resistance genes in the annotated genome using software such as amrFinder, TORMES or starAMR
- Determine the presence or absence of resistance genes based on a suitable threshold such as >97% sequence identity and >90% coverage.
- Assess the significance of the resistance genes based on the type of gene and its prevalence in the samples.

7. Reporting:

- Generate a report summarising the sequence analysis results, including the quality of the sequencing reads, genome assembly, and antimicrobial resistance genes detected.
- Interpretation of the results in the context of the microbial species, microbiological phenotypes, and the local epidemiology of AMR.

DESCRIPTION OF METHODS

Prerequisites:

- A computer with Linux operating system installed (tested with Linux Mint 2.11 and Ubuntu 22.04)
- Min. 32 RAM and 1 TB storage

Equipment and Software:

- Nextflow
- Conda/mamba
- FastQC/AQUAMIS
- TORMES
- AmrFinder
- starAMR
- Snippy

Materials:

- Fastq files containing bacterial raw sequencing data
- CARD database
- ResFinder, PointFinder, and PlasmidFinder databases
- Kraken2 database
- Taxonkit database
- Mash database
- Scientific name of the species
- Multi-locus sequence typing (MLST) scheme for the species
- Reference genomes (RefSeq)

Standard quality metrics

Quality control is crucial in bacterial sequencing for AMR analysis as it can affect the accuracy and reliability of the results. Here are some quality control metrics that are important for Iseq100 sequencing [MOU1] for AMR analysis:

1. Library quality:

The library quality refers to the overall quality of the DNA library, including the distribution of fragment sizes and the amount of adapter dimers present. A high-quality library will have a tight distribution of fragment sizes, with a low amount of adapter dimensions, which leads to consistent sequencing coverage across the genome. It is important to assess the quality of the library at multiple steps in the library preparation process to ensure that the library meets the desired quality standards for Illumina sequencing on the iSeq100/NextSeq1000/2000 platforms.

2. Insert size:

This refers to the size of the DNA fragments that are sequenced. The insert size should be consistent with the library preparation protocol and the sequencing platform. For iSeq 100 [MOU3] sequencing, the recommended insert size is typically between 300 and 600 bp.

3. Genome coverage:

Genome coverage, also known as sequencing depth, refers to the number of times a base in the genome is sequenced on average. A minimum sequencing depth of ≥30x means that each base in the genome has been sequenced, on average, at least 30 times. This level of sequencing depth ensures a high level of coverage and minimizes the risk of sequencing errors in the data. The coverage depends on the read length, the genome size and the total number of reads generated.

For example, if a genome is 3 million base pairs long, the read length is 150 bp and a total of 2 million reads are generated, then the sequencing depth would be (2 million bp x 150 bp) / 3 million bp = 300 million bp / 3 million bp = 100x coverage.

The general formulas is:

Coverage = (Total bases sequenced x read length) / Genome size

In the context of AMR profiling, high coverage is particularly important for several reasons:

- **Detection of low-frequency resistance mutations**: High coverage can detect low-frequency resistance mutations that may be missed at lower coverage. This is particularly important in the context of AMR, where resistance can arise through the acquisition of mutations that occur at low frequencies.

- **Reduction of sequencing errors:** High coverage helps to reduce the impact of sequencing errors on the detection and characterization of ARGs. With sufficient coverage, we improve the accuracy of the sequencing data by reducing the effect of random sequencing errors that can lead to false positives or false negatives in the identification of resistance mutations.

- **Genome assembly**: High coverage can help to achieve high-quality genomes assemblies. By generating more reads, we can resolve complex regions of the genome, resulting in a more complete and accurate assembly. This is particularly important for the detection and annotation of resistance genes, as they may be located in regions of the genome that are difficult to assemble or annotate.

- **Genome annotation**: By providing more reads, high coverage can help to identify lowly expressed ARGs and can enable the detection of novel genes or gene variants

that may be missed at lower coverage. This is particularly crucial for the identification and annotation of resistance genes, which can be challenging due to their variable sequence and expression patterns.

❖ **Characterization of resistance mechanisms**: High coverage can enable the identification of specific resistance mechanisms, such as mobile genetic elements (transposons, plasmid, integrons) or point mutations.

⇒ **The recommended sequencing depth of bacterial genomes in the context of AMR detection is coverage ≥40x.** For example, if we are going to sequence an *E. coli* isolate with a genome size of five Mbp at 40x, we will need at least 1.5 million paired-end reads of length 150 bp.

Total number of reads = (5,000,000 bp * 40) / 150 bp = ~1, 500,000 paired-end reads.

However, considering the cost-benefit tradeoff is essential when choosing the appropriate sequencing depth for a project. While higher sequencing depth can increase accuracy and provide greater confidence in variant calling, the cost-effective depth has a practical limitation. With higher sequencing depth, the number of samples that can be sequenced from one flow cell decreases, which can increase the per-sample cost of sequencing. Therefore, it's crucial to determine the sequencing depth based on the specific project's requirements and budget.

To balance cost and accuracy in bacterial genome sequencing, a recommended sequencing depth of 40x is frequently used. This depth provides adequate coverage for identifying genetic variants and mutations in the bacterial genome while allowing multiple samples to be sequenced on a single flow cell. Depending on the specific research question and the size and complexity of the sequenced bacterial genome, the optimal sequencing depth may vary.

4. Read length:

Read length refers to the length of each sequence read generated by the sequencing platform. A minimum read length of ≥100 bp means that each read is at least 100 base pairs long. Longer read lengths increase the accuracy of the sequencing data and make it easier to assemble the genome.

⇒ **The recommended read length for bacterial NGS is paired ~150 bp**

5. Quality score:

Quality score is a measure of the accuracy of each base call in the sequencing data. Quality scores are expressed as a Phred score, with higher scores indicating a higher probability that the base call is accurate. A minimum quality score of ≥Q30 means that there is a 99.9% probability that each base call is accurate.

⇒ **The recommended average quality score is ≥Q30**

6. Contamination rate:

Non-target DNA sequences in the sequencing data, such as human or environmental DNA, or adapters are referred to as contamination. Cross-sample contamination can occur when DNA from different samples is mixed together during library preparation or sequencing. Adapter contamination is another issue in NGS data analysis. Adapters are short DNA sequences that are added to the ends of DNA fragments during library preparation to allow them to be sequenced on the flow cell. With a maximum adapter content of 1%, the majority of sequencing reads are derived from the DNA fragment of interest rather than adapter sequences. Contamination can reduce the accuracy of the results and lead to the identification of false-positive AMR genes. As a result, it is critical

to check for contamination in the DNA library and in the read data using tools like Kraken2 and remove any contamination before proceeding with downstream analysis.

7. Contig length:

Contig length refers to the length of the contiguous sequence assembled from overlapping sequencing reads. A minimum contig length of ≥500 bp means that the bacterial genome has been assembled into relatively long contiguous sequences, making it easier to annotate and identify genes of interest.

8. Genome completeness:

The proportion of the expected genome that has been assembled is referred to as genome completeness. A genome completeness of 95% indicates that at least 95% of the anticipated genome has been assembled. This indicates that the genome assembly is of high quality and can be used with confidence for subsequent analysis. For bacterial genomes, a commonly used threshold for genome completeness is at least 90% when Benchmarking Universal Single-Copy Orthologs (BUSCO). This threshold ensures that the genome assembly is of sufficient quality to enable accurate annotation and downstream analysis, such as gene prediction, comparative genomics, and functional analysis.

9. Sequence identity threshold:

The minimum level of sequence similarity required to classify two sequences as belonging to the same gene or genomic region is referred to as the sequence identity threshold. A minimum sequence identity threshold of 95% indicates that two sequences must share at least 95% [MOU6] sequence similarity in order to be classified as belonging to the same gene or genomic region.

10. Alignment rate:

The percentage of sequencing reads that can be aligned to a reference genome is referred to as the alignment rate. A minimum alignment rate of 90% indicates that the sequencing data is of high quality.

OVERVIEW ON AMR ANALYSIS WORKFLOW

1. **Sample preparation and DNA extraction**
   - Obtain bacterial samples and prepare them for DNA extraction according to the manufacturer's instructions.
   - Extract DNA using a DNA extraction kit and protocol appropriate for the sample type.
   - Quantify and qualify the DNA using standard QC methods.

2. **Library preparation and NGS sequencing**
   - Prepare the DNA library for sequencing using a commercially available library preparation kit according to the manufacturer's protocol.
   - Sequence the library on an NGS sequencer to generate paired-end reads.

3. **Data quality control and read trimming**
   - Assess the quality of the NGS reads using software such as FastQC.
   - Trim the reads to remove low-quality bases and adapters using software such as fastp or Trimmomatic

4. **Assembly and annotation**
    - Assemble the trimmed reads into contigs using software such as SPAdes or Velvet.
    - Annotate the contigs using software such as Prokka to identify potential antimicrobial resistance genes.
5. **Resistance gene analysis**
    - Identify antimicrobial resistance genes using databases such as CARD or ResFinder.
    - Confirm the presence of resistance genes and their mutations using alignment tools such as minimap2 or BWA.
    - Determine the antimicrobial susceptibility profile of the sample using AMR prediction and genotyping software.

6. **Reporting and interpretation**
    - Report AMR profiles (genotypes) and predicted phenotypes
    - Submit sequences data

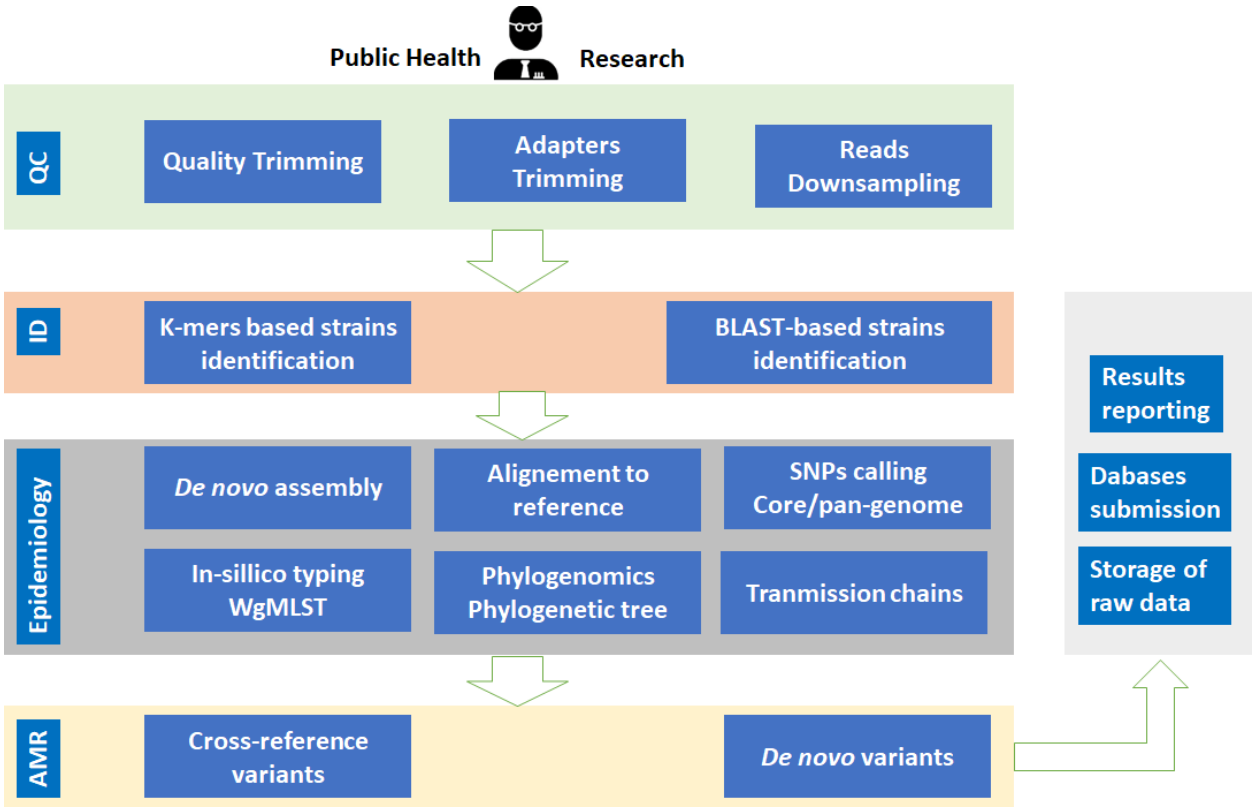

Figure 2: AMR data analysis Workflow

AMR DETECTION PIPELINES

The detection of resistance genes is the most critical step in AMR analysis. One of the key factors to consider is the type of algorithm used by the tool, as different algorithms have varying sensitivities and specificities for detecting different types of AMR. For example, some tools use a k-mer-based approach, while others rely on mapping reads to a reference database. Therefore, multiple detection tools should be combined within

an analysis pipeline to ensure the accuracy and consistency of the results. There are more than a dozen tools or pipelines for detecting AMR in NGS data (Table 1). Selecting the appropriate tools and analysis pipeline for AMR NGS data analysis requires careful consideration of the algorithm used, tool performance, availability of source code, etc. By using a combination of these tools and pipelines, one can achieve more accurate and reliable AMR detection results.

amr_detection_tool	code_availability
TheiaProk	github
StarAMR	github
RGI	github
TBProfiler	github
abritAMR	github
Abricate	github
AMRFinderPlus	github
ResFinder	github
Mykrobe	github
AbriTAMR	github
MegaPath-Nano	github
Graphamr	github
KmerResistance	bitbucket

To meet our training needs, we developed a bioinformatics pipeline using well-known tools like AQUAMIS, TORMES, AMRfinder, and StarAMR. This pipeline was created to detect antimicrobial resistance (AMR) genes in Illumina short reads data with high efficiency. The AQUAMIS tool is used for quality control, TORMES is used for genome assembly, AMRfinder is used for resistance gene detection, and StarAMR is used for annotation and interpretation of the results. These tools were chosen based on their widespread community use and performance in previous studies.

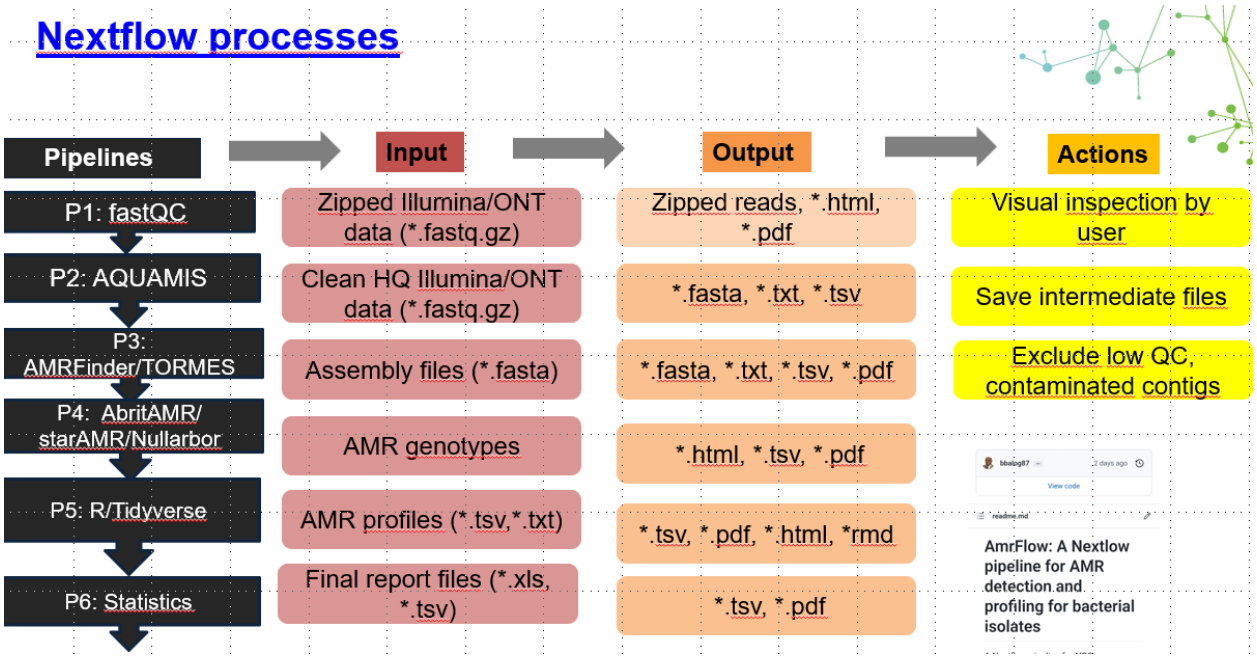

Figure 3: AMR analysis nextflow pipeline

The amrFlow pipeline performs a series of bioinformatics analyses on input fastq files, which are assumed to be paired-end reads from bacterial genome sequencing.

The pipeline currently uses four different tools (FastQC, AQUAMIS, TORMES, AMRFinder, and STARAMR) to perform quality control, genome assembly, annotation, and antibiotic resistance prediction. It then performs a series of bioinformatics analyses on input fastq files, which are assumed to be paired-end reads from bacterial genome sequencing. The pipeline currently uses four different tools (FastQC, AQUAMIS, TORMES, AMRFinder, and STARAMR) to perform quality control, genome assembly, annotation, and antibiotic resistance prediction.

Tools and Softwares

```
FastQC: a tool for quality control of high-throughput sequence data.
AQUAMIS: a tool for de novo genome assembly, quality control, and annotation.
TORMES: a tool for annotating bacterial genomes using homology-based and ab initio methods.
AMRFinder: a tool for predicting antibiotic resistance genes in bacterial genomes.
STARAMR: a tool for detecting antibiotic resistance genes and mutations in bacterial genomes.
```

Figure 4: Nextflow processes

FastQC: The input fastq files are processed using FastQC to check for quality issues, such as sequence length distribution, GC content, and sequence duplication levels. FastQC generates an HTML report and a zip file containing the results.

AQUAMIS: The preprocessed reads from step 1 are assembled using AQUAMIS, which uses a combination of de Bruijn graph-based and overlap-based assembly algorithms to generate a draft genome assembly. The assembly is then evaluated for quality and

completeness using various metrics, such as N50 and BUSCO scores. AQUAMIS also performs gene prediction and functional annotation of the genome.

TORMES: The annotated genome from step 2 is further annotated using TORMES, which uses homology-based and *ab initio* methods to predict genes, functional domains, and noncoding RNAs. TORMES also predicts bacterial virulence factors and prophage regions.

AMRFinder: The genome assembly from step 2 is analyzed for antibiotic resistance genes using AMRFinder. The tool identifies potential resistance genes and mutations in the genome, and generates a report in tabular format.

STARAMR: The genome assembly from step 2 is analyzed for antibiotic resistance genes and mutations using STARAMR. The tool uses a combination of BLAST and MLST-based methods to identify resistance genes and mutations, and generates a report in tabular format.

Pipeline input and output

The workflow expects a directory containing paired-end fastq files, as well as species name and the corresponding MLST scheme. The workflow generates various output files for each tool, including HTML and zip files for FastQC, annotated genome assemblies for AQUAMIS and TORMES, and reports in tabular format for AMRFinder and STARAMR. These outputs are organized into separate directories for each tool and sample, as specified by the publishDir directory of each process.

Step 1: Reads QC

FASTQC:
You can install FastQC using Conda or mamba by creating a new environment and installing it in that environment. Here are the steps to do so:

Ø Create a new environment for FastQC using either conda or mamba:

For conda:

conda create --name fastqc_Env

Ø Activate the new environment: conda activate fastqc_Env

Ø Install FastQC: mamba install -c bioconda fastqc

Ø FastQC should now be installed in your environment.

Note: The -c bioconda option specifies the Bioconda channel from which to install FastQC. If you have already added the Bioconda channel to your list of channels, you can omit this option.

Usage

The usage of the run_fastrqc.sh script is as follows:

$ run_fastqc.sh -d <directory> [-t <threads>] [-o <output_directory>] [-f <fastqc_binary>]

Where:

-d <directory>: specifies the input directory containing paired reads in Fastq format. This is a required parameter.

-t <threads>: specifies the number of threads to use for FastQC analysis. The default is 8 threads.

-o <output_directory>: specifies the output directory for FastQC results. The default is ./FastQC_output.

-f <fastqc_binary>: specifies the path to the FastQC binary. The default is fastqc.

AQUAMIS:

This documentation provides a step-by-step guide on how to install and set up AQUAMIS (Antibiotic Resistance QUAlity control for Metagenomic data using Kraken2, tAxonKit, and MASH), a workflow that performs quality control of metagenomic data prior to performing antibiotic resistance genes identification

Prerequisites

- Ø Linux environment
- Ø Conda or Mamba package manager
- Ø AQUAMIS installed in a Conda environment

Ø Git installed in the Linux environment

Installation

Clone the AQUAMIS repository from the GitLab repository:

> git clone https://gitlab.com/bfr_bioinformatics/AQUAMIS.git && cd AQUAMIS

Create a Conda environment for AQUAMIS and activate aquamis_env :

> mamba create -n aquamis_Env -c conda-forge -c bioconda aquamis -y && conda activate aquamis_Env

Install BUSCO within the AQUAMIS environment:

> mamba install -c bioconda busco –y

Usage

Ø Setup the AQUAMIS databases

Ø Create a sample sheet for AQUAMIS

Ø Run AQUAMIS on the input data.
Before running AQUAMIS, you must set up the necessary databases.

⇒ Run the following command to set up the databases:

bash aquamis_setup.sh -d

Create a sample sheet for AQUAMIS

Create a sample sheet for AQUAMIS using the create_sampleSheet.sh script. Use the following command:

```
> create_sampleSheet.sh --mode ncbi \
--fastxDir /path/to/fastxDir \
--outDir /path/to/outDir \ --force
```

This script will generate a samples.tsv file in the output directory.

The --mode parameter can be set to Illumina, NCBI, flex, or assembly depending on the format of the input files.

Sample naming mode	Patterns
NCBI	SampleID_ReadNumber.fastq Example: Sample1_R2.fastq
FLEX	Any pattern that is neither NCBI nor Illumina. example: Sample1_NovaSeq_S1_L001_i7_index1_R1.fastq
ILLUMINA	SampleID_Lane_Index_ReadNumber.fastq Example: Sample1_001_ATCGTA_L001_R1.fastq

Run AQUAMIS on the input data

To run AQUAMIS on the input data, use the following command:

```
aquamis -l /path/to/samples.tsv \
--taxlevel_qc S \
-d /path/to/output_dir \
--min_trimmed_length 22 \
--threads_sample $THREADS \
--logdir . \
--kraken2db /path/to/kraken2 \
--taxonkit_db
path/to/taxonkit \ -m
/path/to/mash \
-run_name
```

Replace /path/to/samples.tsv, /path/to/output_dir, /path/to/kraken2, /path/to/taxonkit, /path/to/mash accordingly to the path on your system, and run_name with the appropriate values.

--taxlevel_qc: Set to S to perform species-level QC
--threads_sample: Number of threads to use for each sample.
Replace $THREADS with the desired number. -r: A name for the AQUAMIS analysis.

TORMES:

Ø Download configuration

```
wget https://anaconda.org/nmquijada/tormes-1.3.0/2021.06.08.113021/download/tormes-1.3.0.yml
```

Ø Create tormes_env, and install dependencies in tormes-1.3.0.yml

```
mamba env create -n tormes_Env --file tormes-1.3.0.yml
```

Ø Then, activate the tormes environment.

```
conda activate tormes_Env
```

Ø Additionally, the first time you are using TORMES, run (after activating TORMES environment) :

```
> tormes-setup
```

This step downloads and installs additional dependencies required for TORMES and Kraken2. It also downloads the MiniKraken2 v1 8GB database required for Kraken2 to function. If the user has enough disk space and RAM power, it is recommended to download and install the "Standard Kraken2 Database" provided by Kraken2 developers for improved taxonomic identification

Run TORMES

> tormes -m $META -o $OUTDIR -t $THREADS

TORMES will use the metadata file to match reads and genome assemblies and perform a taxonomic classification. The output will be written to the specified output directory.

AMRFinder

Install amrfinderplus using mamba in a dedicated environment:

mamba create -y -c bioconda -c conda-forge -n amrfinder_Env ncbi-amrfinderplus

After installation, activate the amrfinder conda environment using the following command:

conda activate amrfinder_Env

Download the latest AMRFinder database using the command: amrfinder -u

Input Data

The input data should be placed in a directory named "amrfinderINPUT". The following command copies the input files from the TORMES_Out directory to this directory:

```
TORMES_ANNO=/home/nguinkal/AMR-Workflows/TORMES_Out/annotation
mkdir -p amrfinderINPUT && find $TORMES_ANNO/ \
-type f
−name"∗.faa"−o−name"∗.gff"−o−name"∗.fna"−name"∗.faa"−o−name"∗.gff"−o−name"∗.fn
a" -exec cp {} amrfinderINPUT/ ;
```

The script takes as input a set of genome sequences in FASTA format, but also protein sequences along with gff annotations located in the amrfinderINPUT directory. The input

directory is created automatically if it does not exist. The input directory is populated by copying all files with extensions .faa, .gff, and .fna from the TORMES_Out/annotation directory.

The following arguments are used in the script :

- `-n`: input file in FASTA format
- `--organism`: organism to use for searching the AMRfinderplus database
- `--threads`: number of threads to use (default: 8)
- `--output`: output directory (default: "AMRFinder_Out")
- `--name`: name of output file (default: same as input file)
- `--mutation_all`: output file for mutations found (default: AMRFinder_Out/${f%.fna}.mutations.txt")
- `--report_common`: report ARGs that are found commonly in the selected organism
- `--plus`: report additional information on ARGs, such as mutations and variants.

The organism and threads used for AMRFinder analysis are set using the following commands:

```
amrfinder -n $file \
        --organism Staphylococcus_aureus \
        --threads 8 \
        --output AMRFinder_Out/${f%.fna}.amrfinder.out  \
        --report_common --plus \
        --name ${f%.fna} \
        --mutation_all AMRFinder_Out/${f%.fna}.mutations.txt
```

The tools described above have been chained into a meta pipeline amrFlow.

The pipeline checks if the required software tools are installed and clones the AMR NGS Pipeline repository from Github if it does not already exist. The pipeline then sets up the working environment by creating the installation directory if it does not exist, downloading the Miniconda3 installer, checking if Miniconda3 is already installed, and installing it if it is not. It also creates a new Conda environment and instals the required packages for each process. Finally, the pipeline sets up the config file based on the parameters provided by users, and creates the species-specific script (job), which is run using the Nextflow with the specified reads path and organism name.

To use the pipeline, run the following command:

```
> bash amrFlow_setup.sh -h
```

The following command line parameters are displayed:

- -r or --reads: Path to the directory containing the sequencing reads (REQUIRED)
- -o or --organism: Name of the bacterial species (REQUIRED)
- -s or --mlst: MLST scheme for your species (REQUIRED)
- -h or --help: Display help message and exit

This will launch the setup script on your system.

```
$ bash amrFlow_setup.sh --reads AcinetobacterReads --organism Acinetobacter_baumannii --mlst abaumannii_2

======= 2023-04-18 10:55:52.SETTING UP YOUR WORKING ENVIRONMENT...=======

Directory already exists: /home/nguinkal/amrFlow
Miniconda3 is already installed in /home/nguinkal/amrFlow/Miniconda3.
Collecting package metadata (current_repodata.json): done
Solving environment: done ...
```

Figure 5: amrFlow pipeline

ABBREVIATIONS

Abbreviations	
AMR	Antimicrobial Resistance
AST	Antimicrobial Susceptibility testing
MLST	MultiLocus Sequencing Typing
QC	Quality Control
ARGs	Antibiotic Resistance Genes
NCBI	National Center for Biotechnology Information
CARD	The Comprehensive Antibiotic Resistance Database