



UNIVERSITAT DE
BARCELONA



Universitat
Oberta
de Catalunya

Hi-C sequencing pipeline development for linking the resistome and mobilome to the microbiome of wastewater samples

David Calderón Franco

Master of Science on Bioinformatics and Biostatistics UOC-UB

Master thesis area 3: Omics data analysis and integration

Daily supervisor: Andreu Paytuví Gallart

Master thesis responsible professor: Ferran Prados Carrasco

24th June 2020



Esta obra está sujeta a una licencia de Reconocimiento-
NoComercial-SinObraDerivada [3.0 España de Creative Commons](https://creativecommons.org/licenses/by-nc-nd/3.0/es/)

Master thesis record file

Master thesis title	Hi-C sequencing pipeline development for linking the resistome and mobilome to the microbiome of wastewater samples
Author	<i>David Calderón Franco</i>
Daily Supervisor	<i>Andreu Paytuví</i>
Master thesis responsible professor	<i>Ferran Prados Carrasco</i>
Delivery (mm/aaaa)	06/2020
Master	Master of Science on Bioinformatics and Biostatistics UOC-UB
Master thesis area	Omics data analysis and integration
Language of master thesis	English
Keywords	Hi-C sequencing; metagenomics; wastewater; resistome; mobilome; microbiome
Resumen (250 palabras): Objetivos, context, metodología, resultados y conclusiones	
<p>Las plantas de tratamiento de aguas residuales son responsables de tratar y desinfectar el agua mediante procesos físicos, químicos y biológicos. Preocupa la propagación de ADN recombinante y la proliferación de resistencias a antibióticos. Los puntos críticos, como las muestras de aguas residuales, están compuestos por comunidades microbianas complejas que son capaces de incorporar ADN extracelular o intercambiar ADN entre ellos pudiendo generar lo que se conoce como ‘superbacterias’ o microorganismos resistentes a dos o más antibióticos. Hi-C permite el enlace conformacional en 3D entre elementos genéticos. Brinda la posibilidad de vincular qué genes de resistencia a antibióticos (ARGs) y elementos genéticos móviles (MGE) de comunidades complejas, como lodos activados, están en familias microbianas específicas. En esta tesis, datos Hi-C ya disponibles de lodos activados se han utilizado para desarrollar un pipeline. El análisis bioinformático ha consistido en desarrollar y publicar (en GitHub: https://github.com/davidcalfran/Linking-Hi-C-to-metagenome-data-pipeline) un pipeline para cuantificar las interacciones entre grupos microbianos específicos y contigs que contienen hits para ARG, plásmidos e integrones. El análisis de los resultados ha demostrado que hay familias de microorganismos específicos que tienen mayor predisposición a captar o intercambiar ARGs y MGEs, específicamente microorganismos de la familia <i>Aeromonadaceae</i>, <i>Neisseriaceae</i> y <i>Moraxellaceae</i>, que corresponderían a los microorganismos seleccionados como potenciales huéspedes en el estudio de referencia. Los resultados de Hi-C obtenidos con este pipeline siguen la misma tendencia que el estudio de referencia, incluso habiendo mejoras a realizar con el fin de ampliar la visualización y resolución de la interacción gen objetivo y huésped.</p>	

Abstract (250 words): Aim, application context, methodology, results and conclusions

Wastewater treatment plants are the responsible for treating and sanitizing water from different sources by physical, chemical and biological processes. One of the public concerns is the spread of engineered DNA and proliferation of antibiotic resistances released caused by transgenic microorganisms. Hotspots, such as wastewater samples, are composed by a large, complex microbial communities that, under certain environmental factors, are capable of taking up free genetic fragments or exchange DNA between them. Uptake of DNA by microorganisms could generate the well-known superbugs or microorganisms resistant to two or more antibiotics. Hi-C sequencing allows 3D-conformational linkage between genetic elements. Thus, giving the possibility to link which antibiotic resistance genes (ARGs) and mobile genetic elements (MGE) from complex communities, such as activated sludge, are enclosed inside which specific microorganisms. In this master thesis, an already published Hi-C dataset from activated sludge has been used to develop and test the pipeline. The bioinformatic analysis has consisted on developing and publishing (GitHub: <https://github.com/davidcalfran/Linking-Hi-C-to-metagenome-data-pipeline>) a pipeline in order to quantify the interactions between specific microbial clusters and contigs containing hits for ARGs, plasmids and integrons. Results have shown that there are specific microorganisms' families that have a bigger predisposition to uptake or exchange ARGs and MGEs, specifically microorganisms from the *Aeromonadaceae*, *Neisseriaceae* and *Moraxellaceae* families, which corresponded to the microorganisms selected as hosts in the reference study. Hi-C results obtained with this pipeline does still follow the same trend as the reference study even if improvements must be performed in order to magnify the host:target gene resolution.

Table of Contents

List of figures.....	6
1. Introduction.....	7
1.1. Work context and work justification.....	7
1.2. Objectives.....	10
1.3. Approach and methodology used.....	10
1.4. Working program.....	11
1.5. Short summary of obtained products.....	11
2. Material and methods.....	12
2.1. Data retrieving and quality.....	12
2.2. Metagenomic assembly.....	12
2.3. Processing of the Hi-C reads.....	12
2.4. Deconvolution of the Hi-C data.....	12
2.5. Annotation of genome clusters.....	13
2.6. Target gene detection in the metagenome assemblies.....	13
2.7. Linking plasmid, ARG, and Integron contigs to genome clusters.....	13
2.8. Script.....	13
3. Results and discussion.....	14
3.1. Data quality.....	14
3.2. Assembly quality control.....	14
3.3. Code in bash, Python and R.....	15
3.4. Binning quality control.....	16
3.5. Linking microbiome to resistome and mobilome.....	18
3.6. Comparison with reference study.....	22
4. Conclusions.....	25
5. Glossary.....	26
6. References.....	27
7. Acknowledgements.....	29
8. Supplementary material.....	30
The code.....	30
Phylogenetic tree.....	41

List of figures

- Figure 1.** Hi-C sample preparation overview. Figure adapted from (van Berkum et al., 2010). **8**
- Figure 2.** Workflow used for recovering genomes from an environmental metagenome using metagenome assembly and binning. **9**
- Figure 3.** Base sequence quality graph from the **(a)** forward Hi-C Paired End file, **(b)** reverse Hi-C Paired End file, **(c)** forward metagenomics Paired end file and **(d)** reverse metagenomics Paired end file. **15**
- Figure 4.** Graphical representation of bins/clusters completeness, heterogeneity and contamination from the CheckM output analysis. **Note:** clusters/bins displayed were the ones for proceeding with the Hi-C linkage. **18**
- Figure 5.** Hi-C links between the clusters and ARGs, identified in the wastewater sample and affiliated with most abundant bacterial families. Each edge of the phylogenetic tree represents a cluster. Only clusters having a contact with ARGs are shown. The presence or absence of a link is shown on the heatmap to the right of the tree, and the color shading represents the intensity of the normalized Hi-C link signals on logarithmic scale. **20**
- Figure 6.** Hi-C links between the clusters and plasmids, identified in the wastewater sample and affiliated with most abundant bacterial families. Each edge of the phylogenetic tree represents a cluster. Only clusters having a contact with plasmids are shown. The presence or absence of a link is shown on the heatmap to the right of the tree, and the color shading represents the intensity of the normalized Hi-C link signals on logarithmic scale. **21**
- Figure 7.** Hi-C links between the clusters and integrons, identified in the wastewater sample and affiliated with most abundant bacterial families. Each edge of the phylogenetic tree represents a cluster. Only clusters having a contact with integrons are shown. The presence or absence of a link is shown on the heatmap to the right of the tree, and the color shading represents the intensity of the normalized Hi-C link signals on logarithmic scale. **22**
- Figure 8.** Hi-C links between the clusters and ARGs, identified in the wastewater sample and affiliated with most abundant bacterial families. Each edge of the phylogenetic tree represents a cluster. Only clusters having a contact with ARGs are shown. The presence or absence of a link is shown on the heatmap to the right of the tree, and the color shading represents the intensity of the normalized Hi-C link signals on logarithmic scale. **Note:** not cutoff was applied here. **Red arrows** showing significant ARG differences and the effect of applying visualization cutoffs. **25**

1. Introduction

1.1. Work context and work justification

Wastewater may come from different origins like municipal or urban wastewater, domestic wastewater and industrial sewage, among others. Wastewater treatment plants are the responsible for treating and sanitating these different water sources by physical, chemical and biological processes for the removal of solids, pollutants and some nutrients (Van Loosdrecht et al., 1998) . One of the public concerns is the spread of engineered DNA and proliferation of antibiotic resistances released and caused by transgenic microorganisms. This is of special interest in hotspots such as wastewater treatment plants. These hotspots are composed by a large, complex microbial community that, under certain circumstances, are capable of taking up free genetic fragments or exchange DNA between organisms. Uptake of DNA fragments by natural organisms could generate, as example, unwanted resistances against a broad range of antibiotics.

Wastewater treatment plants (WWTP) operate with a natural community resembling that of natural systems (Cyzdik-Kwiatkowska and Zielińska, 2016). The incoming wastewater is much more a human gut microbiome related and the industrial organisms are more related to a community. In the WWTP they meet and have the potential to exchange DNA between urban microbiome and natural microbiome.

The study of the populations and mobile genetic elements (MGE) from activated sludge as well as other complex environments will try to clarify (i) the emission and fate of genetic fragments from industrial settings (ii) the presence of horizontal gene transfer and tracking of mobile elements between microorganisms present in wastewater plants.

Hi-C sequencing, also known as all-vs-all, uses DNA conformation capture to quantify interactions between all possible pairs of DNA fragments simultaneously (Fraser et al., 2015). Before DNA is sent to sequence, DNA found in the problem sample is cross-linked with formaldehyde (**Figure 1**), introducing bonds that freeze the interactions between genomic loci and mobile genetic elements such as plasmids that are near in the 3D space (Rocha et al.,

2016). The resulting cross-linked genome is then cut into fragments with restriction endonucleases such as *EcoR1* or *HindIII*. Those cut fragments, are marked with biotin and ligated so the resulting sequence in a proximity-based approach. The pair of sequences are individually aligned to the genome and other databases such as plasmid databased, thus determining the fragments involved in that ligation event. This allows the linkage of specific plasmids harboring integrons and transposons that may be more prolific to jump between microorganisms and which microorganisms found in complex communities can be easier a natural competent.

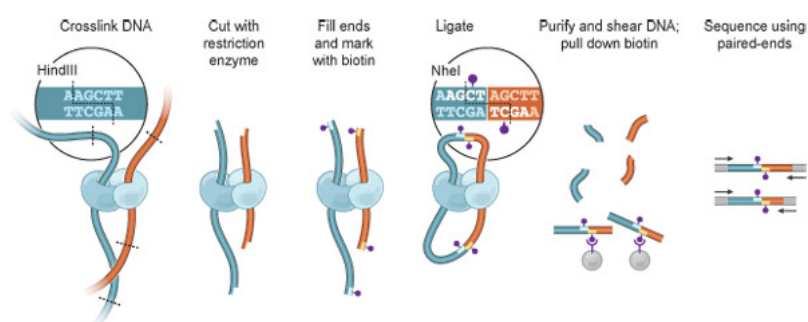


Figure 1. Hi-C sample preparation overview. Figure adapted from (Lieberman-Aiden et al., 2009).

In order to generate links between target genes such as ARGs and MGEs and specific species, it is of great importance to generate in parallel, from the same initial sample, a metagenome assembly. The workflow followed for recovering genomes is detailed in **figure 2**. Environmental genomes are sent to sequence in order to get short reads (from 150 to 300 bp). Then, it is necessary to generate what is called contigs (from contiguous) by using metagenome assemblies' software (i.e. MEGAHIT or SPADIS). Contigs are a set of overlapping DNA segments that together represent a consensus region of DNA. Contigs will be used for binning, grouping nucleotide sequences belonging to individual/similar organisms, the different environmental bacterial metagenomes based on sequence overlapping and homology (Sedlar et al., 2017). The obtained bins will be used for retrieving the different phylogenetic groups belonging to the sample. Moreover, target genes (ARGs and MGEs) will be linked to specific bins. This will allow the study of co-occurrence between specific phylogenetic families and genes of interest.

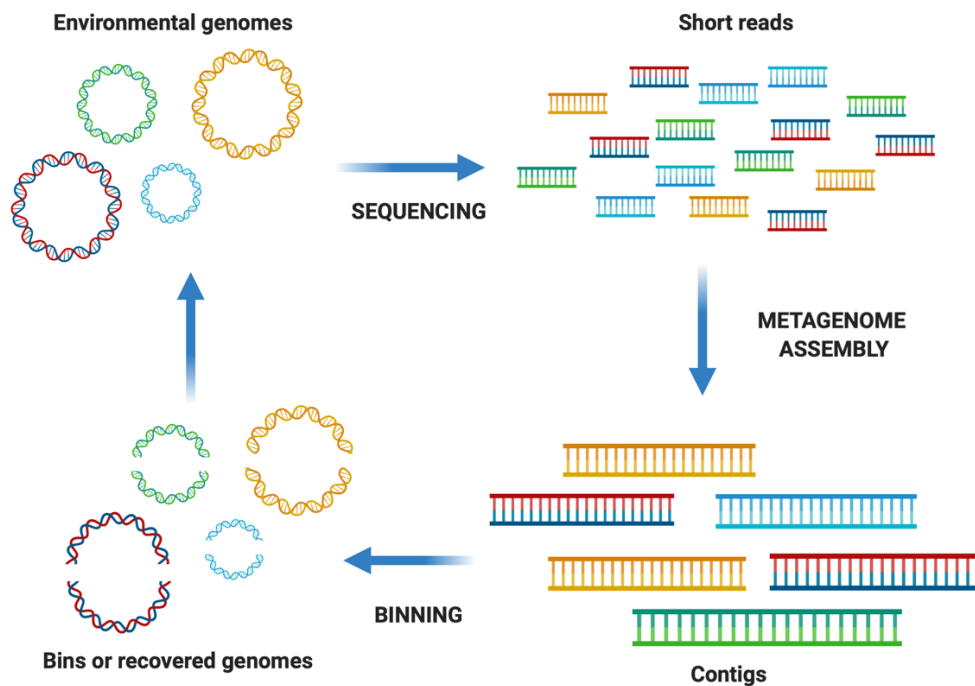


Figure 2. Workflow used for recovering genomes from an environmental metagenome using metagenome assembly and binning.

So far, Hi-C is highly implemented in epigenetics and cancer studies on the biomedical field (Burton et al., 2014; Orlando et al., 2018). However, not that many studies have been published on the environmental field where DNA is constantly being released and can be exchange and transferred. There is an increase of papers being published in high impact journals (Stalder et al., 2019), thus showing the increasing interest on this field due to its consequences on human health and risk assessments development.

There are not free bioinformatic pipelines available for analyzing Hi-C sequencing data obtained from environmental samples. The amount of relevant information Hi-C could give in order to understand better the exchange and uptake of antibiotic resistance genes and mobile genetic elements is enormous. Because of that, an easy to use bash script has been written in combination of python and R code. User will only be requested to give some inputs in order to generate an output plot that will allow researchers the understanding of the dynamics in their system.

1.2. Objectives

1. Developing a Hi-C pipeline in bash using already published datasets from wastewater samples.
2. Linking antibiotic resistant genes and mobile genetic elements to specific microorganisms.
3. Obtaining which microorganisms can be highly potential candidates to uptake, exchange and transfer targeted genes.
4. To compare analysis with already published data, where raw data was obtained from (Stalder et al., 2019)

1.3. Approach and methodology used

To develop the pipeline, a script written in bash in combination with small scripts in python have been used. Different metagenomics and Hi-C binaries have been implemented in the code above-mentioned. In order to plot the results, the environment and programming language R was used (R Foundation for Statistical Computing., 2018). R packages as *ggplot2* (Wickham, 2016), *taxize* (Scott et al., 2020) and *ggtree* (Yu, 2020) have been used for the creation of interactive figures that facilitate the understanding and discussion of the results obtained. Overall, this script will have as an objective the obtention of the results necessary to meet the goal of objectives 2, 3 and 4.

1.4. Working program

		Feb-20	Mar-20	Apr-20	May-20	Jun-20	Jul-20
Tasks	Check quality of the data						
	Align sequencing data						
	Process results						
	Graph results						
	Discuss results						
	Write report						
Milestones	Data quality is correct						
	Microbiome hit list (Metagenomics)						
	ARGs hit list (Metagenomics)						
	MGEs hit list (Metagenomics)						
	Microbiome hit list (Hi-C)						
	ARGs hit list (Hi-C)						
	MGEs hit list (Hi-C)						
	Hi-C to metagenomics data linkage						
	List of microorganisms more prone to exchange DNA						
	Write code in bash and python						
	Finished report						
Deliverables	PEC0: Work contents definition						
	PEC1: Work program						
	PEC2: Work development I						
	PEC3: Work development III						
	PEC4: Thesis writing and submission						
	PEC5a: Presentation preparation						
	PEC5b: Public thesis defense						

1.5. Short summary of obtained products

The expected results pursued during this master thesis are:

- Working plan
- Report
- Virtual presentation
- Project auto-evaluation

2. Material and methods

2.1. Data retrieving and quality

The metagenomic and the Hi-C set of data were retrieved from the NCBI BioSample database. The shotgun reads accession number ID: SRX5057658. The Hi-C reads accession number ID: SRX5057659. They both belong to the BioProject ID: PRJNA506462. The quality of the acquired Illumina reads was assessed by FastQC version 0.11.9 with default parameters (Andrews, 2010).

2.2. Metagenomic assembly

Shotgun metagenomic assembly was created using Megahit (version 1.0.2) with default parameters (Li et al., 2015). De novo assemblies were assessed using QUAST (version 5.1) with default parameters (Gurevich et al., 2013).

2.3. Processing of the Hi-C reads

The set of Hi-C reads was mapped to the metagenomic assembly. Mapping was done using the Burrow-Wheeler alignment tool BWA-MEM (version 0.7.17) (Li, 2013), obtaining a SAM file. SAM file was sorted by name using samtools (version 1.10) (Li et al., 2009).

2.4. Deconvolution of the Hi-C data

Binning process was performed using bin3C (version 0.1.1) with default parameters (Demaere and Darling, 2019), an unsupervised method that exploits the hierarchical nature of Hi-C interaction rates to resolve metagenome-assembled genomes (MAGs) using a single time point. For the binning process, MluCI enzyme was selected as the digestion enzyme used for Hi-C library construction. Bin3C pipeline makes use of the Infomap network clustering algorithm. Assembly output from Megahit and the sorted-by-name result from the Hi-C alignment process were the input data.

2.5. Annotation of genome clusters

Genome clusters were compared with RefSeq genomes (refseq.genomes.k21.s1000.msh) using Mash (version 1.1) (Ondov et al., 2016) to identify any close database matches for new genome clusters. Genome clusters were analyzed using the CheckM lineage_wf (version 1,1,2) workflow (Parks et al., 2015) to assess genome quality and estimate high-level phylogenetic placements for each cluster based on single-copy marker gene analysis. Some genome clusters were excluded on the basis of low completeness, high heterogeneity and high contamination.

2.6. Target gene detection in the metagenome assemblies

Detection of target gene from the contigs was done using BLASTn (version 2.6.0+) with the option E-value < 1e-20. When a contig had multiple hits for the same locus, the best hit was selected (lowest e-value). Detection of the ARGs and plasmids were, respectively, done using the MEGARes database (Doster et al., 2020) and the PlasmidFinder database (Carattoli et al., 2014), accessed in May 2020. Detection of the class 1, 2, and 3 Integron integrase genes was done using the reference sequences AB709942 (intI1), FQ482074 (intI1delta1), JX566770 (intI1R32_N39 aa329337 mutated + 35aa), JX469830 (intI2) and EF467661 (intI3) as Stalder et al. (2019) did.

2.7. Linking plasmid, ARG, and Integron contigs to genome clusters

For each target gene identified in a contig, we considered all Hi-C linkages between the contig containing the target gene and any other contig present in that cluster/MAG. We normalized the number of Hi-C contacts according to abundance of the genome cluster. Phylogenetic analysis of the genomes linked to contigs of interest was performed using the R package taxize (Chamberlain and Szöcs, 2013) and ggplot2 (Wickham, 2016).

2.8. Script

The script was written in bash, Python 2.7 and RStudio. Then, it was uploaded in the public GitHub repository (<https://github.com/davidcalfran/Linking-Hi-C-to-metagenome-data-pipeline>).

3. Results and discussion

3.1. Data quality

Illumina reads data quality were assessed by FastQC and are displayed in **Figure 3a-d**. For this study, paired-end reads were kept in its original length as data quality was considered to be high enough for avoiding future false positives. Despite that, reads from the reverse metagenomics file (**Fig. 3d**) would be recommended to be trimmed at around 145 bp for increasing accuracy.

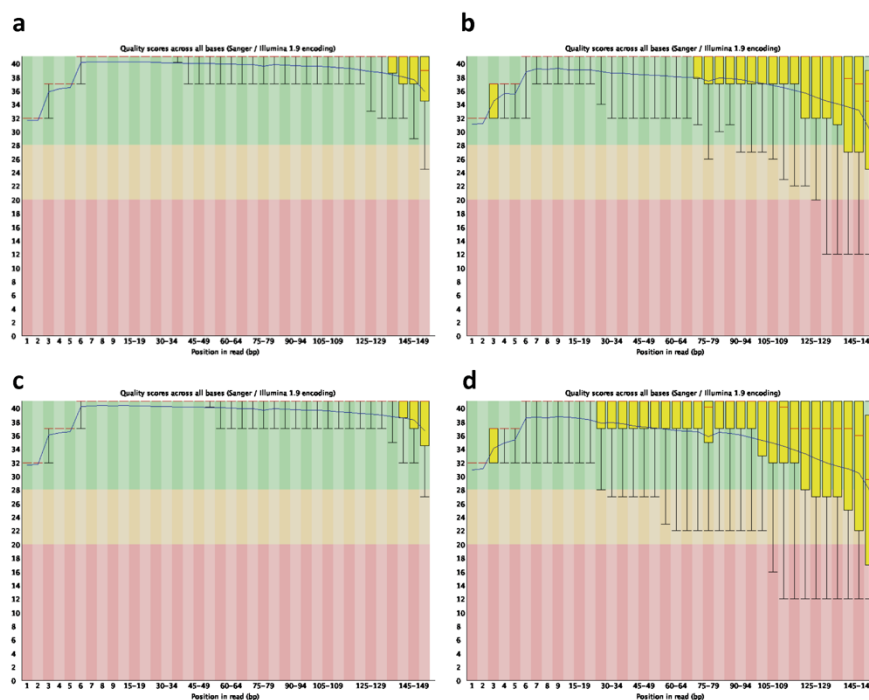


Figure 3. Base sequence quality graph from the **(a)** forward Hi-C Paired End file, **(b)** reverse Hi-C Paired End file, **(c)** forward metagenomics Paired end file and **(d)** reverse metagenomics Paired end file.

3.2. Assembly quality control

For assessing the quality of the metagenome assembly done with MEGAHIT, the QUAST software was utilized. All statistics are based on contigs of size ≥ 500 bp and total length include all contigs. The metagenomic assembly quality control results are summarized in **table 1**. The majority of the contigs generated were longer than 1000 bp due the threshold applied (contgs >1000).

Table 1. Table summarizing the quality parameters calculated by QUAST of the metagenomic assembly

<i>Statistics</i>	<i>Assembly out of MEGAHIT</i>
<i>Number of contigs</i>	210'786
<i>Number of contigs (>= 0 bp)</i>	210'786
<i>Number of contigs (>= 1000 bp)</i>	210'786
<i>Number of contigs (>= 10000 bp)</i>	12'404
<i>Number of contigs (>= 100000 bp)</i>	25
<i>Number of contigs (>= 1000000 bp)</i>	0
<i>Largest contig</i>	249'622
<i>Total length</i>	921'213'071
<i>Total length (>= 0 bp)</i>	921'213'071
<i>Total length (>= 1000 bp)</i>	921'213'071
<i>Total length (>= 10000 bp)</i>	213'350'468
<i>Total length (>= 100000 bp)</i>	3'591'844
<i>Total length (>= 1000000 bp)</i>	0
<i>N50</i>	4712
<i>N75</i>	2903
<i>L50</i>	50'374
<i>L75</i>	111'998
<i>GC (%)</i>	48.18

3.3. Code in bash, Python and R

This code explains step by step how to obtain cluster-link heatmaps for ARGs, plasmids and integrons. The details about the code are explained in the [supplementary material](#).

Considerations before executing the code

In order to run this code successfully, there are some software and binaries that must be locally installed and added to the path. Those requirements are the following:

- bwa is assumed to be installed in path.
- samtools is assumed to be installed in path.
- blastn is assumed to be installed in path.
- CheckM is assumed to be installed in path.
- Bin3C binaries are supposed to be downloaded and be saved in current folder as a folder named "bin3C".
- **refseq.genomes.k21s1000.msh** is supposed to be downloaded and be in current folder. It is necessary for mash. It can be downloaded here:

<https://mash.readthedocs.io/en/latest/tutorials.html>

It is important to save it with the following name: refseq.genome.k21s1000.msh.

- Python2 is assumed to be installed in path.

For running the script, the following code had to be written in the terminal on the folder with all the previously mentioned files:

```
bash Hic_MG_pipeline.sh assembly.fa HiC_1.fastq HiC_2.fastq megares_database_v1.01.fasta  
enterobacteriaceae.fsa integrase_database.fa 1
```

Where:

- assembly.fa is your metagenome assembly generated with MEGAHIT, SPADes or other assembly software.
- HiC_1/2.fastq are the Hi-c sequencing data.
- megares_database_v1.01.fasta is the antibiotic resistance genes database
- enterobacteriaceae.fsa is the plasmid database from PlasmidFinder.
- integrase_database.fa is the manually-generated integrons database
- "1" is the number that starts running the script.

3.4. Binning quality control

For assessing the binning quality, CheckM with the lineage_wf option was used. This step was crucial in order to proceed with the analysis. It gave information on how complete are the bins generated and how much contamination there was. From here, it filtered out the low-quality bins: those whose completeness was below 70% and contamination above 15%. After the binning process, 2250 bins or MAGs were generated. CheckM analysis showed that only 55 bins completeness and contamination were good enough to be used for further analysis (Figure 4).

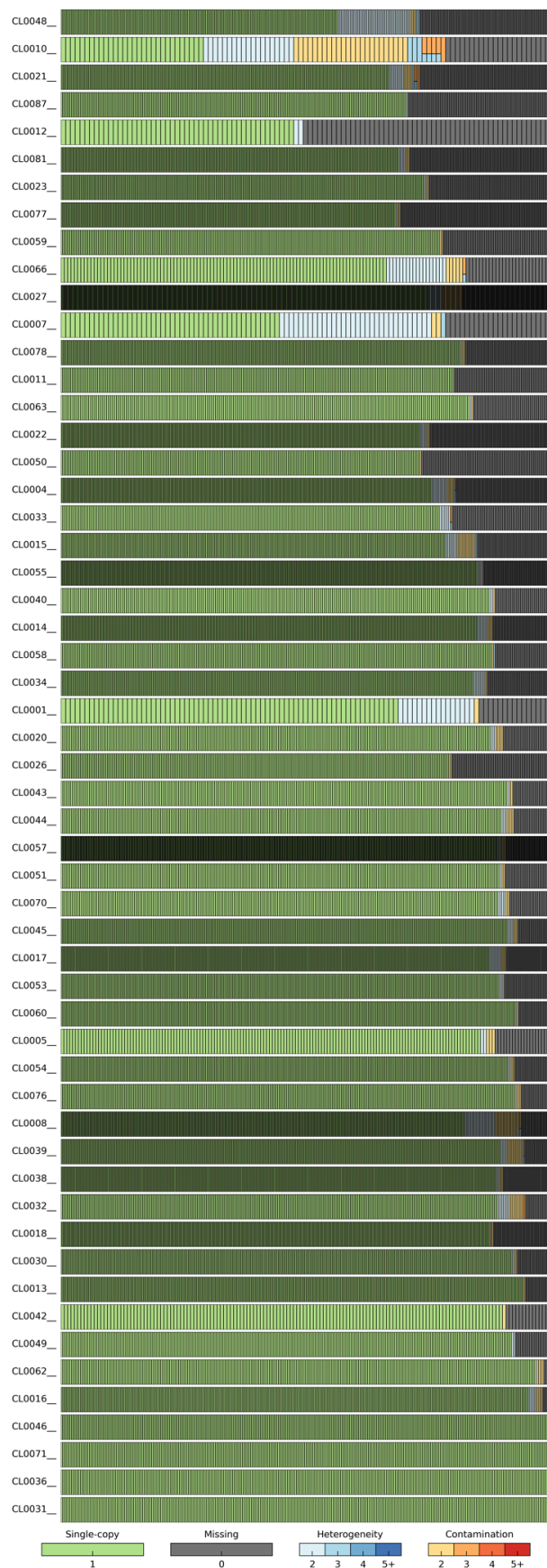


Figure 4. Graphical representation of bins/clusters completeness, heterogeneity and contamination from the CheckM output analysis. **Note:** clusters/bins displayed were the ones for proceeding with the Hi-C linkage.

3.5. Linking microbiome to resistome and mobilome

In this section, antibiotic resistance genes (ARGs), plasmids and integrons were Hi-C linked with specific microorganisms' species in order to elucidate which bacterial families are more prone to be hosts that can uptake and exchange these potentially dangerous genetic elements. The term cluster is used to describe a cohesive group of contigs belonging to a genome of a microorganism, also known as MAGs.

For doing that, all the contigs generated were blasted against one of the target genes databases to retrieve which of them were present and in which affluence. BLASTn hits were assigned to specific contigs. Those contigs were then checked to which phylogenetic clusters belonged. Then, the sum of interactions between the hit contig (i.e. containing and ARG) and all the contigs from the clusters it belonged was calculated. This sum of interactions was further normalized by the number of contigs that the cluster had. Only clusters having a contact with one of the target gene are shown: ARGs (**Figure 5**), plasmids (**Figure 6**) and integrons (**Figure 7**).

Hi-C links between ARGs and specific bacteria are depicted in **figure 5** on top of placing phylogenetic belonging to each cluster. ARGs were mostly linked to contigs in clusters related to the Gamma- and Betaproteobacteria (**Figure 5**). The other links to ARGs were mostly associated with clusters affiliated with the Firmicutes, and very few with clusters affiliated with Actinobacteria, Alpha-proteobacteria, Fusobacteria and Bacteroides.

Mutations in the genes for the subunits *GyrA* and *ParC* of the target enzymes DNA gyrase and topoisomerase IV are important mechanisms of resistance in quinolone-resistant bacteria, especially in bacteria belonging to the *Neisseriabaceae* family (Lindbäck et al., 2002). *Neisseria meningitis* from clinical isolates has also been described as a potential reservoir for rifampicin resistance (*rpoB*) (Skoczynska et al., 2009). Due to the abuse of fluoroquinolones and rifampicin in the last decades, is not surprisingly to observe that one of the most ubiquitous ARGs are *gyrA* and *rpoB*.

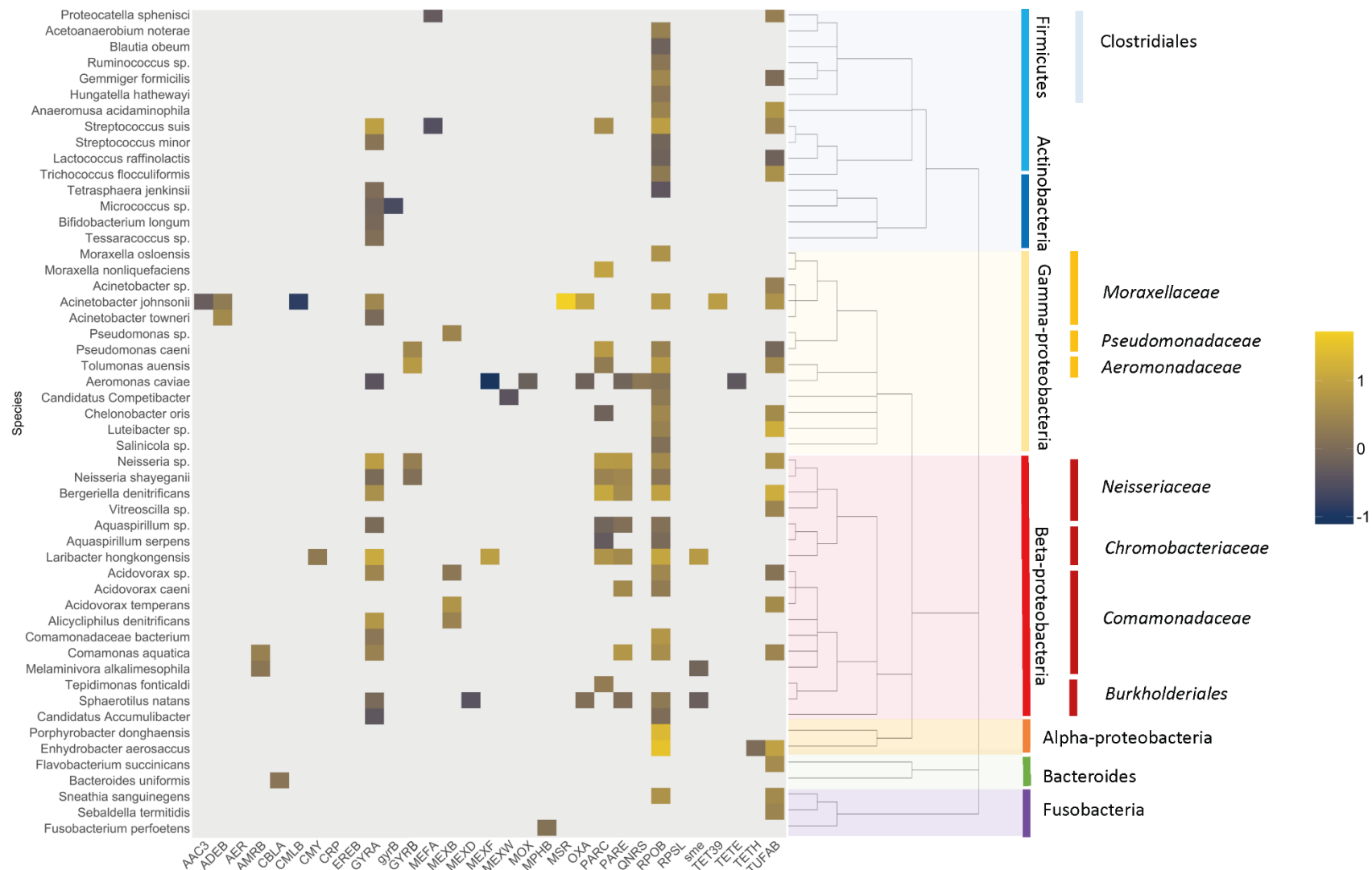


Figure 5. Hi-C links between the clusters and ARGs, identified in the wastewater sample and affiliated with most abundant bacterial families. Each edge of the phylogenetic tree represents a cluster. Only clusters having a contact with ARGs are shown. The presence or absence of a link is shown on the heatmap to the right of the tree, and the color shading represents the intensity of the normalized Hi-C link signals on logarithmic scale. **Note:** cutoff of MAGs containing less than 150 contigs.

Bacteria belonging to the *Moraxallaceae* and *Comamonadaceae* have also been described as potential reservoirs for ARG in water for ornamental fish samples (Gerzova et al., 2014). Special interest must be given to *Acinetobacter johnsonii*, bacteria found in water and soil, on human skin, and in human feces. It is found to be the species harboring the widest range of ARGs in these samples (9 different ARGs). In general, the *Acinetobacter sp.*, specifically *Acinetobacter baumannii* (not found in these samples), has been described as one of the most challenging pathogens due to the emergence and widespread of antibiotic resistance (Xie et al., 2018). Thus, special interest in the study of the prevalence and ARGs carrier must be conducted.

Overall, the bacterial taxa that had the most contacts with known ARGs were affiliated with the *Moraxallaceae*, *Aeromonadaceae* and *Burkholderiales*, families typically associated with aquatic environments. Hi-C linking suggests that *Aeromonadaceae*, *Moraxellaceae*, *Neisseriabaceae*, *Chromobacteriaceae* and *Commamonadeceae* are the most likely reservoirs of ARGs in this wastewater samples.

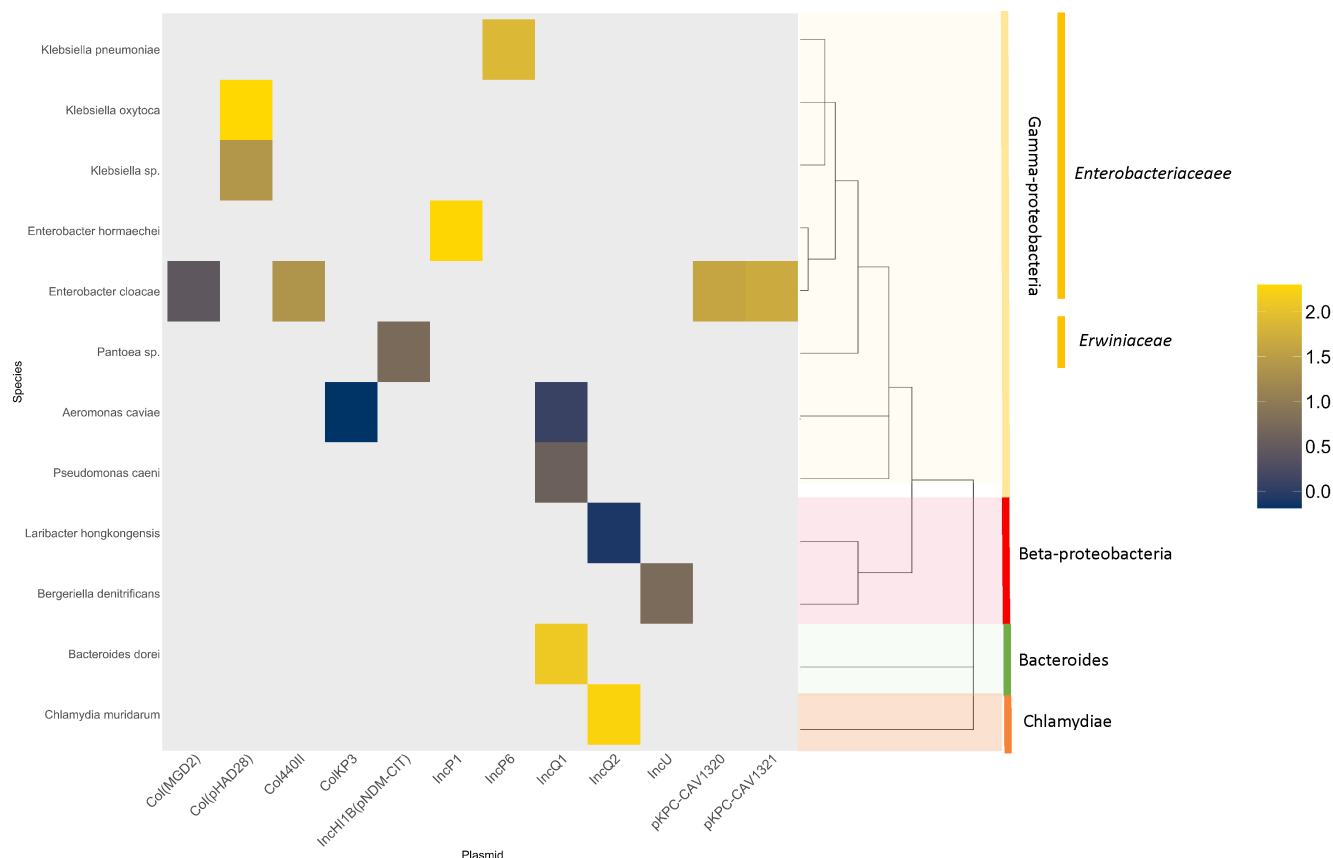


Figure 6. Hi-C links between the clusters and plasmids, identified in the wastewater sample and affiliated with most abundant bacterial families. Each edge of the phylogenetic tree represents a cluster. Only clusters having a contact with plasmids are shown. The presence or absence of a link is shown on the heatmap to the right of the tree, and the color shading represents the intensity of the normalized Hi-C link signals on logarithmic scale.

Regarding the plasmidome analysis, it can clearly be observed that the number of hits were significantly lower when compared to the ARGs analysis (**Figure 6**). Two known groups of broad-host-range (BHR) plasmids (IncQ-1 and IncQ-2) were linked to clusters spanning both *Beta*- and *Gammaproteobacteria* but also *Bacteroides* and *Chlamydiae*. IncP-1 plasmid, also known to be a BHR plasmid, was mainly highlighted in the *Enterobacteriaceae* family. In contrast, markers for narrow-host-range (NHR) plasmids (colE1-type plasmids) were almost exclusively linked to clusters belonging to the *Enterobacteriaceae*. Overall, it is shown here that IncQ (1 and 2) plasmids had the broadest range of putative hosts, followed by the colE1-type plasmids.

Among the integrons, class 1 integrons (AB609942 and JX566770, specially the last one) exhibited links to 7 out of the 10 clusters with hits within a broad range of bacterial families, specifically in the Gamma-proteobacteria and less in the *Bacteroides*, Actinobacteria and *Chlamydiae* families (**Figure 7**).

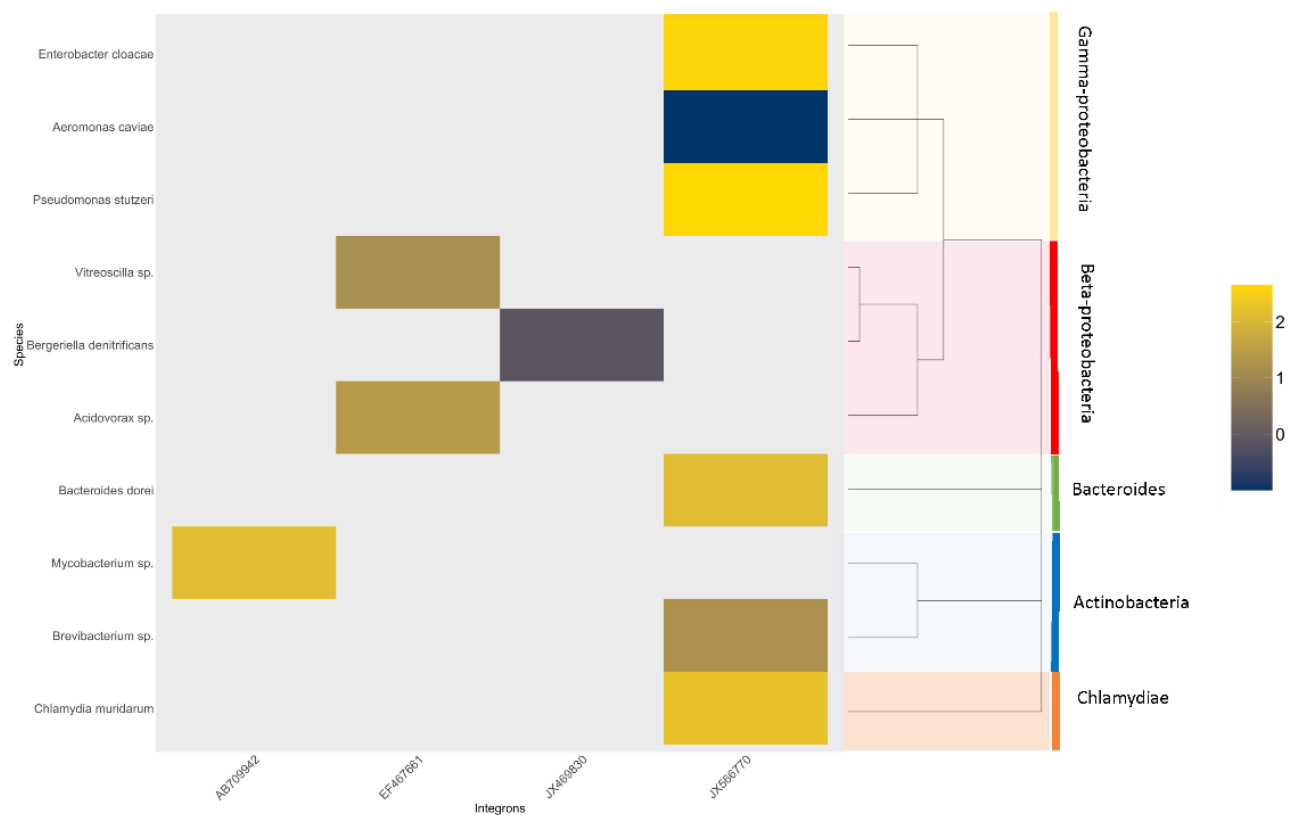


Figure 7. Hi-C links between the clusters and integrons, identified in the wastewater sample and affiliated with most abundant bacterial families. Each edge of the phylogenetic tree represents a cluster. Only clusters having a contact with integrons are shown. The presence or absence of a link is shown on the heatmap to the right of the tree, and the color shading represents the intensity of the normalized Hi-C link signals on logarithmic scale.

No links were found within the Beta-proteobacteria. Class 2 integrons (JX49830) and class 3 Integron (EF467661) were associated to Beta-proteobacteria bacteria, specially affiliated to the *Neisseriaceae* family (*Vitreoscilla sp.* and *Bergeriella denitrificans*).

To sum up, Hi-C links can indeed help determining taxonomic placement of the hosts of ARGs and MGEs in environmental settlements such as complex wastewater samples.

3.6. Comparison with reference study

This work did not aim to compete with already commercial workflows such as the ProxiMeta platform. In general, the macroscopic results obtained with this method are similar to the ones obtained in Stälder et al. (2018) in terms of messages given. The major differences between these studies are the number of hits and links obtained. In this study, similar values for ARGs were obtained but much lower number of links were obtained on the mobilome analysis (plasmids and integrons). This lower number of links, even if giving similar information, lacks the clustering effect that they could get due to multiple species containing target genes from the same taxonomical group. In this study, same taxonomic units linked similar target genes but showing a lower clustering effect.

If we compare the number of clusters or MAGs that were selected after CheckM for their study and ours, the values ended up being similar: 51 vs. 55 MAGs in our study. This clarifies that the starting point for the Hi-C analysis was similar in both studies. Some differences could have arisen through the binning process. For the binning process, bin3C was utilized during this study, where as an input file, the metagenomic assembly, the Hi-C to the metagenomic assembly alignment and the enzyme utilized for digestion of Hi-C genetic material had to be provided. It is hypothesized that any relevant variation between both processes could be driven by the enzyme selected for binning. In our case, MluCI enzyme was used as default as the enzyme used in their study was not referred. Metagenomic assembly and Hi-C reads processing was done the same way so we do not consider these steps as the ones increasing the variability on the results. However, it is important to highlight that same software executed on the same data with different machines and operative systems might give

different results unless they are executed on Docker. Thus, this being a major concern in bioinformatics in terms of reproducibility (DI Tommaso et al., 2017). The binning algorithm is also different. Regarding taxonomy assignment (annotation), it is not believed that many variations was added due to the usage of the same software (Mash) for assigning it and obtaining of similar number of high-quality MAGs.

The bigger variation comes to the way the Hi-C reads were deconvoluted. They use the ProxiMeta platform while we tried to link directly contigs containing ARGs or MGEs to the MAGs they belonged and then, normalize by size of the MAGs (number of contigs it contained). During these steps, BLASTn was utilized to obtain the ARGs and MGEs hits. In their study, they used BLAT with the option `-minIdentity =90` and coverage higher than 80% while in this study BLASTn was utilized with the option `-evalue <10-20`. For detection improvement, best hits should be the ones with the highest bitscore. When contigs had multiple hits for the same locus, we selected the best hit based on the lowest e-value and not the multiplication between the coverage and identity, which was what they did. This could also give high variability in the results.

Regarding the target genes assignment, we see again a similar trend but with substantial differences, especially with the ARGs. We see that the same bacterial families are potential hosts for containing ARGs within their genome or integrated in episomal plasmids. However, when it comes to specific ARGs, we see differences. Significant differences are highlighted specially on highly-abundant ARGs such as sulfonamides (*su1*) and some aminoglycosides (*AANT3'*) resistance genes. On the other hand, ARGs such as rifampicin (*rpoB*) and fluoroquinolones (*GyrA*) resistance genes are not even mentioned on the selected paper and are highly abundant and present with this pipeline. Similar behavior is observed when plasmids and integrons are assessed. However, this last one could be due to a different version of the database used as the integrons database was locally built and the plasmid database from PlasmidFinder was the enterobacteria database one, which remains unclear if it is the same used in the reference study.

Overall, this pipeline replicates successfully what has been published in Stalder et al. (2019). Major drawbacks come on some specific ARGs and links richness. This could have happened

by having applied a more restrictive cutoff (removing all the MAGs for the visualization that contained less than 150 contigs). If we just generate the ARG figure without any cutoff, we can actually see genes that were not present in the ARG analysis such as *sul1*, *tetO* and *qnrS* (Figure 8) in the same bacterial families stated in the reference paper. Phylogenetic tree of non-cutoff ARG analysis can be found in Supplementary material (Figure S1).

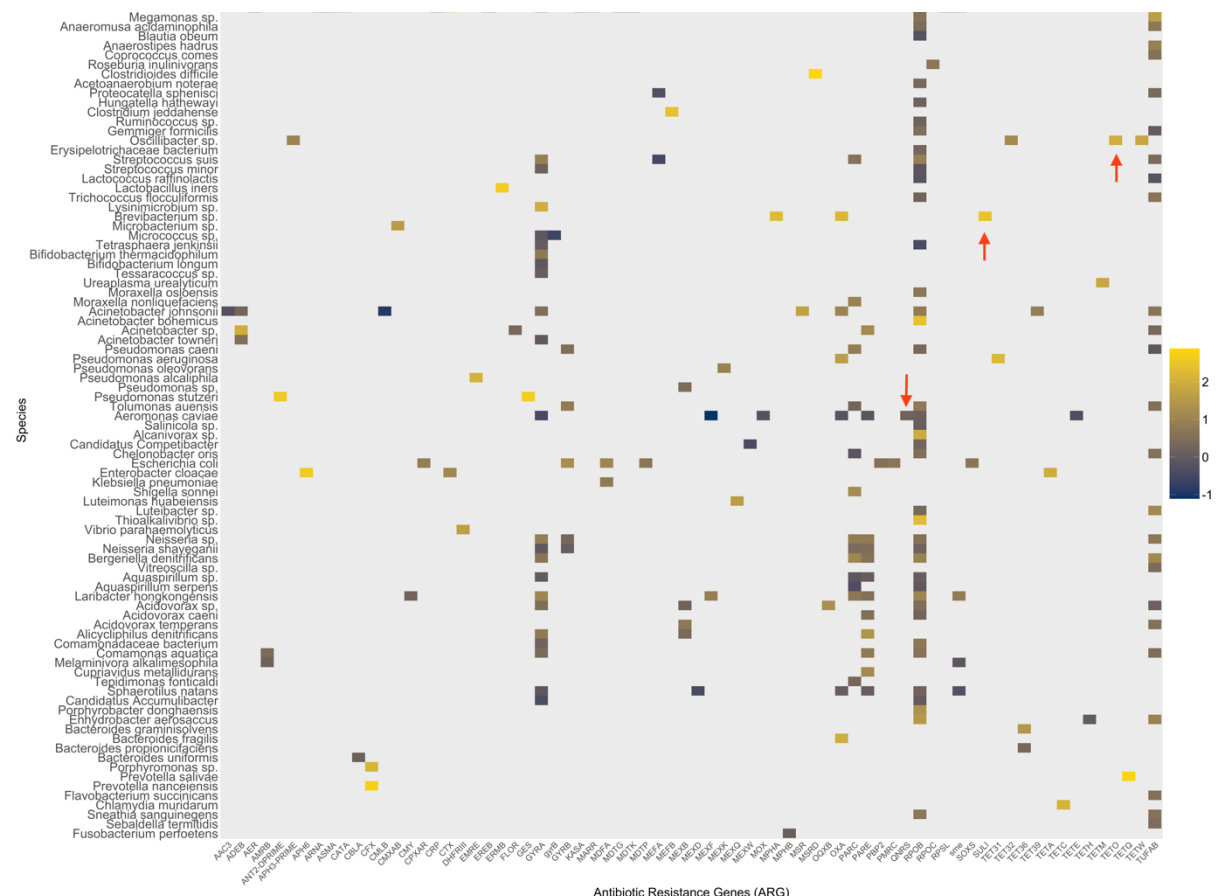


Figure 8. Hi-C links between the clusters and ARGs, identified in the wastewater sample and affiliated with most abundant bacterial families. Each edge of the phylogenetic tree represents a cluster. Only clusters having a contact with ARGs are shown. The presence or absence of a link is shown on the heatmap to the right of the tree, and the color shading represents the intensity of the normalized Hi-C link signals on logarithmic scale. **Note:** not cutoff was applied here. **Red arrows** showing significant ARG differences and the effect of applying visualization cutoffs.

4. Conclusions

- Hi-C linking suggests that *Aeromonadaceae*, *Moraxellaceae*, *Neisseriaceae*, *Chromobacteriaceae* and *Comamonadaceae* are the most likely reservoirs of ARGs in this wastewater samples.
- IncQ (1 and 2) plasmids had the broadest range of putative hosts, followed by the colE1-type plasmids.
- Class 1 integrons were the most represented, specifically in the Gamma-proteobacteria and less in the *Bacteroides*, *Actinobacteria* and *Chlamydiae* families.
- Hi-C results obtained with this pipeline does still follow the same trend as the reference study even if improvements must be performed in order to magnify the host:target gene resolution.

5. Glossary

ARG – Antibiotic Resistance Gene

BHR – Broad Host Range

MAG – Metagenome-Assembled Genome

MGE – Mobile Genetic Element

NHR – Narrow Host Range

6. References

- Andrews, S. (2010). FastQC: a quality control tool for high throughput sequence data. Available at: <http://www.bioinformatics.babraham.ac.uk/projects/fastqc>.
- Burton, J. N., Liachko, I., Dunham, M. J., and Shendure, J. (2014). Species-Level Deconvolution of Metagenome Assemblies with Hi-C–Based Contact Probability Maps. *G3 : Genes/Genomes/Genetics* 4, 1339–1346. doi:10.1534/g3.114.011825.
- Carattoli, A., Zankari, E., García-Fernández, A., Larsen, M. V., Lund, O., Villa, L., et al. (2014). In Silico detection and typing of plasmids using plasmidfinder and plasmid multilocus sequence typing. *Antimicrob. Agents Chemother.* 58, 3895–3903. doi:10.1128/AAC.02412-14.
- Chamberlain, S. A., and Szöcs, E. (2013). taxize: taxonomic search and retrieval in R. *F1000Research* 2, 191. doi:10.12688/f1000research.2-191.v1.
- Cydzik-Kwiatkowska, A., and Zielińska, M. (2016). Bacterial communities in full-scale wastewater treatment systems. *World J. Microbiol. Biotechnol.* 32, 1–8. doi:10.1007/s11274-016-2012-9.
- Demaere, M. Z., and Darling, A. E. (2019). Bin3C: Exploiting Hi-C sequencing data to accurately resolve metagenome-assembled genomes. *Genome Biol.* 20, 1–16. doi:10.1186/s13059-019-1643-1.
- DI Tommaso, P., Chatzou, M., Floden, E. W., Barja, P. P., Palumbo, E., and Notredame, C. (2017). Nextflow enables reproducible computational workflows. *Nat. Biotechnol.* 35, 316–319. doi:10.1038/nbt.3820.
- Doster, E., Lakin, S. M., Dean, C. J., Wolfe, C., Young, J. G., Boucher, C., et al. (2020). MEGARes 2.0: a database for classification of antimicrobial drug, biocide and metal resistance determinants in metagenomic sequence data. *Nucleic Acids Res.* 48, D561–D569. doi:10.1093/nar/gkz1010.
- Fraser, J., Williamson, I., Bickmore, W. A., and Dostie, J. (2015). An Overview of Genome Organization and How We Got There: from FISH to Hi-C. *Microbiol. Mol. Biol. Rev.* 79, 347–372. doi:10.1128/mmmbr.00006-15.
- Gerzova, L., Videnska, P., Faldynova, M., Sedlar, K., Provaznik, I., Cizek, A., et al. (2014). Characterization of microbiota composition and presence of selected antibiotic resistance genes in carriage water of ornamental fish. *PLoS One* 9. doi:10.1371/journal.pone.0103865.
- Gurevich, A., Saveliev, V., Vyahhi, N., and Tesler, G. (2013). QUAST: Quality assessment tool for genome assemblies. *Bioinformatics* 29, 1072–1075. doi:10.1093/bioinformatics/btt086.
- Li, D., Liu, C. M., Luo, R., Sadakane, K., and Lam, T. W. (2015). MEGAHIT: An ultra-fast single-node solution for large and complex metagenomics assembly via succinct de Bruijn graph. *Bioinformatics* 31, 1674–1676. doi:10.1093/bioinformatics/btv033.
- Li, H. (2013). Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM. 00, 1–3. Available at: <http://arxiv.org/abs/1303.3997>.
- Li, H., Handsaker, B., Wysoker, A., Fennell, T., Ruan, J., Homer, N., et al. (2009). The Sequence Alignment/Map format and SAMtools. *Bioinformatics* 25, 2078–2079. doi:10.1093/bioinformatics/btp352.
- Lieberman-Aiden, E., Van Berkum, N. L., Williams, L., Imakaev, M., Ragoczy, T., Telling, A., et al. (2009). Comprehensive mapping of long-range interactions reveals folding principles of the human genome. *Science (80-.).* 326, 289–293. doi:10.1126/science.1181369.

- Lindbäck, E., Rahman, M., Jalal, S., and Wretling, B. (2002). Mutations in *gyrA*, *gyrB*, *parC*, and *parE* in quinolone-resistant strains of *Neisseria gonorrhoeae*. *Apmis* 110, 651–657. doi:10.1034/j.1600-0463.2002.1100909.x.
- Ondov, B. D., Treangen, T. J., Melsted, P., Mallonee, A. B., Bergman, N. H., Koren, S., et al. (2016). Mash: Fast genome and metagenome distance estimation using MinHash. *Genome Biol.* 17, 1–14. doi:10.1186/s13059-016-0997-x.
- Orlando, G., Law, P. J., Cornish, A. J., Dobbins, S. E., Chubb, D., Broderick, P., et al. (2018). Promoter capture Hi-C-based identification of recurrent noncoding mutations in colorectal cancer. *Nat. Genet.* 50, 1375–1380. doi:10.1038/s41588-018-0211-z.
- Parks, D. H., Imelfort, M., Skennerton, C. T., Hugenholtz, P., and Tyson, G. W. (2015). CheckM: Assessing the quality of microbial genomes recovered from isolates, single cells, and metagenomes. *Genome Res.* 25, 1043–1055. doi:10.1101/gr.186072.114.
- R Foundation for Statistical Computing. (2018). *R: a Language and Environment for Statistical Computing*.
- Rocha, P. P., Raviram, R., Bonneau, R., and Skok, J. A. (2016). Breaking TADs: insights into hierarchical genome organization, Epigenomics, Future Medicine. 7, 523–526. doi:10.2217/epi.15.25.Breaking.
- Scott, A., Szoecs, E., Foster, Z., Baumgartner, J., and Donnell, J. O. (2020). R Package ‘taxize’.
- Sedlar, K., Kupkova, K., and Provaznik, I. (2017). Bioinformatics strategies for taxonomy independent binning and visualization of sequences in shotgun metagenomics. *Comput. Struct. Biotechnol. J.* 15, 48–55. doi:10.1016/j.csbj.2016.11.005.
- Skoczynska, A., Ruckly, C., Hong, E., and Taha, M. K. (2009). Molecular characterization of resistance to rifampicin in clinical isolates of *Neisseria meningitidis*. *Clin. Microbiol. Infect.* 15, 1178–1181. doi:10.1111/j.1469-0691.2009.02783.x.
- Stalder, T., Press, M. O., Sullivan, S., Liachko, I., and Top, E. M. (2019). Linking the resistome and plasmidome to the microbiome. *ISME J.*, 2437–2446. doi:10.1038/s41396-019-0446-4.
- van Berkum, N. L., Lieberman-Aiden, E., Williams, L., Imakaev, M., Gnirke, A., Mirny, L. A., et al. (2010). Hi-C: A method to study the three-dimensional architecture of genomes. *J. Vis. Exp.*, 1–7. doi:10.3791/1869.
- Van Loosdrecht, M. C. M., Brandse, F. A., and De Vries, A. C. (1998). Upgrading of waste water treatment processes for integrated nutrient removal the BCFS® process. *Water Sci. Technol.* 37, 209–217. doi:10.1016/S0273-1223(98)00290-X.
- Wickham, H. (2016). *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York Available at: <https://ggplot2.tidyverse.org>.
- Xie, R., Zhang, X. D., Zhao, Q., Peng, B., and Zheng, J. (2018). Analysis of global prevalence of antibiotic resistance in *Acinetobacter baumannii* infections disclosed a faster increase in OECD countries. *Emerg. Microbes Infect.* 7. doi:10.1038/s41426-018-0038-9.
- Yu, G. (2020). Using ggtree to Visualize Data on Tree-Like Structures. *Curr. Protoc. Bioinforma.* 69, 1–18. doi:10.1002/cpbi.96.

7. Acknowledgements

This project has been done while pursuing my PhD on the Environmental Biotechnology section at Delft University of Technology, The Netherlands. I would strongly like to acknowledge my Master Thesis tutor, Andreu Paytuví for having helped and guided me so much during the developing of this project. To my friends Diego Alonso and Fernando Naya for their valuable inputs and hints. To my quarantine team for listening to me complaining while going through some really tough moments. Also, to my old friends from Martorell, my colleagues from the Biotechnology Bachelor at Autònoma de Barcelona and the Delft cool kids. They always make me smile. Finally, to my parents, Olga and Ramón, because they have always trusted on me without hesitation and for understanding my needs to fly a little bit higher even if it is far away from them.

Thank you very much.

8. Supplementary material

The code

```
#!/bin/bash

set -e

#####
# Variables #
#####

MASH_PATH="./mash"
BIN3C_PATH="./bin3C/bin3C.py"

#####
# PARSE ARGUMENTS #
#####

if [[ ! $(type bwa 2> /dev/null) ]]; then echo "bwa program is not in the \${PATH}"; fi

if [[ ! $(type blastn 2> /dev/null) ]]; then echo "blastn program is not in the \${PATH}"; fi

if [[ ! $(type samtools 2> /dev/null) ]]; then echo "samtools program is not in the \${PATH}"; fi

if [[ ! $(type checkm 2> /dev/null) ]]; then echo "checkm program is not in the \${PATH}"; fi

if [[ -z $1 ]]; then
    echo "first argument should be an ASSEMBLY in fasta"
    exit 1
fi

if [[ -z $2 ]]; then
    echo "second argument should be Hi-C PairedEnd file 1"
    exit 1 #0 means script was successful, 1 means script failed
fi

if [[ -z $3 ]]; then
    echo "third argument should be Hi-C PairedEnd file 2"
    exit 1 #0 means script was successful, 1 means script failed
fi

if [[ -z $4 ]]; then
    echo "database for ARGs"
    exit 1
fi
```

```
if [[ -z $5 ]]; then
echo "database for Plasmids"
exit 1
fi
```

```
if [[ -z $6 ]]; then
echo "database for Integrans"
exit 1
fi
```

```
if [[ -z $7 ]]; then
echo "give me a number!"
exit 1 #0 means script was successful, 1 means script failed
fi
```

Align Hi-C paired-end library to the MEGAHIT assembly using BWA mem

```
if [[ $start < 2 ]]; then
# STEP 1: Generate index file from assembly
bwa index $1
fi
```

```
if [[ $start < 3 ]]; then
# STEP 2: align Hi-C to assembly to generate the .sam file
bwa mem -5SP $1 $2 $3 > hic_mem.sam
fi
```

```
if [[ $start < 4 ]]; then
# STEP 3: Generate the .bam file
samtools view -S -h -b -F 2316 hic_mem.sam > hic_mem.bam
fi
```

```
if [[ $start < 5 ]]; then
# STEP 4: Sort by name the bam file
samtools sort -n hic_mem.bam > hic_mem_name.bam
fi
```

Bin our assembly with the aligned Hi-C reads using Bin3C

```
if [[ $start < 6 ]]; then
# STEP5: Create a contact map for analysis
python2 $BIN3C_PATH mkmap -e MluCI -v $1 hic_mem_name.bam bin3c_out
fi
```

```
if [[ $start < 7 ]]; then
```

```
# STEP6: Cluster the resulting contact map into genome bins. Only in ubuntu environments  
with more than 32GB ram  
python2 $BIN3C_PATH cluster -v bin3c_out/contact_map.p.gz bin3c_clust  
fi
```

Binning Quality Control with CheckM

```
if [[ $start < 8 ]]; then  
# STEP 7: Generate a folder with the quality control about the binning process. It can be  
used for "removing" low quality bins from folder and start over STEP 8  
checkm lineage_wf -x fna bin3c_clust/fastq fasta_bins_out  
fi
```

Assign taxonomy using Mash

```
if [[ $start < 9 ]]; then  
# STEP 8: Copy all fasta files in current folder  
cp bin3c_clust/fastq/* .  
fi  
  
if [[ $start < 10 ]]; then  
# STEP 9: Generate a file with hits  
for i in `ls CL*.fna`; do $MASH_PATH screen -w -p 4 refseq.genomes.k21s1000.msh $i >  
$i.tab; done  
fi  
  
if [[ $start < 11 ]]; then  
# STEP 10: Generate a folder with the tab_sorted mash files  
mkdir tab_sorted  
fi  
  
if [[ $start < 12 ]]; then  
# STEP 11: Sort mash results to have on top the best hit  
for i in `ls CL*.fna.tab`; do sort -gr $i > tab_sorted/$i.tab; done  
fi  
  
if [[ $start < 13 ]]; then  
# STEP 12: Copy all sorted files to current folder  
cp tab_sorted/* .  
fi  
  
if [[ $start < 14 ]]; then  
# STEP 13: Extract mash sorted hits on a table  
for i in `ls CL*.fna.tab.tab`; do head -n 1 $i> $i.tab ; done  
fi  
  
if [[ $start < 15 ]]; then
```


STEP 14: Extract mash sorted hits on a table.

bash concat.sh

fi

```
###concat.sh code
#!/bin/bash

out=merged_tophits_mash.tab
# delete the file prior to doing concatenation
# or if ran twice it would be counted in the input files!
rm -f "$out"

for f in *.fna.tab.tab.tab
do
    if [ -s "$f" ] ; then
        #cat "$f" | sed 's/^/$f,/'
        sed "s/^/$f,/" "$f"

    else
        echo "$f,"
    fi
done > $out
```

if [[\$start < 16]]; then

STEP 15: Extract names of fasta files (CL....fna) This is ok.

ls bin3c_clust/fastafasta/ | sed 's/.fna//g' > names.txt

fi

if [[\$start < 17]]; then

STEP 16: Merge both names file and merged_tophits_mash.tab file in one. Then we select only the columns with the CL... code and the name of the species to have a clean mash results file.

paste -d ' ' names.txt merged_tophits_mash.tab > MASH_results.txt

fi

Generate ARGs/Integrans/Plasmids hits on our hits

if [[\$start < 18]]; then

STEP 17: Merge all the bins before generate in a fasta file

cat bin3c_clust/fastafasta/* > merged_fasta.fna

fi

if [[\$start < 19]]; then

STEP 18: Index file the databases that are going to be used for getting the hits

makeblastdb -in \$4 -dbtype nucl

makeblastdb -in \$5 -dbtype nucl

```
makeblastdb -in $6 -dbtype nucl  
fi
```

```
if [[ $start < 20 ]]; then  
# STEP 19: Perform the blastn with the merged fasta file and the indexed databases with e-  
value  $10^{-20}$ .
```

```
# ARGs  
blastn -db $4 -query merged_fasta.fna -evalue 1e-20 -outfmt 7 -out ARG_results.txt  
cat ARG_results.txt | awk '/hits found/{getline;print}' | grep -v "#" | awk '{print $1,$2}' >  
ARG_top_hits.txt
```

```
# Plasmids  
blastn -db $5 -query merged_fasta.fna -evalue 1e-20 -outfmt 7 -out plasmid_results.txt  
cat plasmid_results.txt | awk '/hits found/{getline;print}' | grep -v "#" | awk '{print $1,$2}' >  
plasmid_top_hits.txt
```

```
# Integrases  
blastn -db $6 -query merged_fasta.fna -evalue 1e-20 -outfmt 7 -out integrases_results.txt  
cat integrases_results.txt | awk '/hits found/{getline;print}' | grep -v "#" | awk '{print $1,$2}'  
> integrases_top_hits.txt  
fi
```

Links between aligned contigs between the assembly and the Hi-C reads

```
if [[ $start < 21 ]]; then  
# STEP 20: Generate alignment to links file necessary for STEP 21 scripts.  
samtools view hic_mem_name.bam | awk '{hash[$3"\t"$7]++}END{for (x in hash) {print  
x"\t"hash[x]/2}}' > contig_links.txt  
fi
```

Sum of interactions between ARGs/Integrases/Plasmids and its respective clusters

```
if [[ $start < 22 ]]; then  
# STEP 21: Generate sum of interaction files  
python2 script_ARG_d.py  
python2 script_plasmid_d.py  
python2 script_integrases_d.py  
fi
```

The python scripts are shown in the following code: The idea is to take all the hits from the blast results, extract hit per hit the cluster and contig that it belongs. Then, once we have that number, we extract only the contig and from the contig_links.txt file, it sums all the contig interactions between the blast hit contig and all the other contigs belonging to the same cluster.

```
# -*- coding: utf-8 -*-
"""
Created on Thu Apr 23 17:23:43 2020
Script_ARG_d.py
"""

# Load the Pandas libraries with alias 'pd'
import pandas as pd
import glob

path = r"./bin3c_clust/fasta" # use your path # use your path
all_files = glob.glob(path + "/*.fna")
#
pd_list = [] #una lista de ficheros donde voy a guardar los ficheros filtrados. Se genera vacia al principio.
for f in all_files: #all_files es la lista y la f es cada uno de los elementos
    print(f)
    doc2_temp = pd.read_csv(f, sep='\s+', header=None) #leo el fichero 1
    #Doc2 - Filter Rows with its first column not ">.*"
    doc2_temp = doc2_temp[doc2_temp[0].str.contains(">")].ix[:, [0, 1]] #tratamiento que
    #haciamos antes: filtrar del mayor que y quedarse solo con la columna 0 y 1.
    #Doc2 - Filter First and Second Column
    doc2_temp.ix[:, [0, 1]]
    pd_list.append(doc2_temp)

doc2 = pd.concat(pd_list, axis=0, ignore_index=True) #coge la lista y lo concatena, para tener
1 solo dataframe en lugar de una lista de data.frames. El axis 0 es que lo juntas por el eje
0, que me ponga uno debajo del otro y au. Esto me genera un fichero/documento.
# Read data from file 'filename.csv'
# (in the same directory that your python process is based)
# Control delimiters, rows, column names with read_csv (see later)
doc1 = pd.read_csv("ARG_top_hits.txt", sep='\s+', header=None)
doc2 = pd.read_csv("CL0003.txt", sep='\s+', header=None)
doc3 = pd.read_csv("contig_links.txt", sep='\s+', header=None)

# Data cleaning
#Doc1 - Keep column 1 delete others
doc1 = doc1.ix[:, [0]]
#Doc 3 -Delete rows with equal sign in any column
doc3 = doc3[~doc3[0].str.contains("=")]
doc3 = doc3[~doc3[1].str.contains("=")]
#if k99 in doc3 is not in doc2 - delete interaction

# Data preparation
doc2[0] = doc2[0].str.replace(">", "")
doc2[1] = doc2[1].str.replace("contig:", "")
k99_list=doc2[1].drop_duplicates().values.tolist()
doc3 = doc3[doc3[0].isin(k99_list)]
doc3 = doc3[doc3[1].isin(k99_list)]

#Join doc1 % doc2
doc12=doc1.merge(doc2, on=[0], how='left')

#Join doc12 % doc3
doc12_doc3_c1=doc12.merge(doc3, left_on=[1], right_on=[0], how='left')
doc12_doc3_c1=doc12_doc3_c1.ix[:, ['0_x', '1_x', '1_y', 2]]
doc12_doc3_c2=doc12.merge(doc3, left_on=[1], right_on=[1], how='left')

#Rename columns
header_list = ["CL-doc1", "k99-doc2", "k99-interac", "value"]
doc12_doc3_c1.columns = header_list
doc12_doc3_c2.columns = header_list

#Append
resultado=doc12_doc3_c1.append(doc12_doc3_c2).sort_values(by=['CL-doc1'])
```

```
#Write Result
resultado.to_csv("result_ARG.txt", index=False)

#GroupBy["CL-doc1", "k99-doc2"] and sum
resultado_suma=resultado.groupby(["CL-doc1", "k99-doc2"]).sum()
resultado_suma.to_csv("result-sum_ARG.txt")

#Calculate normalization file by number of contigs per cluster
doc1_norm = pd.read_csv("ARG_top_hits.txt", sep='\s+', header=None)
doc1_norm = doc1_norm.ix[:, [0]]
doc1_norm = doc1_norm[0].str.slice(0,
6).to_frame().drop_duplicates(keep='first').dropna().reset_index(drop=True)
doc2_filter= doc2[0].str.slice(0, 6).to_frame()
doc2_filter[1]= doc2_filter[0]
doc2_filter=doc2_filter.groupby([1], as_index=False).count()
join_norm=doc1_norm.merge(doc2_filter, left_on=[0], right_on=[1],
how='left').ix[:, ['key_0', '0_y']].fillna(0)
header_list = ["CLXXX", "count-doc2"]
join_norm.columns = header_list
join_norm["count-doc2"]=join_norm["count-doc2"].astype(int)
join_norm.to_csv("result_CLXXX0currences_ARG.txt", index=False)
```

It generates 3 files: result_ARG (plasmid or integrase).txt, result-sum_ARG.txt (this is the one that we will use for generating the heatmap) and result_CLXXX0currences_ARG.txt, used for normalization.

###Prepare files for heatmap generation

if [[\$start < 23]]; then

STEP 22: Clean files and join sum of interactions with the events

sed 's/,/ /g' result-sum_ARG.txt | tail -n +2 > result-sum_ARG_clean.txt

sed 's/,/ /g' result-sum_plasmid.txt | tail -n +2 > result-sum_plasmid_clean.txt

sed 's/,/ /g' result-sum_integrases.txt | tail -n +2 > result-sum_integrases_clean.txt

#First we remove spaces from script results to generate columns, then we remove header and then we join with top hits of the specific event.

fi

if [[\$start < 24]]; then

STEP 23: Join them with top hits from blast results

join -1 1 -2 1 ARG_top_hits.txt result-sum_ARG_clean.txt > ARG_hits_suminteractions.txt

join -1 1 -2 1 plasmid_top_hits.txt result-sum_plasmid_clean.txt >

plasmid_hits_suminteractions.txt

join -1 1 -2 1 integrases_top_hits.txt result-sum_integrases_clean.txt >

integrases_hits_suminteractions.txt

fi

```
if [[ $start < 25 ]]; then
# STEP 24: Remove "_" from ARG_hits_suminteractions so the species from MASH can be
added and join them
awk '{gsub("_","",$1)}1' ARG_hits_suminteractions.txt >
ARG_hits_suminteractions_clean.txt
join -1 1 -2 1 ARG_hits_suminteractions_clean.txt MASH_results.txt >
ARG_final_results_R.txt
awk '{gsub("_","",$1)}1' plasmid_hits_suminteractions.txt >
plasmid_hits_suminteractions_clean.txt
join -1 1 -2 1 plasmid_hits_suminteractions_clean.txt MASH_results.txt >
plasmid_final_results_R.txt
awk '{gsub("_","",$1)}1' integrases_hits_suminteractions.txt >
integrases_hits_suminteractions_clean.txt
join -1 1 -2 1 integrases_hits_suminteractions_clean.txt MASH_results.txt >
integrases_final_results_R.txt
fi

if [[ $start < 26 ]]; then
# STEP 24: Clean file part I
awk '{print $1,$3,$5,$14,$15}' ARG_final_results_R.txt > ARG_final_results_R_clean.txt
awk '{print $1,$3,$5,$14,$15}' plasmid_final_results_R.txt >
plasmid_final_results_R_clean.txt
awk '{print $1,$3,$5,$14,$15}' integrases_final_results_R.txt >
integrases_final_results_R_clean.txt
fi

if [[ $start < 27 ]]; then
# STEP 25: Clean file part II: subtract column by column and generate the final file
awk -F"fna.gz" '{print $2}' ARG_final_results_R.txt | awk -F"." '{print $2}' | awk -F" " '{print
$2" "$3}' > ARG_species.txt
awk -F"fna.gz" '{print $2}' plasmid_final_results_R.txt | awk -F"." '{print $2}' | awk -F" "
'{print $2" "$3}' > plasmid_species.txt
awk -F"fna.gz" '{print $2}' integrases_final_results_R.txt | awk -F"." '{print $2}' | awk -F" "
'{print $2" "$3}' > integrases_species.txt
awk '{print $1}' ARG_final_results_R_clean.txt > ARG_cluster.txt
awk '{print $1}' plasmid_final_results_R_clean.txt > plasmid_cluster.txt
awk '{print $1}' integrases_final_results_R_clean.txt > integrases_cluster.txt
grep "CL" ARG_final_results_R.txt | awk 'BEGIN{FS="|"}{split($0,a,"|"); if
(a[length(a)]~"Requires") {print a[length(a)-3]} else {print a[length(a)-2]}}' > ARG_event.txt
awk '{print $2}' ARG_final_results_R_clean.txt | awk 'BEGIN{FS="|"}{split($0,a,"|"); if
(a[length(a)]~"Requires") {print a[length(a)-1]} else {print a[length(a)-0]}}' > ARG_gene.txt
awk '{print $2}' plasmid_final_results_R_clean.txt | sed 's/_.*//' > plasmid_event.txt
awk '{print $2}' integrases_final_results_R_clean.txt > integrases_event.txt
awk '{print $3}' ARG_final_results_R_clean.txt > ARG_sumint.txt
awk '{print $3}' plasmid_final_results_R_clean.txt > plasmid_sumint.txt
awk '{print $3}' integrases_final_results_R_clean.txt > integrases_sumint.txt
fi
```

Sum of interactions normalization by amount of contigs in cluster

```
if [[ $start < 28 ]]; then
```

```
# STEP 26: Clean file part IV: subtract number of contigs per cluster and normalize sum of interactions
```

```
sed 's/,/ /g' result_CLXXXXOcurrrences_ARG.txt | tail -n +2 > Ocurrrences_ARG.txt  
join -1 1 -2 1 ARG_cluster.txt Ocurrrences_ARG.txt | awk '{print $2}' >  
Ocurrrences_ARG_clean.txt  
paste ARG_sumint.txt Ocurrrences_ARG_clean.txt | awk '{print($1/$2)}' >  
normalizedsum_ARG.txt
```

```
sed 's/,/ /g' result_CLXXXXOcurrrences_plasmid.txt | tail -n +2 > Ocurrrences_plasmid.txt  
join -1 1 -2 1 plasmid_cluster.txt Ocurrrences_plasmid.txt | awk '{print $2}' >  
Ocurrrences_plasmid_clean.txt  
paste plasmid_sumint.txt Ocurrrences_plasmid_clean.txt | awk '{print($1/$2)}' >  
normalizedsum_plasmid.txt
```

```
sed 's/,/ /g' result_CLXXXXOcurrrences_integrases.txt | tail -n +2 > Ocurrrences_integrases.txt  
join -1 1 -2 1 integrases_cluster.txt Ocurrrences_integrases.txt | awk '{print $2}' >  
Ocurrrences_integrases_clean.txt  
paste integrases_sumint.txt Ocurrrences_integrases_clean.txt | awk '{print($1/$2)}' >  
normalizedsum_integrases.txt
```

```
fi
```

```
if [[ $start < 29 ]]; then
```

```
# STEP 27: Clean file part III: join the parts and sort them by second column (Species)  
paste ARG_cluster.txt ARG_species.txt ARG_event.txt ARG_gene.txt ARG_sumint.txt  
Ocurrrences_ARG_clean.txt normalizedsum_ARG.txt | sort -k 2 > ARG_R.txt
```

```
paste plasmid_cluster.txt plasmid_species.txt plasmid_event.txt plasmid_sumint.txt  
Ocurrrences_plasmid_clean.txt normalizedsum_plasmid.txt | sort -k 2 > plasmid_R.txt  
paste integrases_cluster.txt integrases_species.txt integrases_event.txt  
integrases_sumint.txt Ocurrrences_integrases_clean.txt normalizedsum_integrases.txt | sort  
-k 2 > integrases_R.txt
```

```
fi
```

```
if [[ $start < 30 ]]; then
```

```
# STEP 28: Add header to the file
```

```
echo -e 'CLUSTER\tSPECIES\tARG family\tARG\tSUM\tNumber of Contigs\tNormalizedSum'|  
cat - ARG_R.txt > ARG_R_clean.csv  
echo -e 'CLUSTER\tSPECIES\tEVENT\tSUM\tNumber of Contigs\tNormalizedSum'| cat -  
plasmid_R.txt > plasmid_R_clean.csv  
echo -e 'CLUSTER\tSPECIES\tEVENT\tSUM\tNumber of Contigs\tNormalizedSum'| cat -  
integrases_R.txt > integrases_R_clean.csv
```

```
fi
```

```
if [[ $start < 31 ]]; then
# STEP 28: Run R
Rscript hic.r
fi
```

```
#!/usr/bin/env Rscript #This is the script for hic.r analysis
library(ggplot2)

#ARG_genes

dataARG<-read.csv("./ARG_R_clean.csv", header = TRUE, sep="\t")

LOG.normalizedabun<-log10(dataARG$NormalizedSum)
mine.heatmap<-ggplot(data=dataARG, mapping = aes(x=ARG,
                                                  y=SPECIES,
                                                  fill=LOG.normalizedabun)) +

  geom_tile() +
  scale_fill_gradient(name = "Log(Normalized sum of interactions)",
                     low = "#003366",
                     high = "#EFEDC2") +

  theme(
    panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(),
    axis.line = element_blank(),
    axis.ticks = element_blank(),
    axis.text.x = element_text(angle=45, size=20, hjust=1),
    axis.text.y = element_text(size=20, hjust=1)
  )

ggsave("heatmapARG_gene.tiff", units="in", width=30, height=30, dpi=300, compression
= 'lzw')

#ARG_family

mine.heatmap<-ggplot(data=dataARG, mapping = aes(x=ARG.family,
                                                  y=SPECIES,

                                                  fill=LOG.normalizedabun)) +
  geom_tile() +
  scale_fill_gradient(name = "Log(Normalized sum of interactions)",
                     low = "#003366",
                     high = "#EFEDC2") +

  theme(
    panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(),
    axis.line = element_blank(),
    axis.ticks = element_blank(),
    axis.text.x = element_text(angle=45, size=20, hjust=1),
    axis.text.y = element_text(size=20, hjust=1)
  )

ggsave("heatmapARG_family.tiff", units="in", width=30, height=30, dpi=300,
compression = 'lzw')
```

This R script will generate the heatmaps for ARGs, plasmids and integrases. For results interpretation, heatmap had to be linked to phylogenetic families and classes. For doing that, a R script was written.

```
#!/usr/bin/env Rscript #This is the script for Heatmap_phylogeny_R_code analysis

library(ggplot2)
library(taxize)
library(dplyr)

# Documento del script ARG

dataARG<-read.csv("./ARG_R_clean.csv", header = TRUE, sep="\t")
dataARG<-subset(dataARG,Number.of.Contigs>=100) #Modify, depends on case.

LOG.normalizedabun<-log10(dataARG$NormalizedSum)
mine.heatmap<-ggplot(data=dataARG, mapping = aes(x=ARG,
                                                  y=SPECIES,
                                                  fill=LOG.normalizedabun)) +
  geom_tile() +
  scale_fill_gradient(name = "Log(Normalized sum of interactions)",
                     low = "#003366",
                     high = "#EFEDC2") +
  theme(
    panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(),
    axis.line = element_blank(),
    axis.ticks = element_blank(),
    axis.text.x = element_text(angle=45, size=20, hjust=1),
    axis.text.y = element_text(size=20, hjust=1)
  )

mine.heatmap

ggsave("heatmapARGtest.tiff", units="in", width=30, height=30, dpi=300, compression =
'lwz')

#Taxize was used for retrieve the species phylum and genus names from NCBI

names<-as.character(dataARG$SPECIES)
class(names)
test<-tax_name(query=c(names), get= c("phylum","genus"), db="ncbi")

#Subsetting genus and phylum to be added to the original output file

ARG_genus<-test$genus
ARG_phylum<-test$phylum
dataARG1<-cbind(dataARG, ARG_genus, ARG_phylum)
dataARG1
```



```
LOG.normalizedabun<-log10(dataARG1$NormalizedSum)
mine.heatmap<-ggplot(data=dataARG1, mapping = aes(x=ARG,
                                                    y=SPECIES, #This can be modified by
                                                    fill=LOG.normalizedabun)) +
  changin SPECIES to Genus or Phylum depending on the level of detail needed.
  geom_tile() +
  scale_fill_gradient(name = "Log(Normalized sum of interactions)",
                      low = "#003366",
                      high = "#EFEDC2") +
  theme(
    panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(),
    axis.line = element_blank(),
    axis.ticks = element_blank(),
    axis.text.x = element_text(angle=45, size=20, hjust=1),
    axis.text.y = element_text(size=20, hjust=1)
  )

mine.heatmap

ggsave("heatmapARG1test.tiff", units="in", width=30, height=30, dpi=300, compression
= 'lzw')

#Remove duplicates for tree

library(dplyr)
names<-as.data.frame(dataARG1$SPECIES)
names<-distinct(names)

names<-as.character(names[,1])

##Generate the classification vector

tree_ARG<-classification(names, db="ncbi")

## Generat tree

tree_ARG_real<-class2tree(tree_ARG, check=TRUE, varstep = TRUE)

tiff(file="tree_ARG.tiff", width=15, height=30, units="in", res=300)

plot(tree_ARG_real)

dev.off()

tree_ARG_real$phylo$tip.label

## Obtain a vector with the species order of appearance in the tree

order_ARG<-
rev(tree_ARG_real$phylo$tip.label[tree_ARG_real$phylo$edge[tree_ARG_real$phylo$edge[,
2] <= 91,2]]) #This 91 has to be adjusted depending on the number of entries. Case-
dependent. Manually.
order_ARG

df<-data.frame(dataARG1)
df_corrected<-gsub("\\b[sp]{1,2}\\b", "sp.", df$SPECIES) #This will add a point after
every sp
dataARG2<-cbind(df,df_corrected)
dataARG2
```

```
# Reverse order to fit the way the tree is generated
level_order<-rev(order_ARG)

#Representation of the heatmap in order of appearance for linking it to the
phylogenetic tree

LOG.normalizedabun<-log10(dataARG2$NormalizedSum)
mine.heatmap<-ggplot(data=dataARG2, mapping = aes(x=ARG,
                                                    y=factor(df_corrected,
                                                    level=level_order),
                                                    fill=LOG.normalizedabun)) +
  geom_tile() +
  scale_fill_gradient(name = "",
                      low = "#003366",
                      high = "#FFD700",
                      ) +
  labs(x="Antibiotic Resistance Genes (ARG)", y="Species") +
  theme(
    panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(),
    axis.line = element_blank(),
    axis.ticks = element_blank(),
    axis.text.x = element_text(angle=45, size=20, hjust=1),
    axis.text.y = element_text(size=30, hjust=1),
    legend.key.size = unit(1,"in"),
    legend.text = element_text(size=30),
    axis.title = element_text(size=30)
  )
mine.heatmap
ggsave("heatmapARG_ordeded.tiff", units="in", width=40, height=30, dpi=300,
compression = 'lzw')
```

The R code before detailed is the one used for ARGs. Same workflow was followed for plasmids and integrases details. Figures obtained with this code: **tree_ARG.tiff (Figure S1)** and **heatmapARG_ordered.tiff** are further arranged by Microsoft PowerPoint in order to generate the figures of this report.

Phylogenetic tree

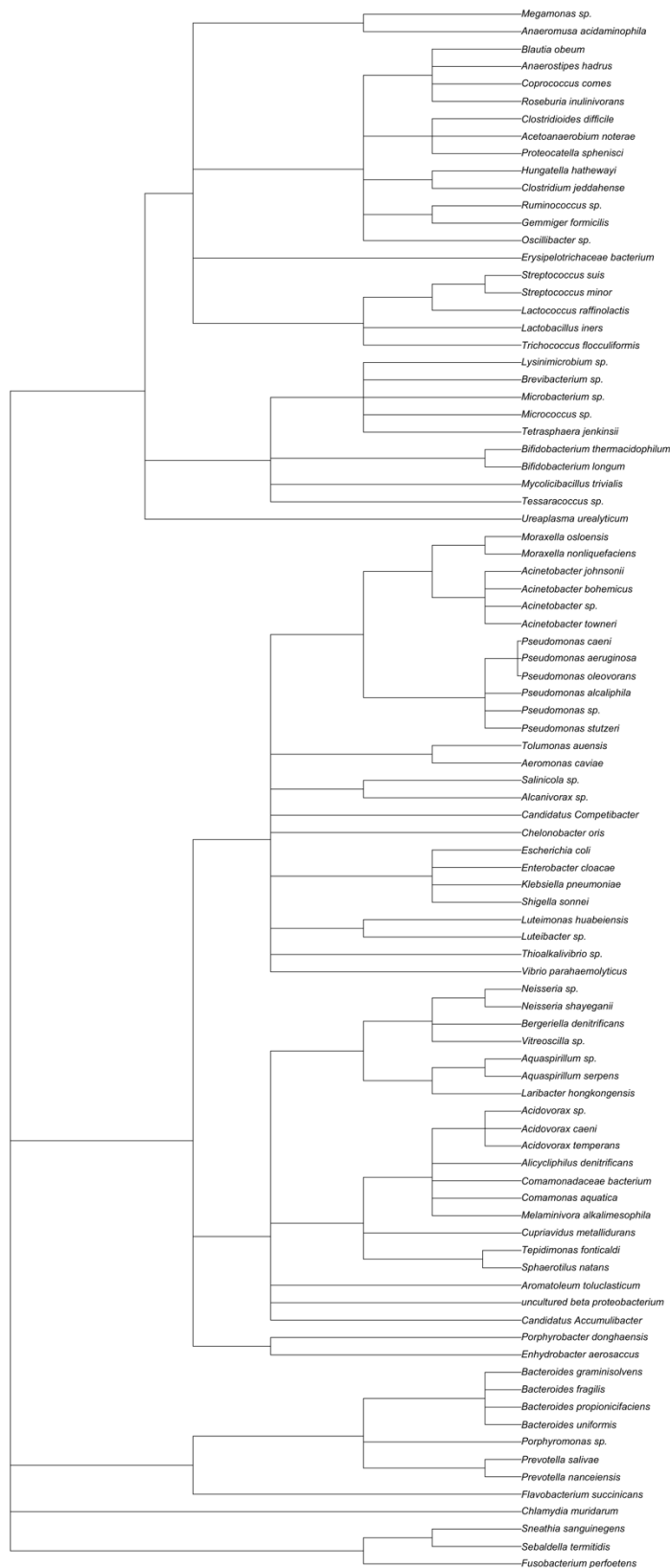


Figure S1. Phylogenetic tree from the cluster-links obtained for the ARGs analysis. All the clusters/MAGs here represented had a link or more with specific ARG target genes.