METAGALAXY

An Oxford Nanopore® sequencing-based metagenomic analysis pipeline

M.D.C. JANSEN

JANUARY 30, 2020

AVANS UNIVERSITY OF APPLIED SCIENCES
Lovensdijkstraat 61-63, 4818 AJ, Breda, The Netherlands

Report information

Author

M.D.C. Jansen

Student Life Sciences

mdc.jansen@student.avans.nl

Education

Biologie en Medisch laboratoriumonderzoek

Avans University of Applied Sciences

Academy of Technology of Health and Environment

Lovensdijkstraat 61-63

4818 AJ, Breda, The Netherlands

Project Supervisor

Dr.ir. M.M. Zhou (m.zhou1@avans.nl)

Lecturer and Bioinformatician

Internship Supervisor

Dr. M.C.M. Verschuren (mcm.verschuren@avans.nl)

Lecturer and researcher in Life Sciences

User case data

Dr. E.J.A Schrauwen (Eja.schrauwen@avans.nl)

Lecturer-researcher

B.Sc. M.J.R. Ivens

Bachelor in forensics

Table of Contents

Introduction

Next generation sequencing (NGS) has become an increasingly important tool for clinical microbiology. Fast and accurate identification of bacterial strains is crucial for successful treatment and limiting outbreaks of antibiotic resistant strains[1]. When a clinical sample has been sequenced, the sequenced can be used for taxonomic identification of organisms present in the sample, clinical diagnostics, and the detection of antimicrobial resistance- (AMR) and virulence genes[1-3]. More importantly, sequencing of these metagenomes has made it possible to perform these analyses on multiple bacteria in a sample. Additionally, the increased use of NGS in clinical microbiology can significantly decrease diagnostic time of patients[4].

One form of analysing metagenomic data from clinical samples is done by analysing metagenomic data from long-read sequences. Long-read metagenomic analysis is usually performed with the assistance of various pipelines for specific purposes like ANASTASIA for novel enzyme discovery[5]; metaWRAP for genome-resolved metagenomic analysis[6]; or WIMP for real-time species identification[7]. Recently, the first hybrid metagenomic was published: OPERA-MS[8]. This pipeline allows for more accurate assemblies with more depth than the non-hybrid pipelines. The afore mentioned pipelines perform their analysis on shotgun-metagenomic data. When analysing targeted-amplicon metagenomic data, it is possible to perform analysis on a single gene, such as 16 rRNA[9].

Most metagenomic analyses are performed on Linux-based systems. The majority programmes that are used to perform the analysis on Linux are open-source and under copyleft laws. This means that any researcher is allowed to freely use and modify the programmes to their specific needs, as long as they cite the original tool and author. Aside from a large selection of programmes that can be used, all supercomputers run on modified versions of Linux. This means that the programmes that are developed on the Linux platform can be used on supercomputers, without having to heavily modify the programmes to make them compatible for use on a supercomputer.

Analysing the metagenomic data can be done by a plethora of tools and pipelines. There is no golden pipeline to perform all the different analysis, especially for long-read sequence data, like the data from the MinION®. This can create difficulties for wet-lab researchers who are not well versed in bioinformatics. Additionally, most of these tools and pipelines do not have a graphical interface, but are limited to a command line view only. To run command line based tools; knowledge of Linux systems and bash commands is required and it can take up to 6 months for researchers to acquire this knowledge. Integrating these pipelines into an user-friendly platforms which provide a graphical interface for pipelines is one of the practical solutions to the earlier mentioned bottleneck and does not significantly prolong the runtime of various pipelines. Galaxy is a web-based platform and has public servers that can be utilised by these pipelines. Private instances can be deployed and used for confidential data or the instance can be run on a large computer cluster.

Galaxy is designed to perform bioinformatic data analysis and currently runs on python 2.7. It was created to assist non-bioinformaticians in analysing their data[10]. The platform has a plethora of tools, allowing its users to perform different kinds of analysis such as determining exons with highest SNP density [11] or CHiP-seq analysis[12]. It even allows for deployment of custom-made pipelines and workflows. These workflows also allow for reproducible analysis, since they can be shared on the platform and can also be published[13]. Galaxy also supports the use of multiple histories, which enables researchers to separate multiple analyses and run multiple workflows independently and simultaneously[14]. The users do not have to program or implement any of the tools they want to use.

Instead, they are given an interactive, web-based interface on which they can upload their own data or import datasets and perform their analysis[15].

The aim of this study was to develop a metagenomic analysis pipeline that uses long-read FASTQ data that is capable of identifying the organisms present in the sample, recover their genomes, and perform anti-microbial resistance (AMR) typing on the recovered genomes. This way AMR genes can be assigned to individual organisms instead of the entire dataset. The pipeline was to be implemented into Galaxy, so the analysis could be performed without prior knowledge of the Linux command line.

Abstract

Long read sequencing of metagenomic samples is becoming an increasingly popular method in clinical microbiology. Since it allows for fast and accurate identification of bacterial infections and the potential antibiotic resistances they may carry. However, there are a limited amount of options when it comes to analysing the data with a single pipeline. Additionally, the solutions that are available usually do not have an user friendly alternative for non-bioinformaticians that would still like to analyse their own data. In this study, MetaGalaxy is presented. A metagenomic analysis pipeline with an user friendly alternative. The aim of this study was to produce a metagenomic pipeline that is able recover genomes from a metagenomic sample, isolate the recovered genomes, taxonomically identify the genomes, and discover anti-microbial resistance(AMR) genes present in the recovered genomes. The produced pipeline, MetaGalaxy, was created to run analyses on Linux-based systems and is written in Python3. This pipeline was also reproduced on Galaxy, which provides an accessible and more user friendly alternative that can be used to obtain similar results to MetaGalaxy. The analysis starts by assessing the quality of the input FASTQ file and taxonomic classification. The reads from the input file are also filtered by quality. Any read with a read length below 1kpb or an average phred score below 7 are filtered out. After filtering, plasmid reads are identified, isolated, and screened for AMR genes. The other reads are used for assembly and five rounds of polishing. After polishing, quality assessment is performed on the contigs and the original assembly graph is visualised. The polished consensus will then be binned to separate the recovered genomes. These bins are then taxonomically classified, to determine which organism is present in which bin. Followed by AMR screening to discover the AMR genes that are present in each binned organism. The analysis that is performed on Galaxy, contains less polishing rounds, to significantly lower the time it takes to perform the analysis. Additionally, the polished consensus is not binned. Instead, contigs are taxonomically annotated and AMR screening is also performed on the polished consensus. The pipeline that was produced during this study is available on GitHub(https://github.com/mdcjansen/MetaGalaxy) along with the results that are presented in this study(https://github.com/mdcjansen/MetaGalaxy/user_case_results). MetaGalaxy completed the analysis on a 10GB file in four hours and fifteen minutes. Taxonomic classification of the input reads yielded results that are comparable to WIMP. Two genomes were recovered during assembly and both were successfully binned, annotated, and screened for AMR genes. Plasmid sequences were also isolated and screened for AMR genes. Analysis on Galaxy yielded similar results. After the assembly the consensus was only polished once, which resulted in a lower quality consensus compared to MetaGalaxy. Gene annotation was performed on the polished consensus, followed by MLST to identify organisms present in the sample. Screening of AMR genes was also done on the polished consensus. MetaGalaxy is able to consistently produce results on metagenomic data and the analysis can be performed on either Linux-based systems and a command-line interface or on Galaxy.

Method

System requirements

The scripts used to run MetaGalaxy are available on Github at: https://github.com/mdcjansen/MetaGalaxy. The scripts are written in Python and Shell and are designed for usage on Linux-based systems. MetaGalaxy can use up to 50GB of RAM during the analysis. By default, MetaGalaxy will use all the threads on the system is runs on, up to a maximum of 256 threads. When MetaGalaxy is downloaded from GitHub, it is less than 1MB in size. When the tool installation script is run, the necessary tools are downloaded, build, and installed. This will increase the size of MetaGalaxy to approximately 160MB. Once the database installation file is run, which will download and build the required databases, MetaGalaxy will use around 530GB of disk space. The pipeline was developed on a HP® Z4 workstation with the following specifications:

- CPU: Intel® Xeon® W-2175 @ 2.50 GHz
- Graphics card: NVIDIA® Quadro® p2000
- RAM: 128 GB 2666 MHz Samsung® M393A4K40BB2-CTD
- HDD used during analysis: 4 TB 7200 RPM SATA Enterprise 3.5” HDD

MetaGalaxy design

MetaGalaxy is designed to recover and identify bacterial genomes from metagenomic samples and identifies AMR genes for the discovered bacteria. It uses long-read FASTQ files as input. The pipeline is also capable of demultiplexing nanopore FASTQ files.

The analysis includes 15 different steps, which have been divided into five sections(Figure 1). During the first set of analyses, MetaGalaxy will determine the quality of the raw reads, filter out low quality and short reads, perform a taxonomic classification on the raw reads and visualise the taxonomic classification.

The second set of analyses includes determining the quality of the filtered reads, isolating the plasmid reads from the filtered FASTQ file and identify anti-microbial resistance (AMR) genes on the plasmid reads.

The third set consists solely of the metagenomic assembly and polishing. During the fourth set of analyses, MetaGalaxy will visualise the assembly, determine the quality of the polished assembly and it will bin the contigs.

Once the contigs are binned, the final set of analyses will be performed. During this last set, MetaGalaxy will assign taxonomies to the bins; visualise the taxonomy, which will also show the rough abundance of each species in the sample; and identify AMR genes in each of the bins.

Figure 1: Schematic overview of MetaGalaxy. The blue box represents the input long-read FASTQ file. The green boxes represent the tools that are used, along with a small description of its function. The red boxes represent the output files created and saved during analysis. MetaGalaxy is divided into five different stages: 1. Quality control on the raw reads, filtering short and low quality reads with, and taxonomic classification which are then visualised. 2. Extracting the plasmid reads from the filtered FASTQ file, antimicrobial resistance (AMR) detection, and quality control of the filtered reads. 3. Assembly and polishing. 4. Quality assessment of the polished assembly, Visualising the unpolished assembly, and contig binning. 5. Taxonomic classification of the bins, visualising abundance of organisms across all bins, and AMR detection per discovered bin.

Demultiplexing

MinION sequencing runs can be done with up to twelve samples at a time. Before sequencing, these samples are multiplexed, which means they are labelled with a barcoding sequence that is unique for each sample. These barcoded sequences are then put into one master mix, which is then sequenced. After sequencing, the produced FASTQ reads have to be assigned to the right sample. This is done by demultiplexing. During demultiplexing, the reads are scanned for the barcodes. Once a barcode has been found on a read, it is assigned to the sample where the sequence originally came from and the barcode can be trimmed off the sequence, so it cannot interfere with the analysis.

The demultiplexing process contains three steps: Demultiplexing, Merging demultiplexed files and performing statistical analysis (Figure 2).

Demultiplexing is done by Guppy v3.4.5. (ONT) and is performed separately from the main programme. To signal MetaGalaxy to perform demultiplexing, the function "--demultiplex" is used. Demultiplexing requires two inputs, the FASTQ file that needs to be demultiplexed and the barcoding kit ID. The ID's that can be used in the pipeline can be shown when the function "--bcavail" is used.

During demultiplexing, Guppy will also trim the barcode sequences, so that they can't interfere with analyses later on. After demultiplexing, a python script will merge the individual FASTQ files of each barcode and the unclassified into a single file, which will result in a total of twelve files. One for each of the twelve barcodes.

Once the files have been merged, NanoComp[16] will compare the twelve FASTQ files and create two violin plots, three histograms and a NanoStat[16] text file. These plots and histograms can also be viewed in an html file, which will also be produced by NanoComp[EXAMPLE]. The first violin plot displays the read lengths and the second violin plot displays the average base call quality scores. The histograms show the read length N50, the number of reads and the throughput in gigabases. The text file that is created will show a general summary which displays the mean read length, mean read quality, median read length, median read quality, number of reads, read length N50, and total amount of bases per barcode. The top 5 highest mean base quality scores and the top 5 longest reads of each barcode is also shown.

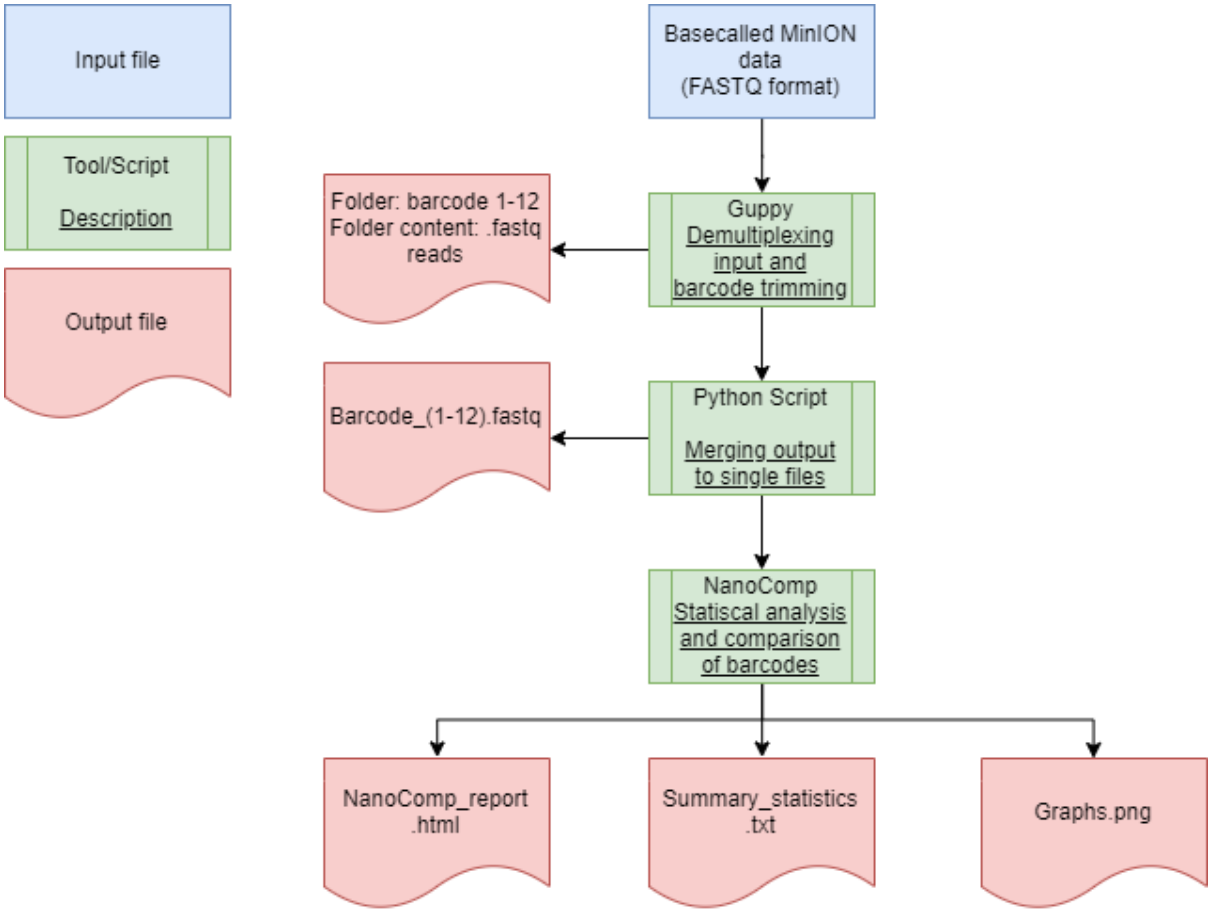

Figure 2: Schematic overview of the MetaGalaxy demultiplexing process. The blue box represents the input file in FASTQ format, the green boxes represent the Tools or Scripts that are used to perform an analysis, and the red boxes are the output files that can be viewed after the analysis is complete. When demultiplexing with MetaGalaxy, Guppy will demultiplex the data and trim the barcodes. After demultiplexing, a python script will merge all the FASTQ reads from each individual barcode into singular FASTQ files. These twelve merged files are then put through quality control, which is performed by NanoComp. NanoComp will compare the read length, base quality, read length N50, the number of reads and the throughput in gigabases for all the twelve barcodes and the unclassified reads.

Quality control and filtering

Filtering is done by Filtlong v0.2.0 (https://github.com/rrwick/Filtlong). All reads smaller than 1kbp and with a phred score below 7 are filtered out. The quality control is performed by NanoPlot before and after filtering, so the change in quality can be observed by the user. NanoPlot produces an html file, which contains three histograms that display the number of reads against the read length, log transformed number of reads against the read length, number of bases against the read length, and log transformed number of bases against the read length. The program also generates a text file which will display a general summary, which includes a mean read length, mean read quality, median read length, median read quality, number of reads, read length N50, and total amount of bases. The top 5 highest mean base quality scores and the top 5 longest reads are also displayed.

Taxonomic classification of raw reads

Taxonomic classification is performed by Kraken2[17]. Kraken2 classifies organisms by matching each k-mer within a query sequence to the lowest common ancestor found in the database. Two databases are used to run kraken2 twice. The first database is constructed using the standard build option in kraken2, whilst the second one is custom build with the kraken2-build script. The second database, is a protein database that contains the complete RefSeq protein database of archaea, bacteria, plasmids, viruses, fungi, and protozoa. It also contains the GRCh38 human protein database, a non-redundant protein database with sequences from large environmental sequencing projects.

Kraken2 produces a report file and a values file. The report file is a tab-delimited file that consisting of six columns. The first column displays the percentage of fragments that are covered at the specific taxon. The next column displays the number of fragments covered by the clade and the third column shows the fragments that are directly assigned to the taxon. The fourth column specifies the taxonomic rank code which indicate one of the following taxons: (U)nclassified, (R)oot , (D)omain, (K)ingdom, (P)hylum, (C)lass, (O)rder, (F)amily, (G)enus or (S)pecies). The fifth column displays the NCBI taxon ID and the last column displays the scientific name of the organism that has been found. The report is split into two files. One file contains all the genera found with an abundance of 2.00 or higher. The second file contains all the species found with an abundance of 2.00 or higher. Taxons that have an abundance lower than 2.00 are discarded, to reduce redundancy and low abundance organisms may be misidentified.

The values file is tab-delimited text file that contains five columns and displays information on classification of each sequence. The one letter code which are either “C” or “U” in the first column shows if the sequence is classified or unclassified. The next column shows the sequence ID that has been obtained from the FASTQ header. The third column displays the taxonomy ID that has been found. When the sequence is unclassified, this value will be zero. The following column displays the sequence length in bp and the last column contains a space-delimited list that shows, in sequential order, the amount of k-mers mapped to a specific taxon ID separated by a colon in this format: taxonomy ID:k-mer count. When the taxonomy ID has the value “A”, it means that the k-mers contained an ambiguous nucleotide. When the taxonomy ID has a value of zero, it means that those k-mers were not found in the database.

These results can also be visualised into an interactive pie chart with the tool Krona[18]. Krona uses the second and third column of the kraken2 values file and produces a html file that can be viewed in any modern browser like Google® Chrome®, Mozilla Firefox and Microsoft® Edge®.

The percentages that Krona displays represents the same value that is presented in the first column of the kraken2 report file. Krona allows users to zoom in on every taxon, which allows for less abundant organisms to be displayed.

Isolating Plasmid reads and anti-microbial resistance typing

Plasmids sequences can be easily lost during metagenomic assemblies, due to their low abundance. These plasmids can still contain AMR genes and therefor it is important to isolate them before the assembly, so they can still be screened for AMR genes. Once the input FASTQ reads have gone through quality filtering, the plasmid reads are isolated.

After isolation, the reads are screened for AMR genes. Identification of plasmids is done by ABRicate v0.9.8 (https://github.com/tseemann/abricate). ABRicate uses a BLAST-based search with the PlasmindFinder[19] database to analyse the data. If ABRicate can cover 60% of a read with an identity of at least 60%, then the read will be considered to be a plasmid read. A python script will then remove these reads from the filtered FASTQ file and stores them in a new, separate plasmid FASTQ file. This file is run through ABRicate again, but with the ResFinder[19] database to identify AMR genes and uses the same coverage and identity parameters as mentioned before to identify the AMR genes. The genes that are found are put in a tab-delimited file that is made up of fifteen columns(Table 1).

Table 1: ABRicate output. The first column displays the name of each column in the ABRicate output. The second column briefly explains the meaning of the information represented in that particular column.

Column name	**Description**
FILE	The file name where the plasmid read was found
SEQUENCE	The sequence name
START	Start coordinate of the gene in the sequence
END	End coordinate of the gene in the sequence
STRAND	Indicates the forward strand (+) or reverse strand (-)
GENE	AMR gene found by ABRicate
COVERAGE	Which part of the gene has been covered
COVERAGE MAP	Visual representation of the coverage. Aligned(=), unaligned(.), gaps(/)
GAPS	Amount of gaps in subject and query
%COVERAGE	Percentage of the gene that is covered
%IDENTITY	Percentage of exact nucleotide matches
DATABASE	Database where the sequence is acquired from
ACCENSION	Source of the sequence
PRODUCT	Gene product (if available)
RESISTANCE	Antibiotic resistance phenotype, semi-colon separated

The results are also summarised into a second file. This file contains three or more columns. The first columns the first column contains the names of the files that are summarised. The next column displays the amount of genes that have been found in the file. The third and following columns display all the genes that have been found. The header contains the name of the gene that has been found and each row displays the coverage in percentage. When a particular gene is not found on a read, then the value in that cell will be a dot “.”(Table 2).

Table 2: ABRicate output summarised. The first column displays the name of each column in the summarised ABRicate output. The second column briefly explains the meaning of the information represented in that particular column.

Column name	Description
FILE	The file name where the AMR gene was found
NUM_FOUND	The number of genes found in that particular file
"Gene 1"	Coverage in percentage of the gene that is covered. A dot "." is placed when the gene is absent
"Gene 2"	Coverage in percentage of the gene that is covered. A dot "." is placed when the gene is absent

Metagenomic assembly, polishing and quality assessment

Once the plasmid reads have been extracted, MetaGalaxy will start the metagenomic assembly. This assembly is done by metaFlye[20]. metaFlye uses the Flye assembler with the metaFlye module enabled. The assembler has an in-build polishing option, however, this is disabled in this pipeline, because a different polisher is used to increase the consensus accuracy. Before assembly can start, an estimated genome size must be specified. This parameter is used by metaFlye for solid k-mer selection. This estimation does not have to be precise and it does not affect any other stages of the assembly. The genome size given, can between 0.5-2.0x as large as the actual total genome size in the sample.

After assembly, the consensus is polished by Racon[21], which uses the assembly consensus, input FASTQ file and a PAF file generated by Minimap2[22]. Minimap2 uses the Nanopore vs reference mapping mode to map the consensus against the input FASTQ file and creates a PAF file which contains the approximate mapping positions between the consensus and the input sequences. Racon uses this file along with the other two previously mentioned files to polish the consensus. During error correction, scores are assigned to each matching base, mismatched base, and gap. Racon builds the polished consensus based on these scores and attempts to produce a consensus with the highest possible score, whilst maintaining accuracy. Matching bases are given a score of 8, mismatched bases are given a score of -6 and gaps are given a score of -8. The assembly is polished per window of 500 bp. Once the consensus has been polished with Racon for four times, one final round of polishing will be performed by Medaka v0.10.0 (https://github.com/nanoporetech/medaka) to produce the definitive polished consensus in FASTA format.

To assess the quality of the polished consensus, metaQUAST[23] is used. Quast performs statistical analysis and a BLAST search to produce various results which can be viewed in a .html report(Appendix I: metaQUAST report

Appendix I: metaQUAST report).The statistical data is divided into six categories: Genome statistics, Misassemblies, Unaligned, Mismatches, Statistics without reference, and Similarity statistics. The first category, genome statistics, displays information on the genome fraction in percentage, the duplication ratio, largest alignment, total aligned length, NA50 and 75, LA50, and 75, NGA50 and 75, LG50 and 75, and LGA50 and 75. The second category, misassemblies, first displays the total amount of missasemblies found, which are then divided into four types of misassemblies: relocations, translocations, inversions, and interspecies translocations. This category also displays information on the amount of misassembled contigs and the length of those contigs as well as scaffold gaps. To produce the unaligned results, metaQUAST performs a BLAST on the polished consensus against a NCBI reference. Which references are downloaded and used for this BLAST are determined beforehand by a preliminary BLAST against a small reference database that comes with metaQUAST.

Once the BLAST has been completed, the unaligned results are summarised in the unaligned category. This category shows the amount of fully and partially unaligned contigs as well as the total unaligned length that is fully or partially unaligned against the reference. The fourth category displays the amount and types of mismatches. First, the total amount of mismatches in all the aligned bases are shown. Then, the number of indels and the total length of all the indels are displayed. Next, the average amount of mismatches per 100 kpb is shown and after that the average amount of indels per 100 kpb are shown. The number of indels are also divided into two subgroups: indels smaller than 5 bp and indels larger than 5 bp. Lastly, the total number of uncalled bases and the number of uncalled bases per 100 kpb are displayed. The statistics without reference category displays information on the number of contigs, contig length, N50 and 75, L50 and 75, and the GC content in percentage for contigs aligned to a specific reference or all the unaligned contigs.

Besides displaying the total number of contigs found, metaQUAST also reports how many of those contigs are equal to or larger than 0bp, 1000bp, 5000bp, 10000bp, 25000bp, and 50000bp. The last category displays the number of contigs that is over 50% similar across all the assemblies and the number of contigs that have misassembled blocks that are over 50% similar across all the analysed assemblies. Since metaQUAST is only used on the polished consensus, there are no other assemblies to compare to. Therefor these values should be zero. metaQUAST also reports the organisms it found during the BLAST search. The exact species that was found can be inaccurate, since metaQUAST is not designed for accurate genome annotation. However it can give some insight on which genus of organisms are present in the sample. The report is a list that contains the name of the species found, the genome size of that species, how many fragments are associated with that organism and the GC content in percentage of the detected organism. Lastly, eleven plots are made that plots one of the statistics against the detected organisms and unaligned contigs. The statistics that are plotted are: number of contigs, largest alignment size, total alignment size, number of misassemblies, misassembled length, number of mismatches, number of indels, number of uncalled bases per 100 kbp, genome fraction in percentage, duplication ratio, and NGA50.

Aside from the statistical analysis, the contigs are also visualised for inspection and clarity. Visualisation is done by Bandage[24] and the produced image is saved as a .png file. Contigs are displayed as coloured bars and connections between contigs are displayed as lines. The visualisation gives an indication on the wholeness of the recovered genomes as well as the amount of plasmids or loose contigs that have been assembled.

Binning, taxonomic classification and anti-microbial resistance typing

Before the contigs can be binned and the taxonomic analysis can be performed, a SAM file has to be created. A SAM file has to be produced in two steps. First, the polished consensus is mapped against the input FASTQ file by Minimap2 to produce a BAM file. Then, the SAM file is created from the BAM file by samtools v1.9 (https://github.com/samtools/samtools) by using the "sort" option.

Binning is performed by metaBAT2[25] and starts by summarizing the contig depths from the BAM file, followed by the actual binning. During the summary, metaBAT2 will only use mapped reads with an 80% identity to the contig to calculate the depth. The summarised depth is stored in a text file, which is then used alongside the polished consensus to bin the contigs. Only contigs with a length of 1500bp or more are used during binning. From these contigs, 75% of the good contigs are used based on connection between the contigs. Edges must have a minimum score of 40 and maximum number of 500 edges are allowed. Lastly, each contig in a library must have a mean coverage of 10 and each generated bin must have a size of at least 50kpb. If any of the parameters are not met, the unbinned contigs are placed in one of three FASTA files: lowDepth, tooShort, or unbinned. Contigs are placed in the first bin, lowDepth, when their initial depth or mean coverage is too low. The second file, tooShort, contain contigs with a size smaller than 1500bp or a bin with a size smaller than 50kbp. If a contig is not binned for any other reason, they are placed in the unbinned file. The created bins are saved individually as a FASTA files and used for the last two analyses.

Firstly, the discovered bins are taxonomically identified with BAT from the CAT/BAT[26] tool. BAT uses DIAMOND[27] to BLAST the bins against a database that is created by CAT/BAT. After the DIAMOND search, Prodigal[28] is used to annotate the findings by DIAMOND. The final output is a tab-delimited file consisting of twelve columns(Table 3). This file displays information on the discovered lineage, lineage scores and the amount of ORFs that were used to taxonomically identify the bin. The fraction bit-score cutoff is set to 0.04, so that the tool is able to identify the exact species from each bin.

The output from DIAMOND is also used by Krona for visualisation. The Krona pie chart that is produced, clearly displays the abundance levels the organisms across all the bins. It is not possible to view the abundance per bin, since DIAMOND does not separate the results from each bin. Instead, the separation of taxonomies is done later on by BAT.

Table 3: CAT/BAT annotated output. The first column displays the name of each column in the summarised ABRicate output. The second column briefly explains the meaning of the information represented in that particular column.

Column name	**Description**
Bin	The bin that has been annotated
Classification	Shows if the bin has been classified, has multiple classifications, or has not been classified
Lineage	The found lineage in NCBI taxon IDs. Semicolon delimited
Lineage scores	Fraction of bit-score for every classification. Semicolon delimited
Superkingdom	Superkingdom assigned to the bin, followed by the lineage score
Phylum	Pyhlum assigned to the bin, followed by the lineage score
Class	Class assigned to the bin, followed by the lineage score
Order	Order assigned to the bin, followed by the lineage score
Family	Family assigned to the bin, followed by the lineage score
Genus	Genus assigned to the bin, followed by the lineage score
Species	Species assigned to the bin, followed by the lineage score

The second analysis is the identification of AMR genes by ABRicate. The same parameters, 60% minimum identity and coverage, are used to detect the AMR genes. Also, the ResFinder database is reused and the report file(Table 1) as well as the summary report file(Table 2) have the same format.

Galaxy workflow

A workflow of MetaGalaxy has been produced as well and is available on the Avans Galaxy instance (https://galaxy.bioinformatics-atgm.nl:2222) as well as on Github(https://github.com/mdcjansen/MetaGalaxy/galaxy_workflows). This version of Galaxy is only accessible through federated eduroam on Avans. The analysis is performed the exact same way up until the first round of polishing. The main difference before the polishing between the galaxy analysis and MetaGalaxy are two modified wrapper scripts(https://github.com/mdcjansen/MetaGalaxy/galaxy_wrapper). The first script is a modified version of the Krona wrapper. This wrapper refers to Krona used by MetaGalaxy, since the version of Krona on the local Galaxy is bugged and does not consistently produce results. The second and last difference before polishing is that the wrapper script for Flye has been modified. Flye on Galaxy is outdated and therefor does not have the metagenomic assembly option. This particular wrapper refers to the Flye assembler used in MetaGalaxy, since this one is up to date and is capable of utilising the metagenomic assembly option. After the assembly, only a single round of polishing is performed with Racon, since Galaxy is incapable of running Racon multiple times on a polished consensus. Galaxy currently does not have an efficient and reliable binner and taxonomic classifier. Instead of binning and taxonomic classification, Prokka[29] is used to annotate the genes, which can then be viewed in IGV[30] or JBrowse[31]. With these results is it possible to determine which organisms are present in the sample. Additionally, multi-locus sequence typing (MLST) is run on the fasta files produced by Prokka. During MLST a sequence identity of 60% against the database is considered to be similar, a minimum of 10% of alleles have to be covered with a minimum score of 10 to match a scheme. The parameters are set to be quite lenient, since nanopore data isn't too accurate and the purpose of this analysis to detect up to four different organisms and not to detect the exact strain of an organism. MLST only reports on a single organism per input file and Prokka creates four FASTA files. This means that MLST can report up to four different organisms, if they are present in the same abundance. Lastly, ABRicate is still used to identify the AMR genes. Instead of reporting the results on a bin, it will report on the contigs. This will result in a larger output file with more AMR genes detected, since MetaGalaxy does not perform AMR analysis on the unbinned contigs.

Results

MetaGalaxy Pipeline

MetaGalaxy has been released on Github(https://github.com/mdcjansen/MetaGalaxy) and can be cloned and used on computers which run on Linux. Once the installation file in the "lib" and "data" directory have been run by the user, MetaGalaxy is operational. As mentioned in Chapter 0 System requirements, MetaGalaxy requires a total of 530GB of free disk space to install all necessary tools and databases.

These 24 tools are used by MetaGalaxy during analysis: ABRicate, Any2fasta, Bandage, BLAST, CAT/BAT, DIAMOND, Filtlong, Flye, Guppy, Hmmer, htslib, Kraken2, Krona, Medaka, metaBAT2, Minimap2, NanoComp, NanoPlot, NanoStat, pplacer, Prodigal, Quast, Racon, Samtools(Appendix II: List of tools). They can all be downloaded from either Github or Bitbucket. Some of these tools are used as dependencies for others, whilst the rest of them are used by MetaGalaxy. Various dependencies are also needed to run the pipeline, including: Java, Anaconda3, GCC 4.9 or later, boost 1.5.3 and Core OS development headers (Appendix II: List of tools).

Once MetaGalaxy has been initiated, it will search for the given input file, output directory, and it will create a log. The programme checks whether or not the specified input file exists. If it does not exist, the analysis will end and the user will be notified that the input file cannot be found. If the specified output directory does not exist, MetaGalaxy will create the output directory. When the output directory is already present, MetaGalaxy will ask the user if the data within the directory is allowed to be cleared before the analyses start. Analysing the input data and outputting it to a used output directory is not recommended, because old file may be overwritten and MetaGalaxy could potentially use old files during the analysis, which would make the final results unreliable. The log file that is created, will output the input parameters that have been given by the user and it keeps track of the progression of the analysis. Once the analysis has been completed, the log file will show how long it took to complete the analysis. If during the analysis a tool creates an error that impairs it from continuing its analysis, the log file will report a critical error on the programme. As of yet, MetaGalaxy is unable to relay the error message from the tool, but the error can be read from the terminal interface.

This pipeline can also be imported into Galaxy, where the workflow will look slightly different than the analysis done by the pipeline. metaBAT2 cannot be used in Galaxy yet and therefore the analysis after the polishing has changed. The only other binner the Galaxy offers is MaxBin2, but this tool was unable produce any results on provided datasets. Instead of binning, the analysis in Galaxy relies on gene annotation by Prokka and MLST. Additionally, there is only one polishing round on Galaxy, to significantly decrease the amount of time needed to perform the analysis. The workflow is available on GitHub and can be imported into any Galaxy instance. There is an additional version available that does have the four polishing rounds, though it is significantly slower and might not yield more statistically significant results depending on the quality and size of the input files.

User case

To test the reliability and effectiveness, MetaGalaxy was tested against a dataset that was produced and analysed. by M. Ivens in mid-2019. The data was acquired from a Urine sample that originated from a hospital patient with an urinary tract infection (UTI). During Ivens's research, he identified which organisms and which AMR genes were present in the sample. A phenotypic resistance profile was produced by the hospitals' diagnostic lab to confirm Ivens's findings. The analysis done by Ivens included a quality control on the demultiplexed reads, Taxonomic classification, and AMR resistance typing. The dataset has a file size of 10GB and is referred to as BC04, since it was the fourth barcode during the sequence run. In M. Ivens's research, it was concluded that BC04 contained two organisms: *Enterococcus faecalis* and *Escherichia coli*. These bacteria did not show any phenotypic resistance, but AMR genes were found for antibiotic resistances that were not tested in the lab.

MetaGalaxy was able to analyse the dataset in four hours and fifteen minutes. The first results were acquired within a few minutes. The first result to be produced was the taxonomic classification of the reads, followed by filtering, and lastely the statistical analysis of the reads. It took the pipeline three hours to complete assembly and the polishing rounds. During the last hour of the analysis, MetaGalaxy binned the polished contigs, annotated the bins, and screened the bins for AMR genes.

Analysis on Galaxy was significantly slower. The first results were still produced within a few minutes. Assembly and polishing was significantly slower in comparison to MetaGalaxy, which caused the total analysis time to increase. The exact time it took for MetaGalaxy to complete the analysis was not accurately determined, but was estimated to be around five hours.

Quality control

Quality control was performed by NanoPlot during Ivens' research and is also used by MetaGalaxy(Table 4). Mean read length calculated by both analyses is exactly the same and the mean read quality calculated during Ivens's analysis is slightly higher than the quality measured by MetaGalaxy. Ivens found a mean read quality of 9.64 versus the quality of 9.6 found by MetaGalaxy. NanoPlot no longer reports the second decimal on its report, therefor it is impossible to determine whether or not NanoPlot in MetaGalaxy produces the exact same mean quality score as NanoPlot did in Ivens's research.

Table 4: NanoPlot results of M. Ivens versus MetaGalaxy. The first two columns display the data produced by M. Ivens. The third and fourth column displays the data produced by MetaGalaxy.

Nanoplot by M. Ivens		**NanoPlot by MetaGalaxy**	
Mean read length:	3,952	Mean read length:	3,952.10
Mean read quality:	9.64	Mean read quality:	9.6
		Median read length:	3,922.00
		Median read quality:	9.8
		Number of reads:	1,507,401.00
		Read length N50:	4,611.00
		Total bases:	5,957,377,504.00

Futhermore, Ivens did not report on the median read length and quality as well as the number of reads, read length N50 and total amount of bases. So it is not possible to compare those results. MetaGalaxy calculated the median read length and quality to be 3922bp and 9.8 respectively. BC04 contains 1.5 million reads consisting of 5.95 billion bases and a Read Length N50 of 4611bp.

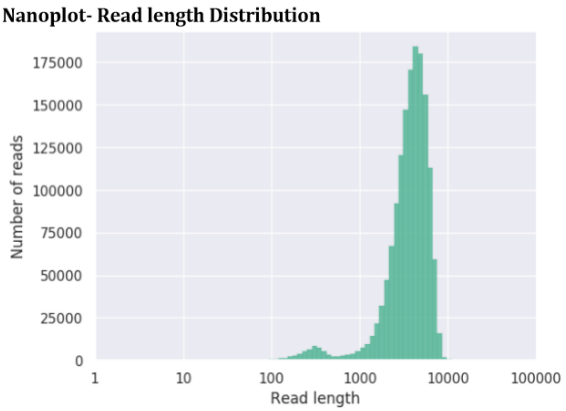

Figure 3: NanoPlot histogram of read length distribution after log transformation. Produced by M. Ivens

Several plots are also produced, two of which are reported by Ivens. The first plot, a histogram, shows the number of reads plotted against the read length(Figure 3). The second plot is a density plot where the average read quality is plotted against the read lengths(Appendix III: NanoPlot results). The read length reported in the density plot by MetaGalaxy is displayed in a linearly, whilst the read length reported in Ivens's density plot is displayed logarithmically. Even though these plots do not look similar, they do display similar results.

The histograms produced by MetaGalaxy(Figure 4) and Ivens(Figure 3) have the same distribution. Where most of the reads are centred around the Median read length of 3922bp as reported by MetaGalaxy. Interestingly, there is also a small peak of reads between the read lengths of 100bp and 1000bp.

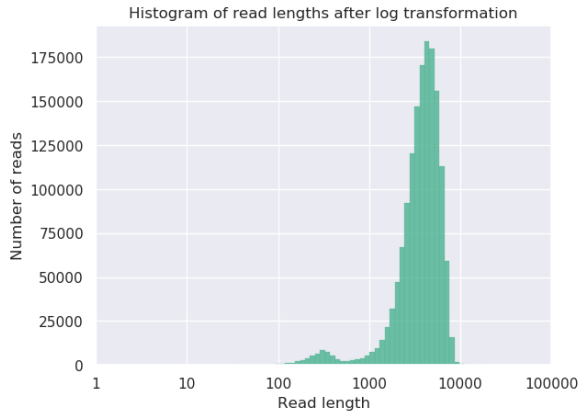

Figure 4: NanoPlot histogram of read length distribution after log transformation. Produced by MetaGalaxy

MetaGalaxy also produced five other plots(Appendix III: NanoPlot results).A histogram on number of reads versus the read length without log transformation, weighted histograms of read lengths with and without log transformation, a cumulative read versus read length plot, and a dot plot reporting average read quality versus the read length.

Taxonomic classification

Taxonomy of the raw reads is determined with the help of Kraken2 in MetaGalaxy and Ivens used WIMP(ONT)(Table 5**Error! Reference source not found.**). MetaGalaxy assigned 86.62% of reads to the *Escherichia* genus, 3.85% of *Enterococcus*, and 6.5% of reads are assigned to *Homo*. This means that 3.03% of reads are assigned to different taxons below 2% abundance or are unclassified. Analysis with WIMP shows that 86.43% of the reads are assigned to *Escherichia,* 3.85% to *Enterococcus,* and 6.5% to *Homo sapiens*. WIMP and MetaGalaxy produce very similar results, with MetaGalaxy assigning 0.19% more reads to *Escherichia* than WIMP. In this regard, WIMP and MetaGalaxy are just as good at identifying organisms, however, the MetaGalaxy output does displays by default all species found, not just the genus. Whereas in WIMP, the user has to select a genus before they can see which species have been found.

Table 5: Comparison of taxonomic classification. M. Ivens used WIMP and Taxonomer for taxonomic classification versus kraken2 that is used by MetaGalaxy.

Taxonomy by M. Ivens			**Taxonomy by MetaGalaxy**		
WIMP			Kraken2		
Taxonomy	Cumulative reads	Abundance (%)	Taxonomy	Cumulative reads	Abundance (%)
Escherichia	1,118,029	86.43	*Escherichia*	1,120,718	86.62
Enterococcus	49,852	3.85	*Enterococcus*	49,777	3.85
Homo sapiens	84,120	6.50	*Homo sapiens*	84,085	6.50

MetaGalaxy also produces an interactive Krona pie chart(Figure 5) of the found taxonomy. The pie chart displays the taxonomic abundance in percentage rounded to a whole number. The user can also zoom in on every taxon to get more detailed information on the organisms found on that taxon. In the Krona pie chart *Escherichia* has an abundance of 87%, *Enterococcus* has an abundance of 4%, and *Homo sapiens* has an abundance of 6%. The chart also shows that 0.6% of reads are unidentified.

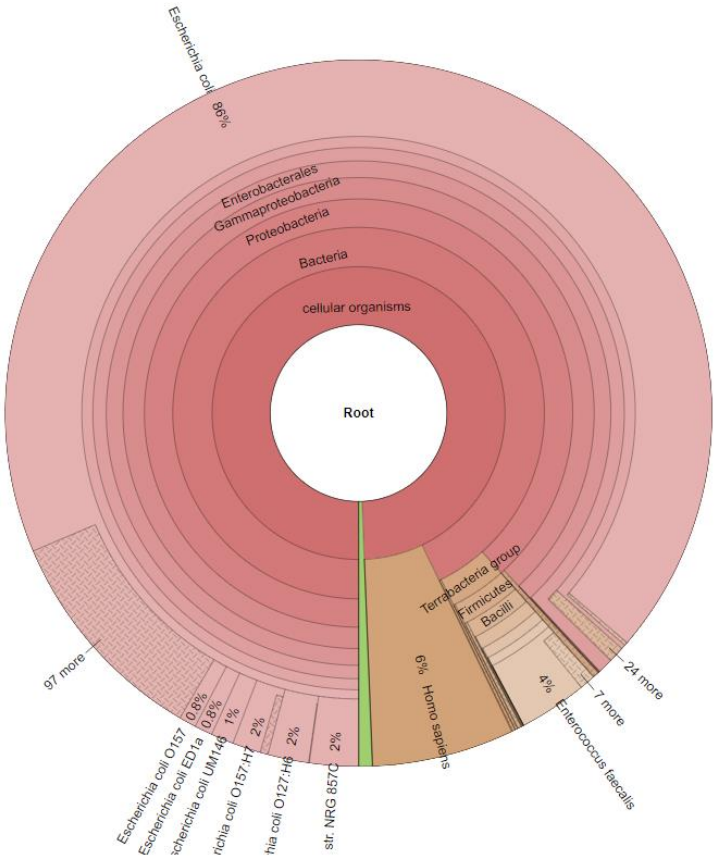

Figure 5: Krona taxonomy pie chart of BC04. Abundance is represented in percentages. The small green slice is 0.6% of unclassified reads.

Anti-microbial resistance

Identification of AMR genes was done by ABRicate for MetaGalaxy and ARMA CARD for Ivens's analysis(Table 6). Ivens performed the AMR analysis on the raw reads, whilst MetaGalaxy does it on binned contigs. Therefore, the results are represented differently. MetaGalaxy will filter out any AMR gene with either the coverage or identity below 60%. A total of four genes are identified: mdf(A), tet(34), blaCFE-1, and lsa(A). Out of these four, only one is assigned to *E. faecalis* and that is lsa(A). Ivens found tet and lsa, but interestingly did not find the mdf or blaCFE-1 gene. Lsa is an AMR gene that can be found in *E. faecalis* and was found in both the ARMA CARD search as well as in MetaGalaxy. This means there is a high likelihood that this gene is present in the bacteria. Lsa codes for multiple resistances including: Lincomycin, Clindamycin, and Dalfopristin. *E. faecalis* was not phenotypically tested(Appendix IV: Anti-microbial resistance) for this resistance, so it is not possible to determine if the bacteria also has the phenotypic trait. Tet was also detected in both resistance typing, though with a low abundance in Ivens AMRA CARD search. Unfortunately, *E. coli* also was not tested for tetracycline resistance, so the phenotypic profile was not determined and therefor it is impossible to determine if this gene is also active in this *E. coli*.

Table 6: Comparison of AMR typing between M. Ivens and MetaGalaxy. Ivens used ARMA CARD for AMR typing, whilst MetaGalaxy utilised ABRicate and the ResFinder database for AMR typing.

AMR typing by M. Ivens			**AMR typing by MetaGalaxy**		
AMR gene	Aligned hits	Average identity (%)	AMR gene	Coverage / Identity (%)	Bacteria
ACT-5	48	70	mdf(A)_1	100.00 / 97.97	*E. coli*
ACT-14	11	70	tet(34)_1	76.34 / 74.79	*E. coli*
APH(3')-Ia	2	90	blaCFE-1_1	95.64 / 71.66	*E. coli*
dfrA14	2	90	lsa(A)_1	100.00 / 98.87	*E. faecalis*
lsaA	79	90			
OXA-48	3	90			
OXA-162	2	85			
tetM	7	85			

The ACT and OXA genes were not found in the bins, but were found in the plasmid reads(Appendix IV: Anti-microbial resistance) which were isolated before assembly. Out of a total of 89 AMR genes found on plasmid reads; MetaGalaxy found 36 aligned hits with ACT genes, with an average identity of 74.43% and 5 aligned hits with OXA genes, with an average identity of 93.48%. It is unlikely that OXA genes are present, since OXA codes for multiple drug resistances including amoxicillin. However, phenotypic testing showed that both bacteria are susceptible to amoxicillin. A significant number of aligned reads were aligned with ACT genes, a gene that codes for class C β-lactamases, better known as AmpC β-lactamases. ACT genes cause resistance against cephalotins, cefoxitins, and penicillins, including amoxicillin. The genes were found on plasmids, which means that a specific pathway might have to be activated before the bacteria show any kind of antibiotic resistance. The dfrA gene was found twice by MetaGalaxy with an average identity of 92.65% and APH genes were found eight times with an average of 90.92% identity. dfrA causes trimethoprim and co-trimaxazol resistance and APH causes resistance to Fosfomycin and gentamicine. Whilst *E. coli* was tested against all these resistances, *E. faecalis* was only tested against gentamicine resistance. Since the plasmid reads were not assigned to any taxonomy, these genes could be present in *E. faecalis*. Only a few reads of each gene were found indicating that further testing, including phenotypic testing, is necessary to validate the existance of these resistances.

Assembly

Assembly and polishing was only performed by MetaGalaxy and resulted in 33 contigs(Figure 6), with two contigs being larger than 50kpb and an N50 of 4.68Mbp. The largest contig is 4.68Mpb and has been assigned by metaQUAST to *E. coli* and second largest contig of 2.71Mbp has been assigned to *E. faecalis.* The total assembled length is 7.906 Mbp and metaQUAST successfully aligned 6.69Mpb to the reference genomes. This results in a genome fraction of 83.02%.

A total of 194 misassembles were detected, all of which were relocations. Meaning that the left flanking sequence aligns over 1kpb away from the right flanking sequence or they overlap by more than 1kpb. The assembly also contains 15,979 indels, where only 439 are larger than 5bp. The total length of indels is 29.6kpb, so the average length of each indel is 1.85bp. These small insertions and deletions are most likely due to inaccuracies in the data as well as the assembler, but they do not have a drastic negative effect on the assembly quality.

Only the unpolished assembly can be visualised(Figure 6), since polishing does not produce assembly graphs. The assembly image shows two large contigs, one of which is almost completely circular. The small black connection between the two edges indicate that there are reads connecting to each other, but there are too few reads to conclusively close the genome. The 29 smaller contigs below the two large contigs are most likely plasmids that were not isolated after filtering the reads or parts of either the *E. coli* or *E. faecalis* genome that could not be incorporated into the larger contigs.

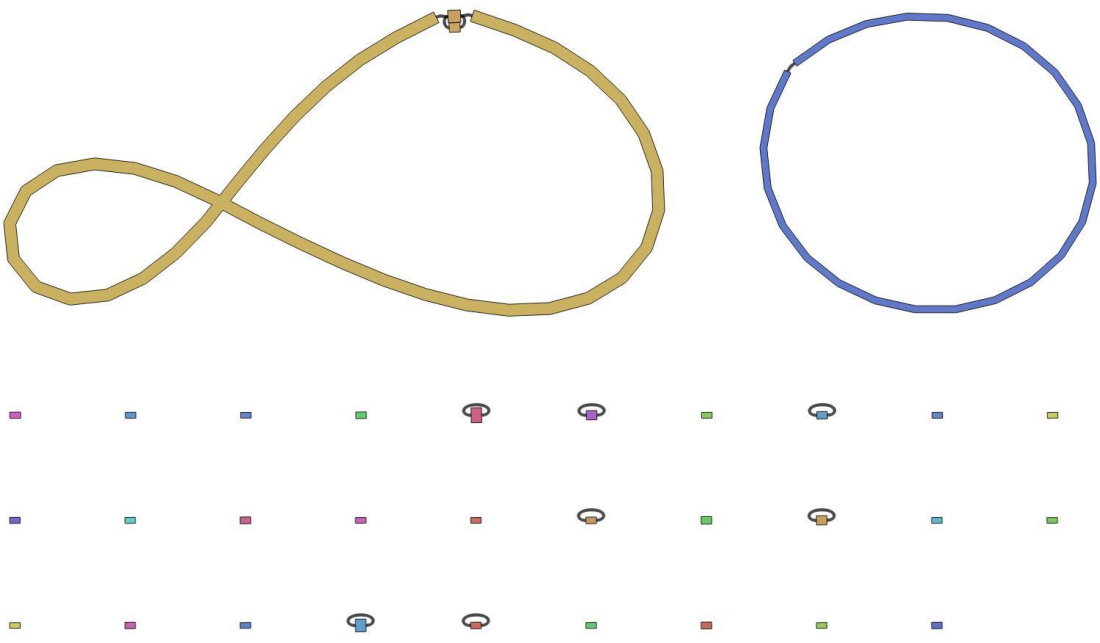

Figure 6: Visualisation of the unpolished assembly. The large contig in yellow has a size of 4.68Mbp and is the genome of E. coli. The large blue contig is E. faecalis and has a size of 2.71Mbp. The three rows of small contigs can either be plasmids or parts of the genome that could not be incorporated into the larger genome.

Binning

metaBAT2 binned the contigs into two different contigs. These contigs were then annotated by CAT/BAT(Table 7). The first bin was classified based on 5489/5505 ORFs and the second bin was classified based on 4027/4067 ORFs(Appendix V: BinsAppendix V: Bins).The bins have a fraction score of 0.1 or lower. This could cause some problems during different analyses, since below a score of 0.5 there is a chance that bins get multiple classifications.

Table 7: Annotated bins. Bins were annotated with CAT/BAT.

Bin	lineage	genus	species
bin.1.fa	1;131567;2;1224;1236;91347;543;561;562	Escherichia: 0.10	Escherichia coli: 0.08
bin.2.fa	1;131567;2;1783272;1239;91061;186826;81852;1350;1351	Enterococcus: 0.07	Enterococcus faecalis: 0.05

Galaxy Workflow

Analysis on Galaxy yielded similar results, with the exception of binning and a lower quality assembly. After assembly and polishing, Galaxy produced 37 contigs, with the largest contig being 4.68Mbp and an N50 of 4.68Mpb(Appendix I: metaQUAST report). The total contig length is 7.91 Mbp, which is slightly larger than the total contig length produced by the pipeline.

The Galaxy workflow found the same AMR genes, though at a lower coverage and Identity(Table 8). Galaxy found an additional tet gene, tetB(60). This gene was not detected by MetaGalaxy, since it does not report on AMR genes with a coverage below 60%. Therefore this gene was filtered out by the pipeline. Galaxy is unable to filter on Coverage and therefor will report on genes without taking coverage into consideration.

Table 8: AMR genes foundon contigs by ABRicate on Galaxy.

AMR gene	Coverage / Identity (%)	Contig
mdf(A)_1	99.68 / 97.65	Contig_7
tet(34)_1	76.34 / 74.79	Contig_7
blaCFE-1_1	95.64 / 71.47	Contig_7
tetB(60)_1	13.28 / 71.66	Contig_9
lsa(A)_1	99.93 / 98.80	Contig_9

Instead of binning and annotating bins, Galaxy annotates genes with Prokka and performs four MLST analyses to taxonomically identify the organisms in the sample. The MLST analyses are done on the four FASTA format outputs, fna, faa, fnn, and fsa that Prokka produces. MLST only reports on a single organism, so for Galaxy to report on multiple organisms MLST has to be performed multiple times. Since *E. coli* is more abundant in this sample, it is found more often than *E. faecalis*(Table 9).

Table 9: MLST output from galaxy. The first column displays the input file name, the second column displays the name of the identified organisms, the third column displays the found sequence type(ST), and the fourth through tenth column display the alleles used to type the organism.

Prokka on data 26: fna	efaecalis	-	gdh(71?)	gyd(2)	pstS(55?)	gki(31?)	aroE(10?)	xpt(2)	yqiL(7)
Prokka on data 26: faa	-	-							
Prokka on data 26: ffn	ecoli	-	adk(~13)	fumC(1176?)	gyrB(592?)	icd(13)	mdh(741?)	purA(37)	recA(25)
Prokka on data 26: fsa	efaecalis	-	gdh(71?)	gyd(2)	pstS(55?)	gki(31?)	aroE(10?)	xpt(2)	yqiL(7)

Discussion

MetaGalaxy is able to accurately and efficiently perform metagenomic analysis and is able to produce its first result within the first few minutes after starting the analysis. The results are just as good as existing programmes, like WIMP and when compared to ARMA CARD, MetaGalaxy is able to detect the same AMR genes. Unlike ARMA CARD, the pipeline can also identify which organism has a particular AMR gene based on binned contigs. Since the plasmid reads are also isolated early on, the AMR genes that are present on the plasmids can also be identified. Unfortunately, these genes cannot be assigned to an organism without more additional analyses.

Multiple genomes can be recovered by MetaGalaxy if a good data set, like from Ivens, is used. Binning allows MetaGalaxy to separate genomes and identify which organisms are present in a sample, perform AMR typing, and allows users to use the recovered organisms for additional downstream analysis.

A stable release of MetaGalaxy is available on GitHub, however, there are still additions that can improve MetaGalaxy in the future. To increase versatility and let advanced users tune the pipeline to their own needs, an advanced options settings would improve usability significantly. Currently, users can only change the output directory, the amount of threads MetaGalaxy is allowed to use, and the estimated genome size for the assembly. All other parameters are fixed and can only be changed by directly editing the code. More options would allow users to more easily finetune the pipeline for their specific purposes. Such extra options would be: changing the amount of polishing rounds, identity and coverage cut offs for AMR typing and plasmid isolation, minimum length and quality cut off during the filtering of raw reads, and an option to select if the user wants to analyse only the discovered bins or wants the bins and the unbinned contigs.

Anaconda3 is a programme that allows its users to create virtual python3 environments where they can install programmes and pipelines without having to build them from source code. It is possible to place MetaGalaxy on Anaconda. This would make installation seamless and the user only has to ensure that the dependencies(Appendix II: List of tools) are installed.[24]

Once MetaGalaxy is available on Anaconda3, it can be imported into any Galaxy instance. A wrapper script still has to be made for Galaxy, but once this wrapper has been made and is available from the Galaxy Toolshed. MetaGalaxy can be cloned into any Galaxy instance and Galaxy will handle the installation of the pipeline. The pipeline can then be executed like any other tool in Galaxy. Currently, a MetaGalaxy workflow exists that works slightly different than the developed pipeline, since not all tools used by MetaGalaxy are available on Galaxy. This workflow will work as a replacement for MetaGalaxy until it is available in the Toolshed.

References

1 Deurenberg, R. H. *et al.* Application of next generation sequencing in clinical microbiology and infection prevention. *Elsevier* **243**, 16-24 (2017).
2 Rossen, J. W. A., Friedrich, A. W. & Moren-Gilad, J. Practical Issues in implementing whole-genome-sequencing in routine diagnostic microbiology. *Elsevier* **24**, 355-360 (2018).
3 Forbes, J. D., Knox, N. C., Peterson, C.-L. & Reimer, A. R. Highlighting Clinical Metagenomics for Enhanced Diagnostic Decision-making: A Step Towards Wider Implementation. *Elsevier* **16**, 108-120 (2018).
4 Hasman, H. *et al.* Rapid Whole-Genome Sequencing for Detection and Characterization of Microorganisms Directly from Clinical Samples. *Journal of Clinical Microbiology* **52**, 139-146 (2014).
5 Kousandreas, T. *et al.* ANASTASIA: An Automated Metagenomic Analysis Pipeline for Novel Enzyme Discovery Exploiting Next Generation Sequencing Data. *Frontiers in Genetics* **10**, 469 (2019).
6 Uritskiy, G. V., DiRuggiero, J. & Taylor, J. MetaWRAP—a flexible pipeline for genome-resolved metagenomic data analysis. *Microbiome* **6**, 158 (2018).
7 Juul, S. *et al.* What's in my pot? Real-time species identification on the MinION™. *bioRxiv* (2015).
8 Bertrand, D. *et al.* Hybrid metagenomic assembly enables high-resolution analysis of resistance determinants and mobile elements in human microbiomes. *Nature biotechnology* **37**, 937-944 (2019).
9 Nurul, A. N. A., Muhammad, D.-D., Okomoda, V. T. & Nur, A. A. B. 16S rRNA-Based metagenomic analysis of microbial communities associated with wild Labroides dimidiatus from Karah Island, Terengganu, Malaysia. *Biotechnology Reports* **21** (2019).
10 Bagnacani, A. *et al.* in *github*.
11 Hillman-Jackson, J., Clements, D., Blankenberg, D., Taylor, J. & Nekrutenko, A. Using Galaxy to Perform Large-Scale Interactive Data Analyses. *Current protocols in bioinformatics* **Chapter 10** (2007).
12 Blankenberg, D. *et al.* Galaxy, a web-based genome analysis tool for experimentalists. *Current protocols in molecular biology* **Chaper 19** (2010).
13 Stewart, P. A. *et al. The Galaxy Platform for Reproducible Affinity Proteomic Mass Spectrometry Data Analysis*. (Humana Press, 2019).
14 Afgan, E. *et al.* The Galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2016 update. *Nucleic Acids Research* **44**, W3-W10 (2016).
15 Goecks, J., Nekrutenko, A. & Taylor, J. Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. *Genome Biology* **11**, R86 (2010).
16 De Coster, W., D'Hert, S., Schultz, D. T., Cruts, M. & Van Broeckhoven, C. NanoPack: visualizing and processing long-read sequencing data. *Bioinformatics* **34**, 2666-2669, doi:10.1093/bioinformatics/bty149 (2018).
17 Wood, D. E., Lu, J. & Langmead, B. Improved metagenomic analysis with Kraken 2. *bioRxiv*, 16, doi:10.1101/762302 (2019).
18 Ondov, B. D., Bergman, N. H. & Phillippy, A. M. Interactive metagenomic visualization in a Web browser. *BMC Bioinformatics* **12**, 385, doi:10.1186/1471-2105-12-385 (2011).
19 Carattoli, A. *et al.* In Silico Detection and Typing of Plasmids using PlasmidFinder and Plasmid Multilocus Sequence Typing. **58**, 3895-3903, doi:10.1128/AAC.02412-14 %J Antimicrobial Agents and Chemotherapy (2014).
20 Kolmogorov, M., Rayko, M., Yuan, J., Polevikov, E. & Pevzner, P. metaFlye: scalable long-read metagenome assembly using repeat graphs. 637637, doi:10.1101/637637 %J bioRxiv (2019).

21 Vaser, R., Sovic, I., Nagarajan, N. & Sikic, M. Fast and accurate de novo genome assembly from long uncorrected reads. *Genome Res* **27**, 737-746, doi:10.1101/gr.214270.116 (2017).
22 Li, H. Minimap2: pairwise alignment for nucleotide sequences. *Bioinformatics* **34**, 3094-3100, doi:10.1093/bioinformatics/bty191 (2018).
23 Gurevich, A., Saveliev, V., Vyahhi, N. & Tesler, G. QUAST: quality assessment tool for genome assemblies. *Bioinformatics* **29**, 1072-1075, doi:10.1093/bioinformatics/btt086 (2013).
24 Wick, R. R., Schultz, M. B., Zobel, J. & Holt, K. E. Bandage: interactive visualization of de novo genome assemblies. *Bioinformatics* **31**, 3350-3352, doi:10.1093/bioinformatics/btv383 (2015).
25 Kang, D. D. *et al.* MetaBAT 2: an adaptive binning algorithm for robust and efficient genome reconstruction from metagenome assemblies. *PeerJ* **7**, e7359, doi:10.7717/peerj.7359 (2019).
26 von Meijenfeldt, F. A. B., Arkhipova, K., Cambuy, D. D., Coutinho, F. H. & Dutilh, B. E. Robust taxonomic classification of uncharted microbial sequences and bins with CAT and BAT. *Genome Biol* **20**, 217, doi:10.1186/s13059-019-1817-x (2019).
27 Buchfink, B., Xie, C. & Huson, D. H. Fast and sensitive protein alignment using DIAMOND. *Nature methods* **12**, 59-60 (2014).
28 Hyatt, D. *et al.* Prodigal: prokaryotic gene recognition and translation initiation site identification. *BMC Bioinformatics* **11**, 119, doi:10.1186/1471-2105-11-119 (2010).
29 Seemann, T. Prokka: rapid prokaryotic genome annotation. *Bioinformatics* **30**, 2068-2069, doi:10.1093/bioinformatics/btu153 (2014).
30 Robinson, J. T. *et al.* Integrative genomics viewer. *Nature biotechnology* **29**, 24-26, doi:10.1038/nbt.1754 (2011).
31 Buels, R. *et al.* JBrowse: a dynamic web platform for genome visualization and analysis. *Genome biology* **17**, 66-66, doi:10.1186/s13059-016-0924-1 (2016).
32 Camacho, C. *et al.* BLAST+: architecture and applications. *BMC Bioinformatics* **10**, 421, doi:10.1186/1471-2105-10-421 (2009).
33 Eddy, S. R. Accelerated Profile HMM Searches. *PLoS Comput Biol* **7**, e1002195, doi:10.1371/journal.pcbi.1002195 (2011).
34 Matsen, F. A., Kodner, R. B. & Armbrust, E. V. pplacer: linear time maximum-likelihood and Bayesian phylogenetic placement of sequences onto a fixed reference tree. *BMC Bioinformatics* **11**, 538, doi:10.1186/1471-2105-11-538 (2010).
35 Zankari, E. *et al.* Identification of acquired antimicrobial resistance genes. *Journal of Antimicrobial Chemotherapy* **67**, 2640-2644, doi:10.1093/jac/dks261 %J Journal of Antimicrobial Chemotherapy (2012).

Appendix I: metaQUAST report

Table 10: Statistics on the polished assembly from metaQUAST by MetaGalaxy. The contigs and Lxx are reported in number of contigs and the lengths as well as the Nxx values are reported in bp

Assembly	polished_assembly
# contigs (>= 0 bp)	34
# contigs (>= 1000 bp)	31
# contigs (>= 5000 bp)	28
# contigs (>= 10000 bp)	24
# contigs (>= 25000 bp)	5
# contigs (>= 50000 bp)	2
Total length (>= 0 bp)	7906767
Total length (>= 1000 bp)	7904560
Total length (>= 5000 bp)	7895607
Total length (>= 10000 bp)	7867107
Total length (>= 25000 bp)	7553060
Total length (>= 50000 bp)	7463926
# contigs	33
Largest contig	4684535
Total length	7906491
Reference length	7793204
N50	4684535
N75	2779391
L50	1
L75	2

Table 11: Statistics on the polished assembly from metaQUAST by MetaGalaxy. The lengths and NAxx values are reported in bp. The LAxx is counted in number of aligned blocks

Assembly	polished_assembly
# misassemblies	181
# misassembled contigs	2
Misassembled contigs length	7463926
# local misassemblies	202
# scaffold gap ext. mis.	0
# scaffold gap loc. mis.	0
# unaligned mis. contigs	1
# unaligned contigs	30 + 3 part
Unaligned length	1204602
Genome fraction (%)	85.852
Duplication ratio	1.018
# N's per 100 kbp	0
# mismatches per 100 kbp	2162.96
# indels per 100 kbp	242.34
Largest alignment	300163
Total aligned length	6697899
NA50	61363
NA75	23784
LA50	33
LA75	83

Table 12: Statistics on the polished assembly from metaQUAST on Galaxy. The contigs and Lxx values are reported in number of contigs and the lengths as well as the Nxx values are reported in bp.

Assembly	polished_assembly
# contigs (>= 0 bp)	35
# contigs (>= 1000 bp)	32
# contigs (>= 5000 bp)	
# contigs (>= 10000 bp)	
# contigs (>= 25000 bp)	
# contigs (>= 50000 bp)	
Total length (>= 0 bp)	7892331
Total length (>= 1000 bp)	7890171
Total length (>= 5000 bp)	
Total length (>= 10000 bp)	
Total length (>= 25000 bp)	
Total length (>= 50000 bp)	
# contigs	35
Largest contig	4690261
Total length	7892331
Reference length	
N50	4690261
N75	2783398
L50	1
L75	2

Table 13: Statistics on the polished assembly from metaQUAST on Galaxy. The lengths and NAxx values are reported in bp. The LAxx is counted in number of aligned blocks

Assembly	polished_assembly
# misassemblies	53
# misassembled contigs	2
Misassembled contigs length	7473659
# local misassemblies	67
# scaffold gap ext. mis.	0
# scaffold gap loc. mis.	0
# unaligned mis. contigs	1
# unaligned contigs	32 + 3 part
Unaligned length	687004
Genome fraction (%)	93.14
Duplication ratio	1.007
# N's per 100 kbp	0
# mismatches per 100 kbp	721.4
# indels per 100 kbp	413.25
Largest alignment	3459324
Total aligned length	7202376
NA50	572200
NA75	115070
LA50	2
LA75	12

Appendix II: List of tools

Table 14: List of tools used by MetaGalaxy

Tool	Function	Github/Bitbucket/Nanopore Community	Author	Reference
ABRicate	Mass screening of contigs for antimicrobial and virulence genes	https://github.com/tseemann/abricate	Seemann T	
Any2fasta	Convert various sequence formats to FASTA	https://github.com/tseemann/any2fasta	Seemann T	
Bandage	Visualising assembly graphs	https://github.com/rrwick/Bandage	Wick R.R.	Wick, et al. [24]
BLAST+	Improved version of BLAST (Basic Local Alignment Tool)		Camacho C	Camacho, et al. [32]
CAT/BAT	Taxonomic classification of contigs or bins	https://github.com/dutilh/CAT	Von Meijenfelt B	von Meijenfeldt, et al. [26]
DIAMOND	Sequence aligner for protein and translated DNA	https://github.com/bbuchfink/diamond	Buchfink B	Buchfink, et al. [27]
Filtlong	Quality filtering for long reads	https://github.com/rrwick/Filtlong	Wick R.R.	
Flye	*De novo* assembler with metagenomic assembly capabilities	https://github.com/fenderglass/Flye	Kolmogorov M Lin Y	Kolmogorov, et al. [20]
Guppy	Demultiplexing reads	https://community.nanoporetech.com/protocols/Guppy-protocol/v/GPB_2003_v1_revM_14Dec2018	(ONT) Oxford Nanopore	
Hmmer	Sequence analysis using profile hidden Markov models	https://github.com/EddyRivasLab/hmmer	Eddy S	Eddy [33]
Htslib	C library for accessing file formats such as SAM and VCF	https://github.com/samtools/htslib	Samtools	
Kraken2	Taxonomic sequence classification	https://github.com/DerrickWood/kraken2	Wood D	Wood, et al. [17]
Krona	Interactive pie charts of taxonomic lineage	https://github.com/marbl/Krona	Ondov B	Ondov, et al. [18]
Medaka	Consensus polishing	https://github.com/nanoporetech/medaka	ONT	
metaBAT2	Binning contigs	https://bitbucket.org/berkeleylab/metabat	Kang D	Kang, et al. [25]
Minimap2	Pairwise sequence aligner	https://github.com/lh3/minimap2	Li H	Li [22]
NanoComp	Quality comparison of multiple datasets	https://github.com/wdecoster/nanocomp	Coster W	De Coster, et al. [16]

NanoPlot	Plotting quality control results	https://github.com/wdecoster/NanoPlot	Coster W	De Coster, et al. [16]
NanoStat	Statistic quality control of long reads	https://github.com/wdecoster/nanostat	Coster W	De Coster, et al. [16]
Pplacer	Phelogenetic placement	https://github.com/matsen/pplacer	Matsen F	Matsen, et al. [34]
Prodigal	Protein prediction	https://github.com/hyattpd/Prodigal	Hyatt D	Hyatt, et al. [28]
Quast	Quality assessment of assembled contigs	https://github.com/ablab/quast	Gurevich A	Gurevich, et al. [23]
Racon	Consensus polishing	https://github.com/lbcb-sci/racon	Vaser R	Vaser, et al. [21]
Samtools	Manipulation of next-generation sequencing data	https://github.com/samtools/samtools	samtools	

Table 15: List of databases used by MetaGalaxy

Database	Function	Author	Reference
ResFinder	Database containing AMR gene sequences	Zankari E	Zankari, et al. [35]
PlasmidFinder	Database containing plasmid specific gene sequences	Carattoli A	Carattoli, et al. [19]

Table 16: Linux dependencies needed for MetaGalaxy to run

Dependecies	Development dependencies
build-essential	qtbase5-dev
git	libqt5svg5-dev
curl	libboost-all-dev
pkg-config	libncurses5-dev
cmake	zlib1g-dev
gcc	libfreetype6-dev
g++	libpng-dev
autoconf	libbz2-dev
python-setuptools	liblzma-dev

Appendix III: NanoPlot results

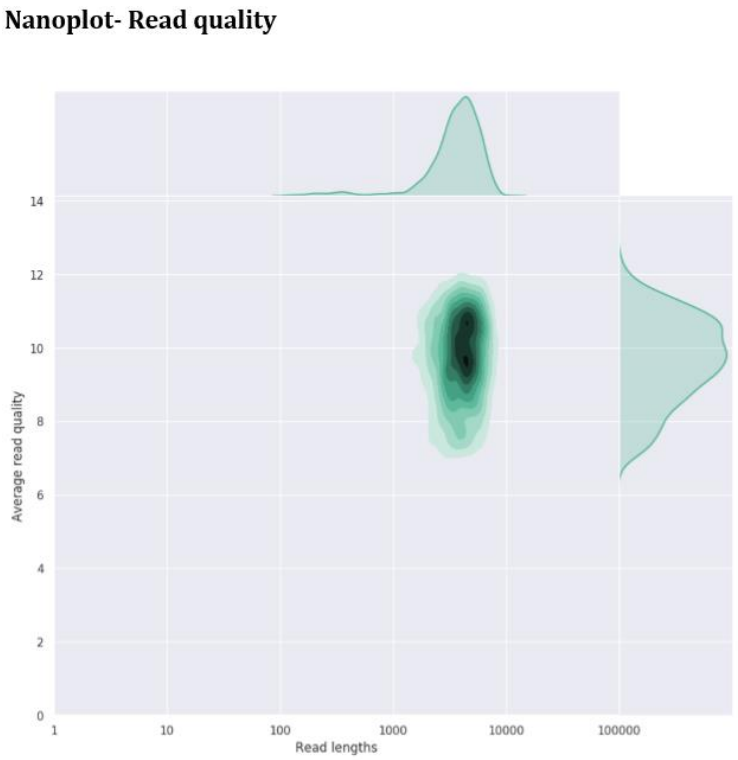

Figure 7: NanoPlot density plot by M. Ivens. The read lengths are presented on a logarithmic scale.

Figure 8: NanoPlot density plot by MetaGalaxy. The read lengths are presented on a linear scale.

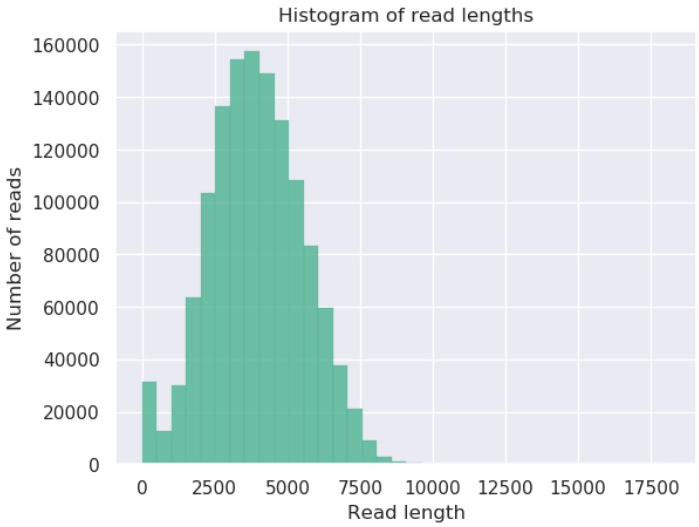

Figure 10: NanoPlot histogram by MetaGalaxy where the number of reads are plotted against the read length

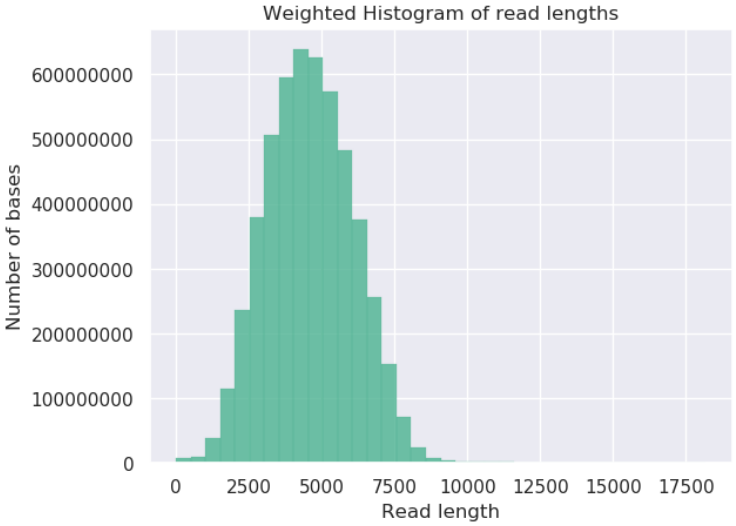

Figure 9: NanoPlot weighted histogram by MetaGalaxy where the number of bases are plotted against the read length

Figure 11: NanoPlot weighted histogram produced by MetaGalaxy. In this histogram, the number of bases are plotted against the read length on a logarithmic scale.

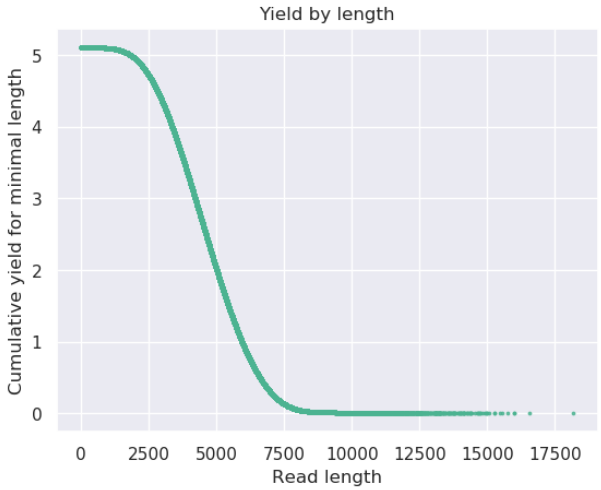

Figure 13: Nanoplot on yield by MetaGalaxy, where the yield is represented in gigabases.

Figure 12: NanoPlot dot plot by MetaGalaxy. The average quality phred score is plotted against the read length. The histograms show the rough abundance of reads.

Appendix IV: Anti-microbial resistance

Antibiotic	**UTI 1**		**UTI 3**	**UTI 4**
	E.coli	*E.feacalis*	*E.coli*	*E.coli*
Amoxicillin	S	S	R	R
Augmentin	-	-	R	S
Ceftriaxone	-	-	R	-
Ciprofloxacin	S	-	S	S
Co-trimaxazol	S	-	S	S
Fosfomycin	S	-	S	S
Gentamicine	S	S	S	S
Nitrofurantoin	S	S	S	S
Piperacillin	-	-	R	-
Trimethoprim	S	-	S	S

Figure 14: Table of phenotypic resistance typing according to the diagnostic laboratory of the Maastricht University Medical Centre by M. Ivens. UTI 1 is BC04. UTI 3 and UTI 4 are different samples from M. Ivens study and are not described in this study. "S" notes that the bacteria was found to be sensitive to the tested antibiotic, "R" notes that the bacteria was found to be resistant to the tested antibiotic, and a dash (-) notes that the bacteria was not tested against the specific anti biotic

The tables shown below are incomplete. The whole file is available on GitHub(https://github.com/mdcjansen/MetaGalaxy/blob/master/user_case_results/MetaGalaxy_results/anti-microbial_resistance/plasmid_amr_results.tab)

Table 17: AMR gene screening by ABRicate. Table is split in half to make it fit the page. The first column displays the file name, the second column displays the name of the read, the third column displays the start of the found gene sequence, the fourth column shows the end of the found gene sequence, the fifth column shows if the gene was found on the forward(+) or reverse(-) strand, the sixth colomn displays which gene has been found, the seventh column shows which proportion of the gene has been coverd, and the last column displays a visual representation of the covered gene and shows if the gene was covered(=), had a gap (/), or is unaligned(.)

#FILE	SEQUENCE	START	END	STRAND	GENE	COVERAGE	COVERAGE_MAP
filtered_barcode04_extracted_plasmids.fastq	00155e6d-6498-4041-912e-e8c314ed1349	886	1694	+	blaCMY-128_1	133-952/1146	.=======/====..
filtered_barcode04_extracted_plasmids.fastq	001816a1-28f3-43e3-8b3f-66f18421685a	2001	2353	+	tet(34)_1	66-421/465	.=======/=====.
filtered_barcode04_extracted_plasmids.fastq	003ea85c-4e31-4f03-8b90-a5f36af0c66c	2	1135	+	mdf(A)_1	130-1229/1233	.=======/======
filtered_barcode04_extracted_plasmids.fastq	00528728-eaea-4bac-81e1-bf50968b3922	504	1450	+	blaACT-6_1	139-1097/1146	.=======/======
filtered_barcode04_extracted_plasmids.fastq	0055085d-20a8-4d9f-966e-54e9ba389eaa	1531	2758	+	mdf(A)_1	1-1211/1233	========/======

Table 18: AMR gene screening by ABRicate. This table continues on after Table 17. The first column displays the file name, the second column shows amount of gaps in the file and the query, the third column shows the percentage of the gene that is covered, the fourth column displays the percentage of exact nucleotide matches, the fifth column shows the database used to generate the results, the sixth column shows the accession number associated with the found gene, the seventh column shows the gene product, and the last column shows the assumed antibiotic resistance phenotypes.

#FILE	GAPS	%COVERAGE	%IDENTITY	DATABASE	ACCESSION	PRODUCT	RESISTANCE
filtered_barcode04_extracted_plasmids.fastq	31/47	69.02	69.09	resfinder	KM985466	blaCMY-128	Amoxicillin;Amoxicillin+Clavulanic_acid;Ampicillin;Ampicillin+Clavulanic_acid;Cefotaxime;Cefoxitin;Ceftazidime;Piperacillin;Piperacillin+Tazobactam;Ticarcillin;Ticarcillin+Clavulanic_acid
filtered_barcode04_extracted_plasmids.fastq	06/07	75.48	72.07	resfinder	AB061440	tet(34)	
filtered_barcode04_extracted_plasmids.fastq	30/50	88.56	91.07	resfinder	Y08743	mdf(A)	
filtered_barcode04_extracted_plasmids.fastq	47/88	79.32	67.5	resfinder	FJ237366	blaACT-6	
filtered_barcode04_extracted_plasmids.fastq	43/91	95.21	86.4	resfinder	Y08743	mdf(A)	

Appendix V: Bins

Table 19: Bins formed by MetaGalaxy. The first column, shows the name of the bin, the second column tells whether or not the bin has been classified, the third column shows the taxonomic lineage ID's used by NCBI, followed by the column that displays lineage scores. The sixth through twelfth columns display the taxonomy found at that specific taxon along with the lineage score. The higher the lineage score, the better. The highest possible lineage score is 1.00.

# bin	Classification	Reason	Lineage	Lineage score	Superkingdom	Phylum	Class	Order	Family	Genus	Species
bin.1.fa	classified	based on 5489/5505 ORFs	1;131567;2;1224;1236;91347;543;561;562	1.00;0.93;0.81;0.24;0.21;0.21;0.20;0.10;0.08	Bacteria: 0.81	Proteo-bacteria: 0.24	Gamma-proteo bacteria: 0.21	Entero bacterales: 0.21	Entero bacteriaceae: 0.20	Escherichia: 0.10	Escherichia coli: 0.08
bin.2.fa	classified	based on 4027/4067 ORFs	1;131567;2;1783272;1239;91061;18682;81852;1350;1351	1.00;1.00;0.99;0.11;0.09;0.09;0.07;0.07;0.07;0.05	Bacteria: 0.99	Firmicutes: 0.09	Bacilli: 0.09	Lacto bacillales: 0.07	Entero coccaceae: 0.07	Enterococcus: 0.07	Enterococcus faecalis: 0.05