A method for detecting horizontal gene transfer events of antibiotic resistance genes using phylogenetic trees

Master's thesis in Engineering Mathematics and Computational Science

LEO BENSON

Department of Mathematical Sciences

Chalmers University of Technology
Gothenburg, Sweden 2025
www.chalmers.se

Master's thesis 2025

A method for detecting horizontal gene transfer events of antibiotic resistance genes using phylogenetic trees

LEO BENSON

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Mathematical Sciences
Systems Biology and Bioinformatics
Chalmers University of Technology
Gothenburg, Sweden 2025

A method for detecting horizontal gene transfer events of antibiotic resistance genes using phylogenetic trees
LEO BENSON

Supervisors: Erik Kristiansson, Department of Mathematical Sciences, Chalmers University
Fanny Berglund, Department of Infectious Diseases, Institute of Biomedicine, Gothenburg University
David Lund, Department of Mathematical Sciences, Chalmers University

Examiner: Erik Kristiansson, Department of Mathematical Sciences, Chalmers University

Master's Thesis 2025
Department of Mathematical Sciences
Systems Biology and Bioinformatics
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Egon Schiele, Four Trees, Vienna 1917.

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2025

A method for detecting horizontal gene transfer events of
antibiotic resistance genes using phylogenetic trees
Leo Benson
Department of Mathematical Sciences
Chalmers University of Technology

Abstract

An algorithm, named the HGT-score algorithm, to computationally assess horizontal gene transfer of antibiotic resistance genes spread was developed using a methodology based on phylogenetic trees and host taxonomy labels. Applying the HGT-score algorithm on validation data showed that the algorithm detects horizontal gene transfer events. This work serves as a good starting point for further investigation and improvements.

The results are two-fold, consisting of the developed methods themselves, and the results that the methods generated when applied on real data. Antibiotic resistance gene (ARG)-based phylogenetic trees were cross-sectioned at different lengths from the root, named cutoffs. A cutoff generates flat clusters of leaves, encoding ARG similarity between hosts within each cluster. Distantly related host bacteria may end up in the same cluster if their genomes contain identical or similar antibiotic resistance genes. HGT-scores were given to clusters depending on the taxonomic similarity of member hosts, with higher dissimilarity resulting in a higher HGT-score.

The method was validated on a labelled dataset of ARGs, where the labels sorted ARGs into HGT-prone genes and not non HGT-prone genes. The HGT-prone data scored higher than non-HGT labelled data, particularly in specific cutoff regions. It is therefore plausible that the method scores HGT events higher, and that the cutoff is an important factor to account for in drawing conclusions as to when the events ocurred.

Lastly, there are several directions to explore in order to improve the method. The HGT-score method is dependent on three key parameters: cutoff, cluster-level statistic, and p. These parameters should all be investigated and calibrated for different use case scenarios. A variant of the method should be investigated, using well-conserved genes rather than taxonomy as the base of gene tree-host species dissimilarity. This open up for the possibility of detecting HGT events across more closely related bacteria. Another interesting direction is fine-tuning cutoffs. Finer cutoffs could provide better resolution in critical parts of the phylogenetic gene trees where many bifurcations occur.

Keywords: antibacterial resistance, horizontal gene transfer, phylogenetic trees, bioinformatics.

Acknowledgements

I wish to express my gratitude to my examiner and supervisor Erik Kristiansson who has been insightful during consultations and patient during delays. I'm also particularly grateful for his inclusion of me in the team during the time of my thesis, and his keen interest in my ideas and work. Thank you to examiners David Lund who engaged with my work and gave helpful advice, and Fanny Berglund who also created fARGene whose output constituted the bulk of the data in this project. I finally want to thank my family, my friends, and my sambo, who have been and continue to be my biggest motivation!

Leo Benson, Gothenburg, June 2025

Glossary

Below is a glossary of terms and abbreviations that have been used throughout this thesis listed in alphabetical order:

Lowest common ancestor (LCA)	The ancestor common to all organisms in a population which is also closest to those organisms in evolutionary time.
p%-common ancestor (p%-CA)	The ancestor common to p% of all organisms in a population which is also closest to those organisms in evolutionary time.
Taxonomy	Scheme of classification of organisms in species, genus, family, order, class, phylum, and kingdom.
Taxonomic level (or rank)	A specific level in taxonomy. The first taxonomic level is species.
Taxon	A taxonomic unit. Humans, chimpanzees, gorillas and orangutans form the family hominidae, which is a taxon.
Antibiotic resistance	Resistance to drugs causing harm to bacteria
Pathogen	Microorganisms that can cause disease
Vertical gene transfer	The transfer of genetic information from biological mechanisms pertaining to parent-to-offspring inheritance
Horizontal gene transfer (HGT)	The transfer of genetic information between organisms through means other than parent-to-offspring inheritance

Contents

List of Figures

List of Tables

1
Introduction

1.1 Background

1.1.1 Antibiotic resistance and horizontal gene transfer

Antibiotic resistance is well understood to be a growing global threat to human health. In fact, the effects of antibiotic resistance is already causing millions of deaths worldwide annually [4]. On a broad level, the World Health Organization considers the biggest driving force of antibiotic resistance proliferation to be misuse and overuse of antibiotics [5]. Antibiotic overuse puts an elevated selection pressure on bacteria with the ability to resist drugs. Thus, with poor antibiotic usage practices, non-resistant bacteria are effectively killed off and the resistant bacteria in the population can freely produce offspring with inherited antibiotic resistance genes (ARGs). If the bacteria containing the ARGs are pathogenic, we have just characteristically described the start of a potentially deadly, almost inevitable, spread of disease.

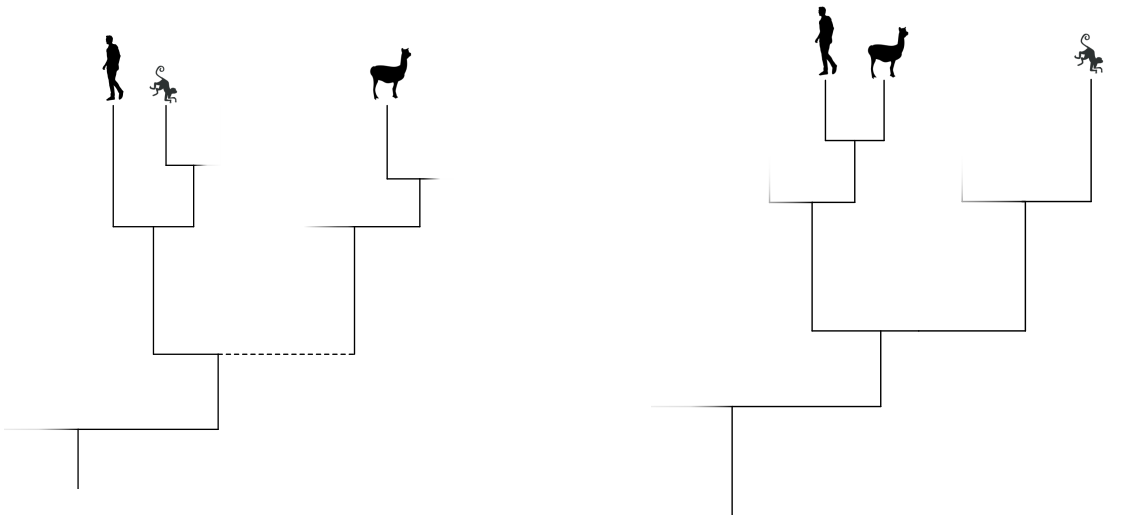
The dissemination of resistance is central to this predicament. Alongside vertical transmission (parent to offspring), horizontal gene transfer (HGT) stands as a key mechanism for ARG spread among bacteria [6]. In a simplified example, imagine a population of bacteria in your gut. Through some mechanism, an antibiotic-resistant bacteria spreads copies of a resistance gene into the bacterial population in your gut. Other compatible bacteria pick up the fragments and incorporate it in their own genetic information, and acquire antibiotic resistance themselves. This means that ARGs can be disseminated across greater parts of bacterial populations than would be possible with only vertical gene transfer. Indeed if the conditions are right, there is evidence that HGT readily occurs across bacterial species, genera, and even higher levels of taxonomic barriers [7][8].

Because of the enormous number of bacterial cells on earth, factors promoting resistance - such as mutations in critical places - probably occur all the time but are not necessarily detected or even directly encountered by humans. There is evidence that commensal bacteria in various environments serve as breeding grounds for ARGs, which makes the scenario even more likely that an antibiotic-resistant bacteria of a typically benign species could transfer ARGs to more distantly related pathogenic bacteria. Furthermore, HGT events are believed to play a major role in transporting these potentially novel resistance genes to environments housing pathogens that can come in contact with humans, such as agricultural land, sewage treatment plants,

hospitals, or other indirect means [9]. In other words, when an outbreak of ARGs occurs in a hospital, this is only the tip of the iceberg. This calls for wider screening to further elucidate the presence of new and emerging resistance genes, and importantly for the development of quantitative methods to detect HGT events involving ARGs spreading across bacterial populations.

1.1.2 Phylogenetic gene trees combined with taxonomic information provides a possibility to infer HGT events

Phylogenetic trees provide a means to show evolutionary relationships between organisms. In more colloquial terms, these trees typically depict "trees of life", visualizing the evolution of species over time. For example, humans and bonobos are relatively close cousins in a typical tree of life of animal evolution [10]. Notably, phylogenetic trees can also be constructed based on genomic data beyond species differentiation. A *phylogenetic gene tree* can, for instance, be assembled using genes found in an organism that associated with a specific type of antibiotic resistance. The genes used to construct the tree would in turn have their origins in different bacterial organisms - their host organisms - which may be of different species. However the phylogenetic gene tree would be built only with respect to the genetic information of the resistance genes. This means that species that are more closely related in terms of evolutionary history might not necessarily be close in a gene tree. However, even if a gene tree is built only with respect to a certain gene or other genetic fragment, in some situations we can expect the gene tree to closely match a tree of life. If a gene or family of genes is well conserved, meaning that it remains relatively constant through evolutionary time, then we can expect a gene tree based on this gene or family of genes to closely resemble a tree of life. On the other hand, if we construct phylogenetic gene trees based on genes that are known to be highly variable, we would probably end up with a gene tree not resembling a tree of life (Fig 1.1).
Now, consider that Figure 1.1b reflects a real result from constructing a phylogenetic gene tree. There are a few scenarios that reflect what could have caused humans and alpacas as host organisms to end up close in a gene tree. One scenario is that a genetic event took place far back in evolutionary time, at the point of common ancestry between humans, alpacas and bonobos, which caused the gene to evolve in three different versions separately from that point onward, side by side. In this scenario, the human and alpaca gene versions coincidentally maintained similarity over time. Another possibility is that a more recent event occurred, in which the gene either from alpaca to human or vice versa was transferred by means of HGT. A HGT event might also have taken place further back in evolutionary time, and like in the first case the different versions may have evolved side by side since then. For complex multicellular organisms like humans, bonobos and alpacas, HGT does occur[11]. However, the occurrence of HGT events in these organisms is notably less frequent compared to the common and prevalent phenomenon of bacteria-to-bacteria HGT. Drawing an analogy with the human-alpaca example, in a phylogenetic gene tree centered around antibiotic resistance genes, the juxtaposition of two distantly related host bacteria in close proximity could serve as compelling evidence of an



(a) Hypothetical phylogenetic tree of life. Humans and bonobos are more closely related species to each other than to alpacas.

(b) Hypothetical phylogenetic gene tree. There may be a gene such that the construction of a phylogenetic gene tree places the gene hosts, humans and alpacas, more closely to each other than to bonobos.

Figure 1.1: Differences between phylogenetic trees of life and a hypothetical phylogenetic gene tree. The two trees do not accurately reflect true phylogeny and are only used for illustrative purposes.

HGT event involving ARGs between these bacterial species. This concept introduces the possibility of using such patterns to develop a methodology for quantitatively assessing the likelihood of a gene being transferable through HGT across organisms spanning varying taxonomical levels.

1.2 Aim

The aim of this thesis is to develop and validate a method for inferring horizontal gene transfer (HGT) events from bacterial genetic data. This method focuses on quantifying discrepancies between resistance gene trees and host taxonomy as a signal of HGT. Additionally, its accuracy is validated, and its practical applicability is demonstrated through analyses of real-world bacterial sample data. Ultimately, this work seeks to contribute to the largely underexplored field of antimicrobial resistance gene (ARG) inference using phylogenetic trees, providing a foundation for future research and methodological improvements.

2
Methods

This chapter aims to explain the developed methods in this thesis. The first part describes the data sources used in the thesis. The following parts describe constructing the main method through gene tree building, gene tree cutting, transformation of data to an appropriate format, and lastly the HGT-score algorithm.

2.1 Data description

Data from various sources were used and generated in different stages of the overall method. The flow of data, from data sources to critical steps in the HGT-score method, is shown in Figure 2.1. The source data was fetched from NCBI GenBank and BLDB, and transformed by running fARGene. The data are described in the following sub-chapters.

2.1.1 Bacterial genomic data from NCBI GenBank

Genes are made up of nucleotides (shortened as A,C,T or G) put together in sequences. A common way to store nucleotide sequences so that they are accessible in a computational manner is through string representations in FASTA file format [12]. Huge amounts of genomic information are stored like this across many databases, many of which are publicly accessible. For the purpose of this thesis, it was of interest to apply the developed method on large amount of sequenced bacteria. A suitable database was the NCBI GenBank database [13], which contains all publicly available DNA sequences from recorded samples from many institutions across the world. Entries in the database also include the presumed species of the isolate in the sample, and other metadata. Since all genes of an organism exist within its genome, this also means that this database gives access to all the genes of all sequenced bacteria, including antibiotic resistance genes.

2.1.2 fARGene was used to predict ARGs for each bacterial genome

Whether a gene codes for antibiotic resistence or not is another question. While there is a wealth of known ARGs that are registered and easily looked up in each sequenced bacterial genome, undiscovered or novel ARGs would be missed. To catch both known and undiscovered ARGs, the ARG prediction tool fARGene [1] (fragmented Antibiotic Resistance Gene Identifier) was used. The tool uses Hidden Markov Models (HMMs) as its predictive engine, where each HMM corresponds to a

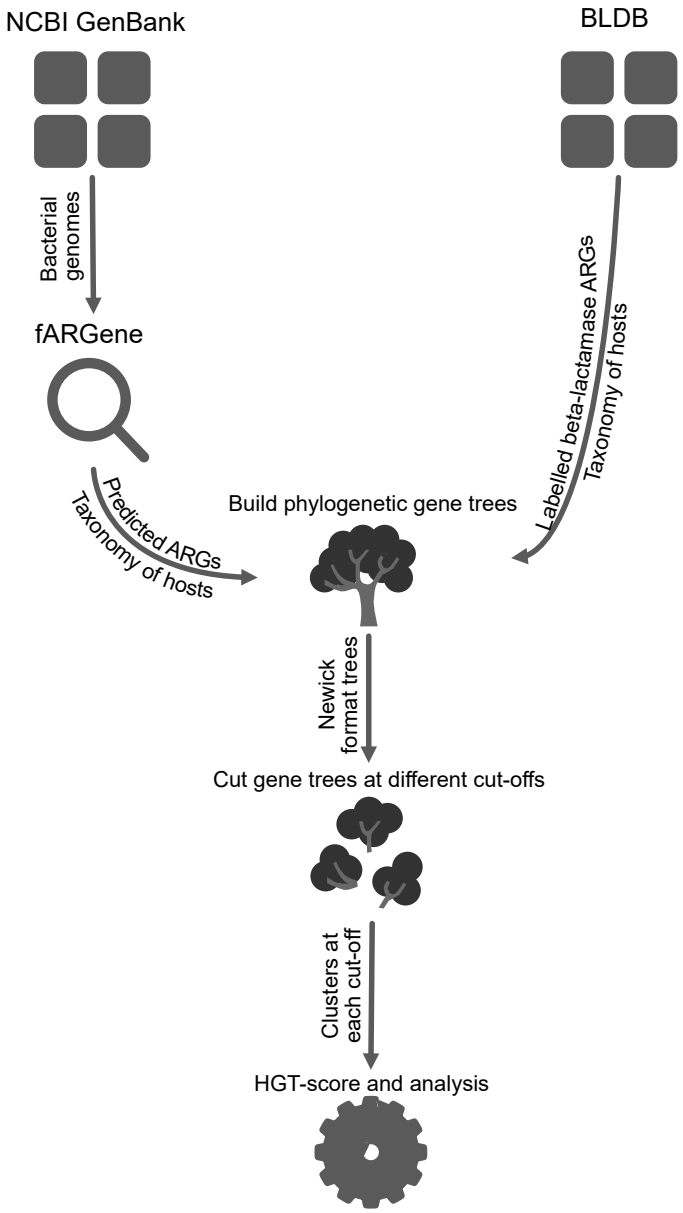


Figure 2.1: General description of data flow in the project. NCBI GenBank and BLDB were the main data sources used in this thesis. Genomic data were transformed into ARGs through the use of fARGene [1], and phylogenetic gene trees (one per ARG type) were built using FastTree [2]. Additionally, taxonomy for each bacterial host were inferred using taxonomizr [3]. Gene trees were cut, producing clusters of bacterial hosts whose ARGs for a given type were similar. The dissimilarity in host taxonomy within each of these clusters was measured. As the last step, a metric was assigned to each cluster to indicate the potential for HGT having occurred.

different antibiotic resistance type. For example, β-lactamase fARGene was applied on bacterial genomic data originating from the NCBI GenBank database, using settings specified in Appendix A.1. For each genome and HMM, the output from fARGene was a list of predicted genes in FASTA format. fARGene was run on all sequenced bacteria from the GenBank database using a heavily parallelized method, and results were stored in a recursive directory structure for ease of computational access.

2.1.3 Labelled validation data from BLDB

To validate the HGT-score method, the same type of information as for analysing predicted resistance genes was used, but from another source. Labelled data from beta-lactamase database (BLDB)[14] consisting of β-lactamase class A resistance genes was used to this end. In the database, each β-lactamase class A ARG is labelled either Acquired (A), Natural (N), or is unlabelled. Acquired, meaning that they are genes that likely have been passed down with the use of bacterial mechanisms commonly associated with HGT, and vice versa for Natural. Unlabelled ARGs were ignored. Host information for each gene was also extracted.

2.2 Phylogenetic gene trees

2.2.1 Building gene trees

To serve as the primary object on which evaluation of HGT is to take place, phylogenetic gene trees were built. Since these gene trees should reflect relationships between genes, i.e. closely related genes are supposed to be close to each other in the tree and vice versa, genes are used as the building blocks of the trees. To make results more interpretable, choose each phylogenetic gene tree to represent one type of antibiotic resistance gene, meaning that genes only from a specific ARG type are used in the corresponding gene tree. fARGene [1] has the ability to predict ARGs pertaining to many types of antibiotic resistance. The main type of predicted ARG used in this thesis was class A β-lactamases [15]. In general, beta-lactamases are enzymes that provide resistance against beta-lactam antibiotics like penicillins and many others. Other types of ARGs that fARGene predict are other classes of β-lactamase, tetracycline resistance genes, erm 23S rRNA methyltransferases, and Mph macrolide 2'-phosphotransferases (see [1]). The method of building phylogenetic trees relies on the FastTree 2.1 software. FastTree takes aligned genetic sequences as input, creates a rough first tree and then refines it in further steps (see [2] for details). For producing alignments of genetic sequences, the MAFFT software [16] was used with automatic settings. Thus, for each type of antibiotic resistance gene, a corresponding gene tree was built using FastTree with automatic settings, which outputs a phylogenetic gene tree. Some important parts of the phylogenetic tree are now explained, and some nomenclature is introduced.

Consider a phylogenetic gene tree in rectangular representation (Fig. 2.2). Notice that the tree-like structure arises from bifurcation nodes in the tree. Each of

these nodes represent a point of common ancestry of everything to the right of that node. In phylogenetic trees representing evolutionary relations between species, this concept is perhaps more easy to understand since the common ancestor at that point in time would be represented by an organism. The interpretation however is the same when speaking about gene trees. For example, a branching of a gene tree could represent that a gene was duplicated and those duplicates evolved separately but side by side from the branching node onward. In other words, an ancestral gene is analogous to an ancestral organism. There are elements apart from bifurcation nodes which symbolize evolutionary processes. Lengths of the branches themselves contain information about evolutionary time: the longer the branch, the more genetic change has occurred. Enter another important concept: the phylogenetic tree *root*. In order to draw conclusions about evolutionary relationships in time, the phylogenetic tree must be rooted using an outgroup, meaning that it must have a node symbolizing the most recent common ancestor of all the leaves in the tree. This allows for us to interpret a flow of genetic information from the root on the left (furthest back in evolutionary time), toward the leaves (most recent evolutionary time). How a tree is rooted introduces another important part of the tree, the *outgroup*. In gene trees, the outgroup itself is a genetic sequence that is known to be more distantly related to the rest of the genes of a given tree than they are to each other. The node which binds the outgroup to the rest of the tree is thus the most recent common ancestor to all leaves including the outgroup, in other words the root of the tree. Another part of phylogenetic tree nomenclature is a *clade*, a subgroup of a phylogenetic tree containing at least one leaf, and the lowest common ancestor of all other included leaves. The green rectangle in Figure 2.2 containing leaf A, B, and C is a clade because it includes the lowest common ancestor of all leaves. However the red rectangle is not, since it includes A,B and C but only the lowest common ancestor of A and B.

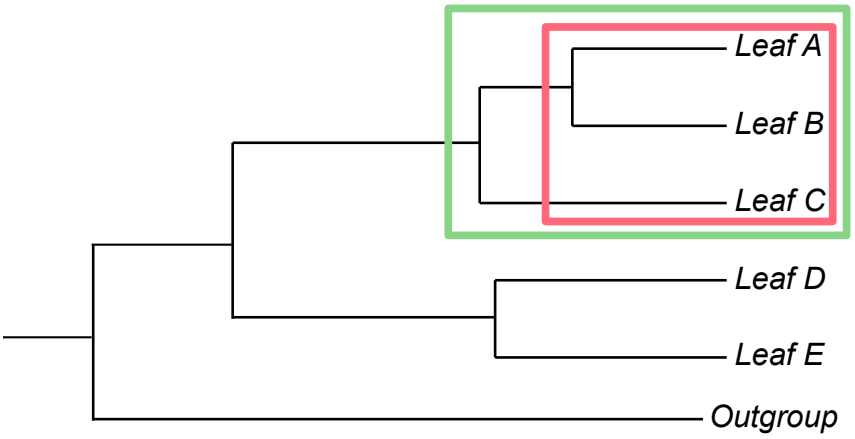


Figure 2.2: Rectangular representation of a phylogenetic tree. The tips of the tree are called leaves. Each bifurcation, or branching, of the tree gives rise to clades, which are parts of the phylogenetic tree which must include leaves and at least one common ancestor unless it includes only itself. The part of the tree marked with green constitutes a clade, whereas the part marked with red does not.

For each collection of identified ARGs per ARG type, the output of FastTree was a full representation of a phylogenetic gene tree in Newick format [17]. This includes qualities like bifurcations, branch lengths and leaves, however the tree is unrooted per default. Including an outgroup in the MAFFT alignment allows for the final step of the tree building, *re-rooting* of the tree, using the Biopython Phylo package [18]. This process was repeated for all considered types of antibiotic resistance genes, using all of their respective and corresponding predicted ARGs as input. The result is one phylogenetic gene tree per antibiotic resistance type. In Figure 2.3, a subset of the tetracycline ribosomal protection gene (tet rpg) tree is shown in cladogram format (meaning that branch lengths are made not meaningful to provide good visualization), and in circular format.

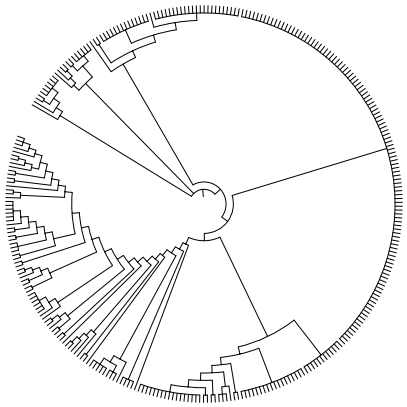


Figure 2.3: Subset of a tetracycline ribosomal protection gene (tet rpg) tree constructed from predicted tet rpg ARGs using fARGene.

The tree building step was implemented using the mentioned packages, bash scripts for pipelining, and Python. All gene trees were stored in Newick format in the SAGA database of Erik Kristiansson's lab group.

2.2.2 Cutting phylogenetic gene trees

Now that the phylogenetic gene trees have been built and the correct evolutionary relationships between ARGs within each antibiotic resistance type have been established, we want to be able to infer HGT events from the gene trees. It has been explained that an event of having two leaves next to each other in a gene tree, but whose hosts in which they reside (bacteria) are distantly related, probably can arise due to HGT. To turn this idea into something computable, we need to formalize the definition of closeness even further. The proposed method in this thesis involves as a first step to perform cuts in the gene trees, which produces clusters. Cutting a phylogenetic tree can be seen as taking clades with nodes above a certain threshold and collapsing all leaves within those clades. In order to cut in the tree, we define this threshold as a tree *cut-off*, which is a positive real number and is interpreted as a value of cumulative branch length calculated from a root. When a cut is made at a certain cut-off, all leaves in a clade with cumulative branch length larger than that cut-off are placed in the same cluster. In practice, tree cutting is achieved by

using the ETE3 package [19] which contains methods for treating phylogenetic gene trees as objects with attributes such as nodes, leaves, branch lengths, etc.

For a specific cut-off and ARG tree, the Newick file of the tree is first read and turned into a *tree* object. Then all leaves of the tree object are obtained, along with their full ancestral lineages (from leaf toward root). For each leaf, their respective lineages are traversed backward, recording each branch length along the way to produce a list of branch lengths. The cumulative sum of branch lengths is then calculated on the reversed list of branch lengths (to get the opposite direction, of root toward leaf) to be in accordance with the direction of evolutionary time. With this, we have a representation of evolutionary distance between each node of the ARG tree in question. Now, we need to find where in this list of cumulative branch lengths to perform a cut. Since the list is always increasing, this is done by simply finding the index of the smallest value in the list that is larger than the cutoff. For example, in a list 0,3,6,9,12 the cut-off value of 3.5 is smaller than 6, 9 and 12, however 6 is the smallest value out of them, and the index of 6 is 2 (starting at the 0:th index), and 2 is then set as the cutting index. To achieve a cut, the tree node at the same index (starting from root) as the cutting index is chosen, and the descendant leaves of that node are collapsed into a list. This is repeated for each leaf, and only unique lineages are kept, to ensure that each leaf only shows up in a single cluster. Parsing of the object representations of the leaves is done to ensure that the id connecting an ARG to its host organism (called accession numbers) are kept intact.

In *Python* terms, the result after a cut is thus a list of lists, where each inner list represents a cluster of bacterial genomes, and each element of an inner list is a leaf: individual bacterial genomes. The leaves themselves are represented by a string of characters, called *accession number*, that identifies the genome (in reality it is an *id* of the sample in GenBank) of the host organism which the ARG was found in. Using the shallowest cut-off of zero will yield only one cluster containing every leaf, whereas the deepest cut-off, the value of which depends on the tree, will yield a number of clusters equal to the amount of leaves in the tree, i.e. one cluster per leaf populated only by that leaf. A simple example to illustrate tree cutting is found in Figure 2.4 below. Cut-off 1 corresponds to the shallowest cut possible where all leaves in the tree are collapsed into a single cluster, whereas cut-off 5 corresponds to the deepest cut possible and each individual leaf is only collapsed onto itself, resulting in one cluster per leaf populated only by that leaf, i.e. singletons. Cuts with shallower depth than cut-off 1 will yield the same result as for cut-off 1, and the same is true for cut-offs deeper than cut-off 5. The schematic below shows cutting a phylogenetic tree in rectangular representation. In the circular representation case such as in Figure 2.2, one can instead imagine tree cutting at increasing cut-offs as cutting the tree along the diameter of a circle of increasing radii. The result is however the same in either representation.

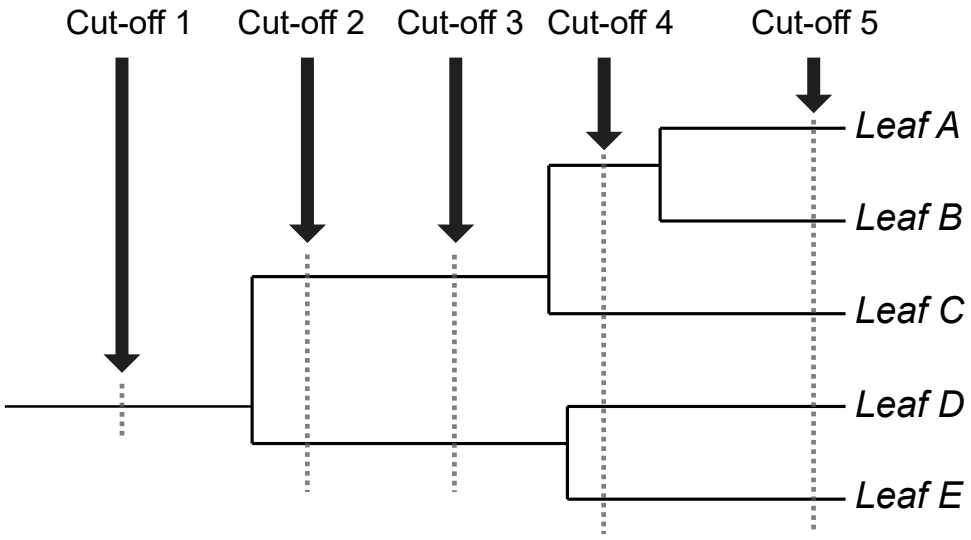


Figure 2.4: Representation of cutting a phylogenetic tree at different cut-offs. Cut-off 1 will yield only one cluster containing all leaves, cut-offs 2 and 3 will both yield two clusters, the first containing Leaf A, B and C and the second containing Leaf D and E. Cut-off 4 yields four clusters, first with Leaf A and B, second C, third D, and fourth E. Cut-off 5 represents the strictest cut-off possible for this tree yielding five clusters, each corresponding to an individual leaf.

All phylogenetic gene trees were cut using cut-offs from 0 to 210, with a step-size of 3. This particular choice was mostly informed by that 210 seemed to be a reasonable stopping point for most ARG trees. Cut-offs past 210 in most cases did not result in any new clustering meaning that all leaves existed as singletons. Some ARG trees however produced only singletons much earlier than a cut-off of 210, and on the other hand some ARG trees did not arrive at a deepest cut-off point by 210. The step-size of 3 seemed like a good enough resolution while also taking computational time into consideration. To speed up computation, parallel processing was used. For each ARG tree and cutoff, a file with one row per cluster was stored in the SAGA database, like the rerooted phylogenetic gene trees. A file resulting from a cut-off of 0 always has one row containing all leaves, and a file resulting from the deepest cut-off would always have one row per leaf, i.e. only singletons, in the tree.

To summarize how the results from the tree cut-off method are organized, the highest level consists of the different antibiotic resistance types which represents the phylogenetic trees built from their predicted ARGs from fARGene. This level is followed by the tree cut-off level, where each antibiotic resistance type has cut-offs from 0 to 210 with step-size 3. Each cut-off in turn results in a variable amount of clusters, depending on how shallow or deep the cut-off is. Finally, the ”atomic” unit of data are leaves, which are also variable and contained within each cluster. See Figure 2.5 for a representation of this.

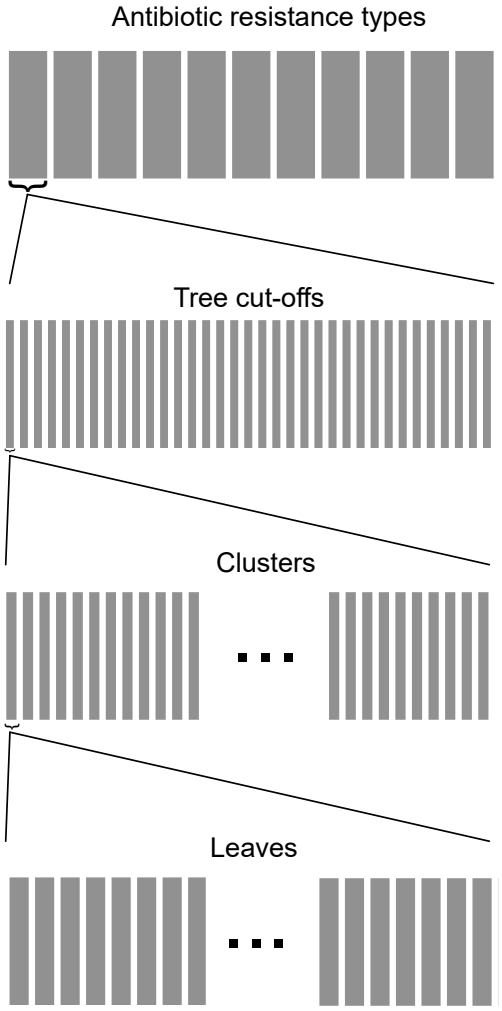


Figure 2.5: Representation of the result organization following the tree cut-off method. The order is, from highest to lowest: antibiotic resistance types, tree cut-offs, clusters, and leaves. The dots (...) on the cluster and leaf levels represent that the amount of clusters and leaves are variable.

2.3 Fetching and processing of host taxonomy into taxon counts

In addition to the genetic relationships through clusters from phylogenetic gene tree cutting, the HGT-score method is also necessarily based in the taxonomic information of the host organisms themselves. Specifically, the taxon counts per cluster. Before generating the taxon counts, the full taxonomic lineage (species, genus, family, order, class, phylum, kingdom) of each host must be known. To produce this data, several processing steps were taken.

2.3.1 Preparation of taxonomic information

Using the prepareDatabse function from the taxonomizr R package [3], an SQL file mapping a *taxonomic identifier* of each host to taxonomic lineages (phylum, class, order, family, genus, species) was created. Another file relating accession numbers and taxids was used to obtain the taxid corresponding to each host's accession number. The SQL file was queried to retrieve the taxonomic ranks of species, genus, family, order, class, and phylum, along with the relevant taxids. This collection of information was turned into a data table (Figure 2.6 with columns for each taxonomic rank and one column identifying the host, consisting of the retrieved taxa and their corresponding accession numbers. Since multiple ARGs can be predicted from the genome of a single host, this produced some duplicates in the dataframe, which were removed (since we at this point in the processing are only interested in the relation "taxonomic lineages to accession number"). In some cases, the full lineages were not available due to lack of knowledge, mislabelling, or other errors. Out of the 44779 total unique accession numbers, 459 of them produced taxonomic lineages with at least one missing taxon. For some of these cases, it might have been possible to do more thorough reviews and identify at least probable taxons at the missing levels. However, for simplicity they were removed. The cleaned accession-taxa dataframe was exported as a csv file.

	Phylum	Class	Order	Family	Genus	Species	assembly_accession
0	Actinobacteria	Actinomycetia	Bifidobacteriales	Bifidobacteriaceae	Bifidobacterium	Bifidobacterium longum	GCA_000003135.1
1	Firmicutes	Bacilli	Bacillales	Bacillaceae	Bacillus	Bacillus cereus	GCA_000003135.1
2	Firmicutes	Bacilli	Bacillales	Baccilaceae	Bacillus	Bacillus mycoides	GCA_000003925.1

Figure 2.6: Structure of accession-taxa dataframe. The first three rows of the resulting file mapping accession number (column named "assembly_accession") to the taxonomic ranks, which are one per column.

For continuing the method, the relevant data objects at this point were: the accession-taxa dataframe, and the genetic tree cut-off files containing the clustering (one for each antibiotic resistance type and cut-off). To assemble the relevant data in the same place, a *python* nested dictionary named *genedict* was created with the structure genedict[resistance_type][cut-off][data_type], where the dictionary keys symbolized by "resistance_type" were the different considered antibiotic resistance types, the keys symbolized by "cut-off" were the different tree cut-offs used, and the "data_type" keys were two different ways to represent the leaves in the clusters. The first representation is that which is already used in the cut-off files: by accession number, which provided an index for mapping. The second, and crucial to the HGT score method, representation is by taxon counts.

2.3.2 Computation of taxon count per cluster

Taxon counts were computed as counts of each taxon, for each taxonomic rank, per cluster. This relies on the accession-taxa dataframe 2.6. To illustrate, again consider the phylogenetic tree in Figure 2.4. Let us use the clusters resulting from cut-off 3 as data for the example, meaning the two clusters (A,B,C) and (D,E). We symbolize the leaves' taxons pertaining to species, genus, family, order, class, and

phylum, respectively as $N_s, N_g, N_f, N_o, N_c, N_p$ where $N \in A,B,C,D,E$. For example, considering the first cluster, let the leaves A,B, and C belong to the same phylum, class, order and family. Let A and B share genus, but let C be of a different genus. Finally let us assume that A,B and C each are of different species. For the second cluster let leaves D and E be of different family to each other but share taxons for order, class, and phylum. The *taxon counts* of a cluster is defined as the counts of taxons on each taxonomic rank. Shown in Table 2.1 are the respective taxon counts of the two clusters in our example.

Table 2.1: Taxon counts of two clusters resulting from using cut-off 3 on the phylogenetic tree of Figure 2.4.

(a) Taxon counts of the first cluster. A,B and C are of different species, A and B share taxons for genus, and A,B and C share taxons for family and above.

Species	A_s: 1	B_s: 1	C_s: 1
Genus	AB_g: 2	C_g: 1	
Family	ABC_f: 3		
Order	ABC_o: 3		
Class	ABC_c: 3		
Phylum	ABC_p: 3		

(b) Taxon counts of the second cluster. D and E share taxonomy up to and including family, but differ in taxons for order and above.

Species	D_s: 1	E_s: 1
Genus	D_g: 1	E_g: 1
Family	D_f: 1	E_f: 1
Order	DE_o: 2	
Class	DE_c: 2	
Phylum	DE_p: 2	

The process of collecting taxon counts for each cluster was repeated for each antibiotic resistance type, and for each cut-off, for a total of 1190 cut-off cluster files.

2.4 The HGT-score method

As the core of the overall method, a scoring algorithm was created to quantify how likely it was that a HGT event has occurred between hosts, given input according to Section 2.3. Here, we walk through the steps in the algorithm to produce the output.

2.4.1 HGT-score algorithm

Each cluster for a given tree cut-off is given an HGT-score. In summary, the HGT-score is a positive rational number and ranges from 0 to 7 symbolising the taxa of species (1), genus (2), family (3), order (4), class (5), phylum (6), and kingdom (7). A score of zero represents a non-valid cluster (the validity criterion is shown in Equation 2.2).

The algorithm relies on the idea that a greater taxonomic difference within a cluster means that HGT is more likely to have occurred, and shall thus score such clusters higher. Note that the HGT-scoring is done only within each cluster, meaning

that clusters are directly independent of one another in the scoring. Initially, the accession-taxa dataframe, indexed on species, and the taxon counts were loaded for use. A list of the names of the different taxonomic ranks species, genus, family, order, class, and phylum was also initialized, to ease access of the taxon counts.

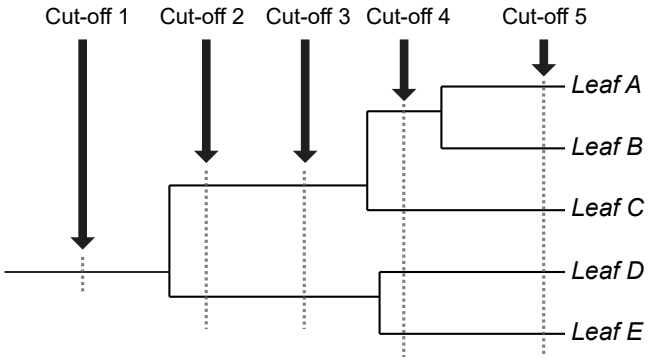


Figure 2.4. Representation of cutting a phylogenetic tree at different cut-offs (reused from earlier).

Consider again the example of using cut-off 3 in Figure 2.4, which results in clusters in turn represented by taxon counts, one taxon count per taxonomic rank, as in Table 2.1. Let us first focus on the first cluster (A,B,C). We record the amount of leaves in the cluster, which is 3. We then create a list of all different species in the cluster, A_s,B_s,C_s. Then, to get a list of the full taxonomies of all the species, the accession-taxa dataframe is queried by each species in the species list. The taxon counts for the cluster are retrieved. For every leaf, we calculate the fraction of shared taxons for each taxonomic rank. Symbolising this fraction as $f(N_l)$, the fractions in our example would be, with one taxonomic rank per row of equations

$$f(A_s) = \frac{1}{3} \qquad f(B_s) = \frac{1}{3} \qquad f(C_s) = \frac{1}{3}$$
$$f(AB_g) = \frac{2}{3} \qquad f(C_g) = \frac{1}{3}$$
$$f(ABC_f) = \frac{3}{3} = 1$$
$$f(ABC_o) = 1$$
$$f(ABC_c) = 1$$
$$f(ABC_p) = 1.$$

For each leaf, we can observe how this ratio grows as we traverse up the taxonomic ranks. This sequence for leaf A is $\frac{1}{3}, \frac{2}{3}, 1, 1, 1, 1$. Notice that once the ratio reaches 1 for a given leaf, it will also be 1 for every leaf in the cluster.

We define the taxonomic rank at which the ratio reaches 1 as the *lowest common ancestor* (LCA). In this case, the LCA is ”family” since that is the lowest rank at

which the ratio is 1. To quantify concept of LCA, we symbolize taxonomic ranks as numbers, where 1 is species and 7 is kingdom. This makes the LCA of this cluster equal to 3. Now we define another quantity, the *p%-lowest common ancestor* (p%-LCA), which is the lowest taxonomic rank where the fraction of shared taxons is equal or bigger than a threshold p. Note that this level unlike the LCA is not necessarily the same for all leaves. For example, the 55%-lowest common ancestor for the leaf A is genus (with fraction $\frac{2}{3} \geq 0.55$), since that is the lowest taxonomic rank where its fraction of shared taxons is equal to or higher than 0.55. For B, the same holds true. However for C, the fraction of shared taxons at genus level is $\frac{1}{3} < 0.55$. Only at family does C reach a fraction of shared taxons $\frac{3}{3} \geq 0.55$. If a leaf should not reach a fraction of shared taxons greater than or equal to p even at the level of phylogeny, it is assumed that kingdom is the p%-LCA, with a score of 7, and the same is true for the LCA. A property to note is that the p%-LCA is always less than or equal to the LCA, which might be intuitive with the realization that LCA is in fact the 100%-LCA. Now, the HGT-score of leaf N is defined as the fraction between the LCA and the p%-LCA,

$$H(N,p) = \frac{LCA}{p\%\text{-}LCA(N)}. \tag{2.1}$$

The smallest possible value for the HGT-score is 1, which is produced when $LCA = p\%\text{-}LCA$. For example, if we let $p = \frac{3}{4}$, the 75%-LCA and LCA for leaf A are both family, yielding $H(A) = \frac{3}{3} = 1$. The largest possible value for the HGT-score is 7, which occurs if the LCA of a cluster is 7 and the p%-LCA is 1. But for our example cluster, a HGT-score of 7 is not possible regardless of the choice of p, since the LCA is always 3 and p%-LCA cannot be smaller than 1. This means that the LCA is the maximal HGT-score of any leaf in this cluster. In our specific case, the maximal HGT-score of 3 can be achieved if p is set to $\frac{1}{3}$ or lower.

The choice of p additionally constrains the lower boundary on the number of needed leaves in a cluster for a scoring to be valid. For example, given $p = 0.99$, at some taxonomic rank we need a taxon similarity of 99% to be able to calculate a HGT-score for any leaf. If we have a cluster with only two leaves, we would never be able to reach this threshold of similarity without simultaneously reaching the LCA (100% similarity), so the 99%-LCA does not have any logical meaning in that case. The best we can do for generating meaningful results from a chosen p%-LCA with a cluster of size 2 is 50%. In general, generate a meaningful HGT-score to any leaf in a cluster, the lower limit of the cluster size for a given p is

$$\textit{smallest cluster size}(p) = \frac{1}{1-p} \tag{2.2}$$

and the clusters that do not fill this criteria have their HGT-score set to zero, and are treated like the singleton clusters. This means for example that for $p = 0.99$ the minimum cluster size is 100. In fact, in our previous example, using $p = \frac{3}{4}$ was invalid, since $\frac{1}{1-\frac{3}{4}} = 4$, but the cluster size of (A,B,C) was only 3. Since we aim to relate a score not to each individual host, but to a collection of hosts in a cluster, we further define the HGT-score of a cluster as the maximum HGT-score of any leaf

in the cluster as per Equation 2.3

$$H(C) = \max(HGT(N)), N \in C. \tag{2.3}$$

This is done to increase the sensitivity of the method. The use of other statistics could also be evaluated. In any case, the cluster HGT-score is calculated for every antibiotic resistance type for every cut-off and for each cluster.

Finally, the final result or product of the method is an HGT-score for every cluster, which is stored in another dictionary object of structure *scoredict*[resistance_type][cut-off], where each dictionary entry is a list of the HGT-scores in each cluster for the particular antibiotic resistance type and gene tree cut-off.

2.5 Validation of HGT-score using labelled data

2.5.1 Treating labelled data

The model was validated by assigning HGT-scores to clusters which originate from labelled data, and then comparing the scores against the expected outcome. The labelled data consisted of antibiotic resistance genes of type beta-lactamase class A, fetched from BLDB (Beta-Lactamase DataBase) [14], which in total contained 1737 beta-lactamase class A sequences. There were two labels, "Acquired" (A) and "Natural" (N), and we also consider a third label which is neither, or unlabelled. The labels refer to in what genetic context that the ARGs have typically been observed to inhabit, with respect to mobility between bacterial hosts, which is important in order to allow HGT. If the label is "Acquired", this means that the ARG generally is contained within mobile DNA often enabling HGT [20], with the opposite being true for the Nat label. The genes in the validation data were represented in the same FASTA format as the predicted ARGs from fARGene, the same method to build and cut phylogenetic trees was used to generate the cut-off cluster files. However, the labelled data did not have any associated accession numbers, so it is not straightforward to retrieve the full taxonomy like for the fARGene files.

To address this, a method to compare the similarities of the labelled beta-lactamase class A genes with antibiotic resistance genes already present from the fARGene results in the database was employed. This method relied on usage of the *BLAST* algorithm [21], implemented in a software called *BLASTp* [22] which performs local sequence alignment to find similarities between different biological sequence information. Applying the method starts with some preparatory work by construction of a database to query against, in order to reduce the computational effort needed when searching the sequences. The database was created using the *makeblastdb* command on a file containing the sequences of labelled beta-lactamase class A genes, along their corresponding labels. The database was queried using BLASTp, with a file containing the sequences of the predicted fARGene beta-lactamase class A genes, using command line options *-outfmt 6* to get output in a easy-to-parse format, *-qcov_hsp_perc 80* to set a heuristic threshold of 80%, on how good the alignment must be to be included, and *-num_threads 20* to speed up computation. The

BLASTp output is a tab-separated file and is easily read into a pandas DataFrame. After reading the output into a pandas DataFrame, the relevant columns *qseqid*, *sseqid* and *pident* are extracted, where *qseqid* refers to the names of the queries which in this case are the fARGene beta-lactamase class A genes, *sseqid* are the names of the sequence that the query sequence has been aligned to, and *pident* is the extent to which the query and subject sequences have the same sequence at the same positions. A 100% *pident* means that the query is identical to the subject. The structure of the relevant part of the data is shown in Figure 2.7.

	qseqid	sseqid	pident
0	class_A.hmm_GCA_000003645.1_seq1	AAA22668.1|Bcl-3|_Nat	95.098
1	class_A.hmm_GCA_000003645.1_seq1	AAK53749.2|Bcl-2|_Nat	91.909
2	class_A.hmm_GCA_000003645.1_seq1	CAA58448.1|Bcl-4|_Nat	92.233

Figure 2.7: Structure of *BLASTp* output from querying beta-lactamase class A ARGs from fARGene against labelled class A beta-lactamase genes from BLDB. Only the first three rows, and only the used columns are shown. *qseqid* is the query name, *sseqid* is the subject sequence name, *pident* is the percentage identity.

When labelling the fARGene genes, if there is a BLAST match between a fARGene gene and a labelled BLDB gene, we want there to be a high enough similarity between them so that the fARGene gene plausibly also can ”inherit” the label of the BLDB gene. Therefore, a *pident* threshold of 95% similarity is used to filter out low matches. After filtering, 460709 matches remain. However, this still leaves a possibility for multiple matches between each fARGene gene and the BLDB genes. Therefore, the match with the largest *pident* value for each fARGene gene is chosen, which leaves a remaining 7674 unique matches. Since there are 11401 leaves in the beta-lactamase class A fARGene gene tree, this means that $11401 - 7674 = 3727$ leaves are chosen to be unlabelled. 3100 fARGene beta-lactamase class A genes were labelled Nat, and 4570 were labelled Acq.

Performing the labelling of leaves was a straightforward mapping of the accession number present in the *qseqid* column of the BLASTp output with the Acq or Nat label in the *sseqid* column.

2.5.2 Labelling clusters

Since the genetic information for the now-labelled fARGene beta-lactamase class A genes is still the same, we can simply continue the method on the cluster cut-off files from the tree-cutting. At this point however, we want to divide the validation set into two parts, each with different labels in order to see if the HGT-score method can discern between likely HGT clusters and unlikely ones.

At this point we have clusters where the leaves within the cluster are labelled Acq, Nat, or nothing. Since we are interested in scoring the clusters themselves, we need a strategy to ”color” the clusters from the labels of the leaves within them. This was

done by considering a cluster C with at least one Acq-labelled leaf in it as an "Acq" cluster, and a cluster with no Acq leaves in them and at least one Nat-labelled leaf in them as a "Nat" cluster, and the rest as "no label" as described below.

$$Label(C) = \begin{cases} \text{Acq}, & \text{if } \textit{at least one leaf in } C \textit{ is Acq} \\ \text{Nat}, & \text{if } \textit{at least one Nat leaf in } C \textit{ is Acq} \text{ and } \textit{no Acq leaf in } C \\ \text{No label}, & \text{otherwise} \end{cases}$$

An alternative labelling strategy was also employed, where the "Acq" clusters were chosen in the same manner as above, but where the "Nat" clusters were simply chosen as all clusters that are not "Acq".

2.5.3 Measuring difference in HGT-score w.r.t. label

To yield comparable and easily understandable metrics to assess the effectiveness of the HGT-scoring method on the labelled clusters, the mean across all clusters - Acq and Nat, respectively - was calculated for each tree cut-off. This allows us to easily color points on a Mean HGT-score vs. tree cut-off graph, and clearly see differences at each cut-off. However, to assess whether the difference between mean HGT-score of an Acq and Nat-cluster at a certain cut-off, we need to have a measure of the noise in the data. In other words, we want to assess how likely it was that the mean HGT-scores ended up as they did by chance.

Therefore, separate mean HGT-score calculations per cut-off were done, this time on clusters with the labels of Acq and Nat-labelled leaves randomly switched around. The process of re-labelling clusters was then done using the same logic as previously. Note the importance of not including the unlabelled leaves in this randomization since it would inflate the amount of Nat/Acq clusters compared to the real labelled data. The label-randomization / re-labelling of clusters was repeated for n=300 iterations, and the mean for both Acq and Nat-labelled clusters were calculated. Additionally, the Acq/Nat-differences were calculated for each cut-off, the means also recorded along with the 5:th and 95:th percentiles.

3

Results and discussion

This chapter will present results in three main parts: a summary of the HGT-score method, validation of the HGT-score method showing that the method correctly signals for HGT events, and finally results from applying the method in various cases.

3.1 The HGT-score algorithm

The HGT-score method developed in this thesis consists of several parts including data extraction, data preparation, and the HGT-score algorithm. The algorithm is defined per Equations 2.1 and 2.3 and is described in detail in the Method chapter 2.4. This section will recapture and illustrate the core properties of the HGT-score.

3.1.1 Properties of the HGT-score

The HGT-score quantifies the mismatch of host taxonomic distance and host proximity in a phylogenetic gene tree built using the hosts' respective ARGs. The branches of the gene tree are severed at different lengths based on a *cutoff* value. The severed branches contain groups of various leaves, which are called clusters. Membership of ARGs in such a cluster implicitly reflects ARG proximity as a function of cutoff. Lower cutoff values are closer to the tree root leading to fewer and larger clusters, and higher cutoffs are closer to the leaves, leading to more numerous and finer clusters (illustrated in Fig. 2.4). Given a cluster, the HGT-score is calculated for leaf N in the cluster as

$$H(N,p) = \frac{LCA}{p\%\text{-}LCA(N)}. \tag{2.1}$$

LCA is the taxonomic rank (1 to 7, from species to kingdom) of the least common ancestor among all hosts in the cluster. p%-LCA is the lowest taxonomic rank at which the leaf shares p% of its taxonomy with the other hosts in the cluster. Then, the HGT-score for the whole cluster is calculated using a *cluster-level statistic*, which for the produced results of this thesis was chosen as the maximum HGT-score of any host in the cluster. The choice of p further imposes a size criterion on clusters which is $\frac{1}{1-p}$, and disqualifies clusters with a smaller number of hosts than this fraction from receiving a HGT-score.

The three central parameters which affect the output of the algorithm are thus:

cutoff, p, and choice of *cluster-level statistic*, and their roles are summarized in Figure 3.1.

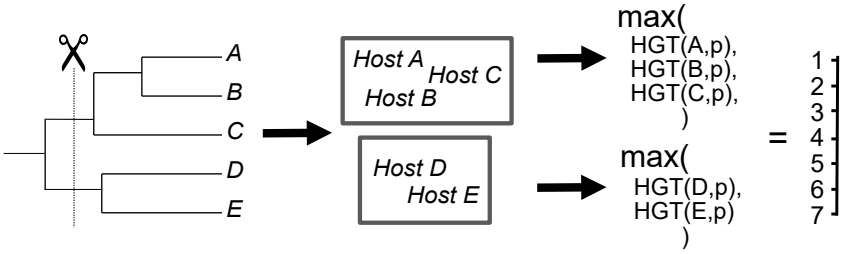


Figure 3.1: The roles of cutoff, p and cluster-level statistic in the HGT algorithm. A cutoff on a phylogenetic gene tree produces clusters. The taxonomies of the ARG hosts in each cluster is used as information when calculating the HGT of each cluster, along with p. Finally, the maximum HGT-score within the cluster is given to the cluster, which ranges from 1 to 7 for valid clusters.

The effects of the p and cutoff parameters are demonstrated using Figure 3.2. A phylogenetic gene tree was constructed using host bacteria with 11402 class A beta-lactamase ARGs, and the same tree is repeated for the three circular cladograms shown, with p of 55%, 77.5% and 99% respectively. Note that the branch lengths of the shown tree were adjusted such that each leaf is equidistant from the center, making it a deformation of the real underlying tree. The root of the phylogenetic tree, the point point in evolutionary time common to all genes involved in the tree, corresponds to the center of the cladograms. Moving outward from the center eventually leads to the tips where each leaf corresponds to an ARG and its origin host bacterium. To apply a cutoff in this circular representation can be seen as taking a concentric circle with radius proportional to the cutoff, and cutting along it create the clusters. The results of cutoffs 75, 93, 99, 102 and 108 are seen as five circular bands, with cutoff increasing radially. For each cutoff, clusters were HGT-scored with results shown as colors on each band, from off-white as 0 (disqualified clusters) to black as 7, the maximum HGT-score possible. I want to address some imperfections in the visualization, so it does not muddle the interpretation of the results. As mentioned above, the rendering of the tree forces branch lengths are such that all leaves are equidistant from the center. This produces some visual artifacts in the cladograms which do not accurately reflect reality. One such artifact is the black rim on the border of the trees, which is actually the 11402 leaves used to build the gene tree which become squished together due to the branch length forcing. In reality, there is branching happening close to the rim not visible to the eye here. These artifacts and imperfections are however isolated to the tree visualization itself, and not inherited by the cutoff bands due to them being based on the underlying real tree.

When presenting the results that are illustrated in Figure 3.2, I will be directing attention by seeing the cladograms as analog clocks (12 o' clock is straight up, 6 o' clock straight down), and with the first cutoff band referring to the innermost band,

and the fifth the outermost band. The majority of clusters for the chosen cutoffs for all choices of p contained too few leaves in order to receive a HGT-score, as evident by the mostly off-white colored bands. The effect of the size constraint when varying p can also be seen in the highlighted portions of the 77.5%-LCA and 99%-LCA cladograms around 2-3 o' clock (labelled A and B, respectively). Just underneath the high scoring large purple cluster in A are some valid clusters with a lower score, become non-valid in B due to the change in p. There are many examples of this that can be seen around the cladograms. One more is in the first cutoff band at around 6-7 o' clock. In the 55%-LCA and 77.5%-LCA cladograms, there are clearly valid clusters with HGT-score of around 1 to 3, but these vanish in the 99%-LCA cladogram. For higher p, in several cases the bands show a lower HGT-score for the same clusters when comparing to lower p. This is illustrated most clearly in the big cluster for the first and second bands, between 3-6 o' clock, where the HGT-score is the same for p values 55% and 77.5%, but lower at 99%. On the other hand, there are also several cases where the HGT-scores of clusters stay consistent across all choices of p. For example, for the cluster in the last band at between 12-1 o' clock, the HGT-score stays the same across all three cladograms. The same thing can be seen in the third band at between 5-6 o' clock. The cluster-level statistic which was chosen as "the *maximum* HGT-score within a cluster" has an effect in the observed dynamics. If high-scoring leaves are relatively rare and dispersed across clusters, the HGT-scores of clusters in this case becomes reliant on few leaves, and the HGT-score of a cluster becomes sensitive to p, or even cutoff, if that leaf leaves the cluster. In the converse case, high-scoring leaves clustering together leads to the HGT-score of such clusters to be less sensitive to p. In general, the first case will mostly be occurring, since a high HGT-score of a leaf per definition means that the leaf needs to be sufficiently different in taxonomy to other leaves in the same cluster, and there is a limited number of ways for leaves to be sufficiently different. Using the maximum HGT-score to score a cluster makes the scoring very dependent on one host. A risk is that this host originates from bad data, and a high HGT-score would thus arise from unclean data.

Bringing our focus to the cutoff parameter. It is the value of cutoff which ultimately generates the "still image" in evolution of hosts' ARG relationships in the underlying gene tree as clusters. From Figure 3.2, two examples of cutoff-induced effects are highlighted. The first effect is when large clusters with high HGT-score are subdivided into smaller clusters with lower HGT-score, and the second effect is the persistence (and non-persistence) of clusters across cutoffs. The most visible occurrence of the first effect is in the cluster between 3 and 6 o' clock, and the same effect occurs in all of the three cladograms. Between the second (cutoff 93) and third (cutoff 99) bands, this cluster gets subdivided into only two viable clusters where each receives an HGT-score of 1. When using *maximum* as the cluster-level statistic, it holds that a low HGT-score of a cluster is equal to high similarity of taxonomy among *all* hosts within that cluster. The observed effect must mean that the increase in cutoff led to a new set of clusters (which is equal to branching occurring in the underlying phylogenetic gene tree) whose internal taxonomy relation is in agreement with the gene tree. This effect thus have several explanations. One explanation is that HGT did occur between the involved hosts at a point further

back in gene-evolutionary time and those genes quickly differentiated accordantly with species evolution. Consider an opposite scenario, where the larger cluster is subdivided into smaller clusters, but instead of resulting in lower scoring clusters the sub-clusters receive a higher HGT-score. This would be greatly indicative of HGT occurrence, and additionally pinpoints the event in evolutionary time, which given an accurate enough choice of outgroup and ARGs used to build the tree can give an even more accurate estimate of recency. This scenario occurs multiple times in Figure 3.2, particularly visible in the detailed view of A. One example appears around 2-3 o' clock for the p choice of 55% and 77.5%, the latter visible in the detailed view A just below the purple cluster. In the first band, this cluster is larger and has a HGT-score of 2, and in the next band is subdivided into one smaller cluster with score 2, and a larger cluster with score 3. This cluster and its HGT-score then persists across the following cutoff bands. The same effect is observed in the next-to-last cluster at the bottom of A. A similar but less persistent cluster shows a slightly larger increase in HGT between the first and second cutoff bands before being split into non-valid clusters at subsequent cutoffs. The last visible cluster in the bottom of A also exhibits this pattern. Notably, *none* of these examples appear in B where p=99, further emphasizing that the choice of p is a crucial factor when calibrating the method for detecting HGT events.

The maximum cutoff used in Fig 3.2 is 108. Higher cutoffs than this could be used to generate valid clusters. However, there is a maximum cutoff where no more valid clusters are generated, which is dependent on the maximum cumulative branch length of the phylogenetic gene tree and choice of p. The maximum cutoff is the first cutoff in an increasing order which results in zero HGT score-valid clusters. We can be sure that no more valid clusters can be generated after this point, since cluster validity is determined by a cluster size threshold, and the cluster size monotonically decreases with increasing cutoff. The maximum cutoff varies between different phylogenetic trees, and with choice of p. This is shown in Appendix A.1 for a sample of phylogenetic trees built on resistance types including β-lactamase type A. For this resistance type, the maximum cutoff is 183. In comparison, for the phylogenetic gene tree built on *tetracycline ribosomal-protective* ARGs, the maximum cutoff is 66. For the *aph3p* resistance gene tree, the maximum cutoff was not reached since there was still around 10 valid clusters at the highest attempted cutoff of 210.

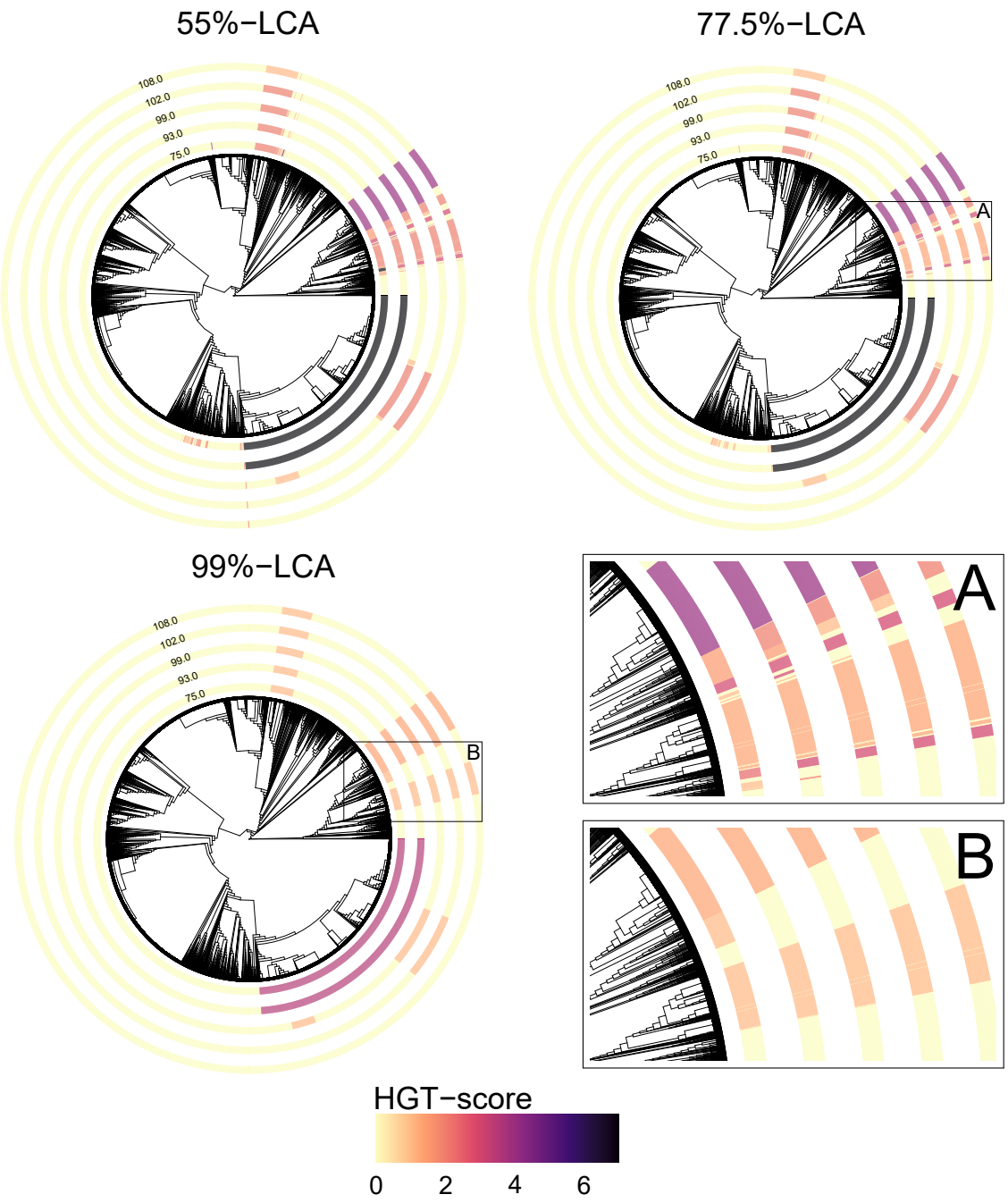


Figure 3.2: Cladograms with HGT-scores from different p values, from a phylogenetic tree built on β-lactamase class A ARGs. p values used are 55%, 77.5% and 99%. HGT-scores are shown as colors on the five circular bands (legend at the bottom). The bands in turn represent clusters resulting from cutoffs in increasing order, radially: 75, 93, 99, 102 and 108. **A** and **B**: detail from the 77.5%-LCA and 99%-LCA cladograms.

3.1.2 Validation of the HGT-score method

Validation of the HGT score method was performed by applying it to a set of ARGs with differences in HGT prevalence, without informing the method itself of any such label. The ARGs prone to HGT produced a higher average HGT score compared to the non-HGT ARGs *and* compared to when random labels were used, which indicates that a HGT signal is effectively captured by the HGT-score algorithm.

To serve as validation data, labelled beta-lactamase ARGs from the BLDB database [14] were used. In BLDB, ARGs are classified into two distinct labels: Acquired (A), where the gene has previously been established as readily transferable through HGT, and Natural (N), indicating a gene known to exhibit lower horizontal transferability. The fARGene-predicted beta-lactamase ARGs were BLAST-aligned with the BLDB genes and inherited the labels (A or N) on a 95% identical basis. Clusters containing at least one Acquired-labelled resistance gene were in turn labelled Acquired, and the rest of the clusters were labelled Natural.

The HGT-score algorithm was run separately on these two sets of clusters, using 3 different p-LCA values: 55%, 77.5%, and 99%. The results of HGT-scoring is shown in Figure 3.3 as a dot chart of HGT-score cluster means on the y-axis against tree cutoff on the x-axis. The HGT-score cluster means is the average HGT-score taken across all valid clusters which were produced by the corresponding tree cutoff on the x-axis. In the chart, we see three classes of means: *Acquired cluster means* (dark blue dots), *Natural cluster means* (orange dots), and *random Acquired cluster means* (smaller, light blue dots). The Acquired cluster means display a higher HGT-score compared to the Natural cluster means, for the majority of tree cutoffs. There are some exceptions in the cutoff extremities at cutoff 12.0 and at cutoffs 165.0 to 174.0 for the 55%-LCA and 77.5%-LCA charts, where the Natural cluster means are higher the Acquired cluster means. In the 99%-LCA chart, Natural cluster means are higher compared to the Acquired cluster means at cutoffs 33.0 to 63.0 (except at 57.0).

In these results we see that Acquired ARG host clusters are scored higher than Natural ARG host clusters. By additionally comparing the HGT-score of the true Acquired clusters to randomly assigned Acquired clusters, additional evidence is given to show that the algorithm successfully signals HGT events. To give an indication of the *signal to noise* that the HGT-score produces in this case, the third set of smaller points in light blue in Figure 3.3 represents HGT-score means of Acquired clusters where the cluster labels were set completely at random, with averages taken over 150 randomizations. Visual inspection shows that the HGT-score for real Acquired means is consistently higher in comparison to random means, across most cutoffs and all p%-LCA values. There is one exception, at cutoff 9.0 for the 55%-LCA and 77.5%-LCA graphs. For the Acquired cluster means, a visible pattern is distinct peaks in HGT-scores, including a peak between cutoffs 78.0 and 99.0 across all three values of p. Other peaks are also seemingly conserved across p, particularly at cutoffs 18.0, 45.0, 105.0, 108.0 and 117.0.

While the value of HGT-scores in general follow a decreasing trend with increasing p-LCA values, the relation between the Acquired and Natural HGT-score averages stays mostly the same. There are some clear exceptions, mostly outside of the 24.0 to 72.0 cut-off range. At increasing values of p, the number of valid HGT-score means for both the Acquired and Natural-labelled points become fewer, due to that the clusters need to be larger to be assigned a valid HGT-score. A $p = 0.55$ sets a minimum cluster size of $\frac{1}{1-0.55} = 2 \approx 3$ rounded up, whereas $p = 0.99$ sets it to $\frac{1}{1-0.99} = 100$. There are however differences in valid cluster-dynamics between the two labelled sets as well. Namely, the number of Natural-labelled points become fewer in comparison with the number of Acquired-labelled points, with increasing p.

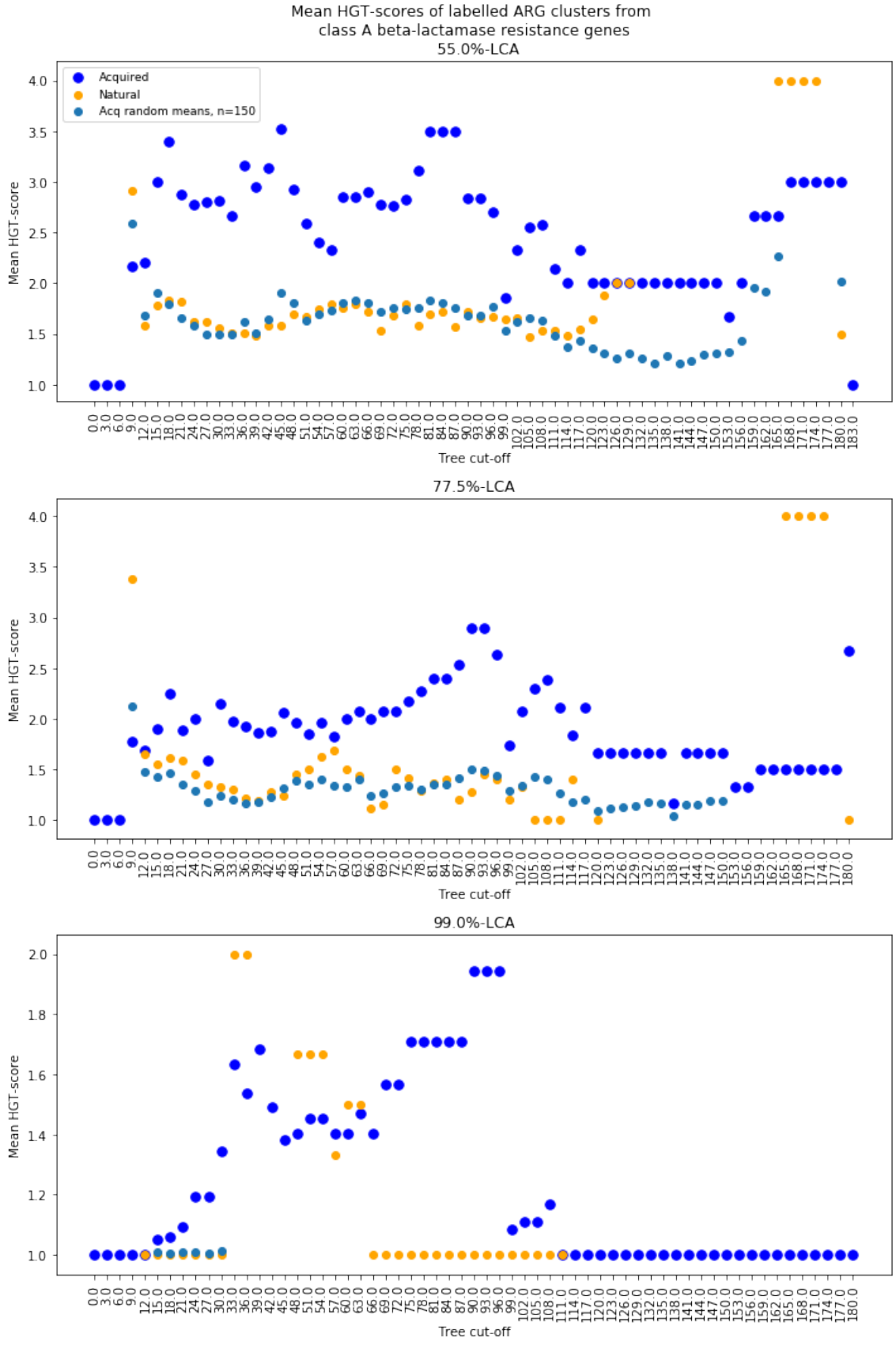


Figure 3.3: Dot chart of mean HGT-scores at tree cutoffs from 0 to 183. Dots are colored according to their validation label Acquired (orange)/Natural (blue) class A beta-lactamase resistance gene clusters. The mean HGT-score is the average HGT-score across clusters of size greater than $\frac{1}{1-p}$ for the corresponding tree cutoff. Each graph resulted from HGT-scoring on different p-LCA values, from 55%-LCA up to 99%-LCA.

3.2 HGT-score algorithm on unlabelled class A beta-lactamase resistance genes

Screening of predicted antibiotic resistance genes from fARGene was done. In this case, as opposed to the labelled dataset, no assumptions about gene mobility are made. The average HGT-scores across all valid clusters for seven different %-CA levels, $0.550, 0.625, 0.700, 0.775, 0.850, 0.925$ and 0.990, and for cutoffs $0.0, 3.0, 6.0, ..., 210.0$, settings the same as for the validation case, were calculated.

3.2.1 Evaluating the HGT-score method through cross-validation of literature and BLAST

The HGT-score method was applied on unlabelled data with gene trees built for specific resistance types. To mirror a realistic use case of the method, a filtering step identified high-scoring clusters for further analysis by identifying the host organisms along with their original sample in the NCBI GenBank database. A brief literature review and further analysis of high-scoring clusters revealed hosts likely involved in recent HGT of β-lactamase ARGs. The analysis also revealed, in some samples, low confidence in the taxonomic species evaluation.

A small *Python* script was written to perform a screening of resistance type β-lactamase class A, by outputting clusters with a HGT-score (using p=77.5% and cluster-level statistic as maximum), above a threshold here chosen as 3.0. This corresponds to selecting clusters with at least one host whose taxonomy at the minimum shared family taxon with 77.5% of the other hosts. At a cutoff of 174.0, which is close to the maximum non-redundant cutoff of 183, one output cluster had a HGT-score of 4.0, signifying order-level taxon differences. See output in Figure 3.4. Apart from mostly containing unspecified *Prevotella*-genus hosts (count of 8), the cluster also contained one host of *Prevotella copri*, and one host of species *Bacteroides ovatus*, where the latter is the host which resulted in the cluster's 4.0 HGT-score.

```
Tree cutoff: 174.0
Clusters with score above 3.0
----- Cluster 1 // Max HGT-score 4.0
species: Bacteroides ovatus
 count: 1
species: Prevotella sp.
 count: 8
species: Prevotella copri CAG:164
 count: 1
```

Figure 3.4: Sample output of cluster contents and HGT-score information. Generated from screening all clusters per cutoff of the phylogenetic gene tree built on β-lactamase class A ARGs. A threshold of 3.0 displays clusters of only higher HGT-score.

Metadata about species label confidence was gathered for the originating sample for

each host in the cluster, summarized in Table 3.1. 3 of the 10 hosts originated from samples where their taxonomy ware deemed inconclusive, due to either extensive contamination or otherwise poor data. There are several cases of poor data where full determination of taxonomy is impossible. However, some cases allow identification up to the *genus* level even if the host *species* remains undetermined (for example, the several generic *Prevotella sp.* in Table 3.1). In the context of the HGT-score, such hosts can still be included if the primary goal is to identify ARG transfers across broader taxonomic distances. Given this, taxonomy is a central part of the HGT-score method, so to include hosts in the dataset whose taxonomy is uncertain to a high extent in turn puts the accuracy of cluster HGT-scores at risk. Further data cleaning steps using metadata such as taxonomy confidence is thus needed upstream in the full HGT-method.

Table 3.1: β-lactamase ARG host species identified in the cluster presented in Fig 3.4. The *taxonomy check* column estimates the accuracy of species identification, fetched as metadata from the original NCBI GenBank sample entry. *Sample accession ids* links to each corresponding sample, for reference.

Species	**Taxonomy check**	**Sample accession id**
Bacteroides ovatus	OK	GCA_00178275.1
Prevotella sp.	OK	GCA_019409215
Prevotella sp.	Inconclusive	GCA_019409625.1
Prevotella sp.	Inconclusive	GCA_019409885.1
Prevotella copri	OK	GCA_000435255.1
Prevotella sp.	Inconclusive	GCA_019409665.1
Prevotella sp.	OK	GCA_019409535.1
Prevotella sp.	OK	GCA_019409605.1
Prevotella sp.	OK	GCA_019408965.1
Prevotella sp.	OK	GCA_019409095.1

Only considering hosts which passed the taxonomy check, the cluster contains 6 hosts of genus *Prevotella* (5 unidentified species and 1 *Prevotella copri*) and one host of species *Bacteroides ovatus*. The cluster was found at a particularly high cutoff value, which means that their β-lactamase resistance genes are of very high evolutionary similarity, which in turn indicates possible recent HGT.

Further broadening the view to a biological context, the bacterial species of the cluster typically coexist in the human gastrointestinal tract [23] [24], creating frequent opportunities for interaction, which would increase the likelihood for HGT. These bacteria are commensal in humans and typically non-pathogenic in their natural environment. Even though these species themselves are normally benign, the HGT of ARGs still poses a clinical risk by creating a reservoir of resistance that could be exploited if an opportunistic pathogen emerges. When HGT spreads ARGs across diverse taxonomic groups, the potency of this antibiotic resistance gene reservoir would be enhanced, not only since more individual hosts equals more opportunities for HGT, but also because the taxonomic range of host species that harbor these

genes becomes larger. This expanded host diversity makes subsequent ARG transfers between easier, since the ARGs now have more opportunities to move between more closely related hosts.

Focusing again on the individual hosts in the selected cluster in Table 3.1, we gain more insight into the feasibility of whether HGT might have occurred between hosts in the cluster or not. A short scientific literature review provides context that β-lactamase type resistance is readily found in those species. In one 2012 Canada screening study, 47.6% of *B. ovatus* strains (30/63) were multidrug resistant to β-lactam antibiotics. In the past, some β-lactamase class A ARGs have been identified as likely being transferred by HGT by *Prevotella spp.* and *Bacteroides spp.*, one of them being the *cfxA* gene [25]. Upstream in the HGT-score method, fARGene [1] is used to predict ARGs from the genomes of sequenced bacteria from NCBI. A BLAST [21] analysis of the two β-lactamase ARGs predicted in the Bacteroides ovatus host within our cluster (Table 3.1) reveals that one gene shares 99.3% sequence identity with a cfxA gene originating from a Prevotella sp. bacterium. The other gene shows 99.7% identity with a β-lactamase ARG derived from a Bacteroides sp. bacterium. Complete BLAST results are provided in Appendix A.2. These findings suggest a potential HGT event in which the cfxA gene may have been transferred from a Prevotella bacterium within the cluster to the B. ovatus host. Since a high cutoff was used to obtain the cluster, it further suggests that this HGT event was recent.

4

Discussion

I will use this chapter to reason about strengths of the HGT-score method, inherent weaknesses, suggest areas of highest potential improvement, and discuss its usability.

4.1 Alternatives to involve taxonomy of hosts to enrich the HGT-score method

The HGT-score method uses the NCBI taxonomy metadata to assign taxonomies to ARG hosts, in order to generate a HGT-score for each cluster. An alternate approach would be to use a known well-conserved genetic region in the hosts' genomes to construct a evolution gene tree, and infer HGT by finding disagreements in the evolution gene tree compared to the ARG gene tree. In theory, this could give a more accurate measure of taxonomic distance compared to the approach used in the HGT-score algorithm developed in this thesis. In turn, there would be a difference in HGT-score output. Now, the HGT-score per host is a discrete integer value between 1 and 7, or 0 if the cluster of the host is invalid. In comparison, using a species tree instead of taxonomy labels would allow for a more continuous HGT-score, but it would probably be more difficult to interpret. In this sense, using taxonomic rank differences as basis for HGT-scoring makes for a result which is fairly straightforward to interpret. In the end, it would likely be possible to calibrate the method so that a "continuous" HGT-score could be interpreted as well. For example, by using data with known HGT and taxonomic rank differences. Compared to the method used in this thesis, the alternative method would also require more phylogenetic trees to be constructed, which would increase computational complexity, which in turn could lead to increased processing time.

Integrating environmental or ecological niche metadata from the hosts is another way to enrich the results. The primary use for this would be filtering out candidate high-scoring clusters. The source data and the HGT-score method itself are not perfect, and false positives of direct HGT will be produced, in this context manifesting itself through clusters having a high HGT-score despite no HGT event taking place. Here, an extra plausibility filter such as excluding a cluster, or a specific host within in a cluster where it is improbable that the hosts have been in the same environment, would reduce noise and thus give the HGT-score results a higher resolution. This approach would probably have the greatest effect in metagenomic sequencing where all samples have clear spatial or environmental labels.

4.2 The HGT-method using taxonomy - gene phylogeny disagreement produces a blindspot

Perhaps the most blatant overall limitation of the developed HGT-score method is its inability to signal for single HGT-events between closely related hosts. This arises from what the method is also based on: using disagreement between the taxonomy of the ARG host and its gene tree cluster neighbors. By the HGT-score method definition, hosts inside a cluster contain genetically similar ARGs (where the level of similarity increases with cutoff), where high HGT-scores are given to clusters with a higher degree of difference in taxonomy ranks. In the opposite case, low HGT-scores occur when the taxonomical similarity is high. If the ARG:s transferance journey between hosts ends after only one HGT event between hosts of the same species, it will go under the radar of the HGT-score method. However, the method would be resilient to successive HGT events which carry an ARG between a series of similar hosts of *cumulatively* increasing taxonomic distance, given that the ARG sequence remains largely the same. For example, consider HGT between hosts A, B, C, D and E, where A is similar to B in taxonomy, B is similar to C, and so on, but A is not similar to E. In a perfect scenario, hosts A and E (along with the other hosts) would end up in the same cluster, and depending on the choice of HGT-score parameters, this would be a high-scoring cluster.

I want to point out that even in the case of single HGT events between hosts, the blindspot does not undermine the value of the overall method. HGT of ARGs between more distantly related species is of high interest to be discovered, which is what the HGT-score method accomplishes. An user should however be aware of this limitation, use other methods or complementary information for detection of HGT between closely related species.

4.3 Use of HGT-score method

The scenarios in which the HGT-score method can be used are all after HGT has occurred making it a tool for retrospective detection and not for prescriptive analysis. The choice of parameters can make it more or less sensitive, and the choices should be made according to the use case. I think that the method has the most potential in a (close to) real-time monitoring use case, where sequenced bacteria genomes are input via some routine sampling process. For this use case, the HGT-score method is missing the recalculation of the phylogenetic gene tree, when new hosts are added or removed from the analysis. Computationally efficient algorithms which preserve evolutionary integrity such as EPA [26] for addition and FastTree rebalancing [2] for removal could be integrated here. Another approach is to simply remake trees with each batch of samples. In a hospital, wastewater treatment facility, or a farm, surveillance could be set up to signal for when HGT of ARGs is detected using the method. When this happens, as we saw in Section 3.2.1, even more actionable and useful insight was gained of the biological context of high scoring clusters using supplementary analysis. Extending the automatic pipeline with this step in particularly

interesting cases would lower the barriers to gain HGT insights more quickly.

Clearly relating cutoff to recent HGT would also increase usability in real use cases. Two steps to do that would be to: 1. Take more care into choosing a known good outgroup. A better outgroup gives a more accurate direction of evolution in the phylogenetic gene tree, and results in a better ”biological clock”. 2. Normalize the possible cutoffs in the phylogenetic tree. A naïve approach is to use the largest nonredundant cutoff as a maximum cutoff, and 0 as a minimum. This increases interpretability between different gene trees. For example, 0 being minimum cutoff and 1 being maximum, the cutoff ”0.99” would mean roughly the same thing across phylogenetic gene trees.

4.4 Next steps

4.4.1 Further validation on different types of data

Validation of the HGT-score method provided evidence that it does detect HGT events. This was done using one dataset of ARGs that are known to easily transfer with HGT, and those not. To increase the confidence in the method, more such validations should be made, particularly with ARGs from resistance types other than β-lactamase class A. Further, while the validation data used contained ARGs which generally show high horizontal transferability, it is not a guarantee that HGT would happen in select hosts. To be more confident about the HGT-score method validation, one would have to use validation data that we are also entirely sure of. The most effective way to accomplish this is to simulate HGT in a dataset through data synthesis. Another dataset of hosts with no HGT occurrence could similarly be generated. In a biological setup, similar data could be produced by sequencing specific hosts which are known from experiments to have partook in HGT, and vice versa.

4.4.2 Calibration of p

The parameter p should be further calibrated to different scenarios. In particular, the sensitivity of results based on p should be methodically assessed with validation data, preferably simulated data. This allows us to remove uncertainty about the underlying data and in a controlled manner learn more about which values of p are better, and for which cases. From the results, it seems that a p of around 0.775 (77.5%) gives a tradeoff between specificity and generality, and which generates actionable results in HGT-score. In the extremes, a p of 0.55 led to higher HGT-scores, but not much higher than 0.775. However a p of 0.99 quite clearly led to the HGT-score signal being lost in some cases, due to the cluster size requirement becoming too large.

4.4.3 Efficient setting of cutoffs to pinpoint HGT events in evolutionary time

It was established in Section 3.1.1 that one of the most telling signs of HGT occurrence is if one cluster is broken down into two clusters, and either of the resulting clusters receive a higher HGT-score after the splitting. If cutoffs are chosen too widely apart, in the worst case these events could be missed. To nail down HGT-events even more, better choices of cutoffs can help. One approach to minimize the gap is to simply choose the smallest possible distance between each cutoff. The downside is that it would increase the computational work needed. Another approach is to choose cutoffs recursively based on observed cluster subdivision. For example, initially use wide spaces between cutoffs for any gene tree, say jumps of 30: 0, 30, 60, 90 and so on. After observing that between cutoff 90 and 120, the number of valid clusters increased from 50 to 100, we will want to increase the resolution in that area and generate cutoffs with smaller distance apart. Then we generate clusters for cutoffs with jumps of 5: 95, 100, 105 and so on, and notice that between cutoffs 95 and 100 is where the action happens. This goes on until we reach a level of resolution where we are satisfied. This approach would speed up the process of generating cutoffs significantly, and makes the method more scalable if it would be used with new ARG gene trees often.

4.4.4 Upstream data cleaning and data flexibility

In some observed cases, clusters contained hosts whose sequenced genomes had been disqualified from NCBI due to not been qualitative enough. This means that the results are "contaminated" to an extent which is not completely known. Data cleaning steps were included in the method, by removing hosts with incomplete taxonomy data and by using a minimum threshold in sequence alignment between fARGene results and β-lactamase ARGs. An extra data cleaning step would use the host metadata from NCBI or other sources to qualify or disqualify those hosts from continuing to downstream HGT-scoring. The method could thus be used with even higher confidence in results.

The data cleaning step of removing hosts without full taxonomy (e.g., undetermined species) could be optional in cases where the user is looking for , allowing for greater flexibility. When ARG transfer occurs between two hosts with a taxonomic distance beyond the genus level, knowledge of the exact host species is unnecessary, as long as the genera are known.

5
Conclusion

Validation of the HGT-score method using ARGs known to readily transfer via HGT and those that do not suggests that, on average, the method successfully identifies HGT events. There are a multitude of directions to explore further to improve the method. Selection and application of three critical parameters in the method - p, cluster-level statistic, and cutoff - should be further tested and calibrated to enhance reliability for real-world use. This calibration should be attempted on simulated or known genomes with the goal of directly assessing the effect of the parameters by comparing them with the expected outcome. The robustness and interpretability of the HGT-method output can further be enhanced by refining the upstream data cleaning.

Due to its relative computational simplicity by using preprocessed gene trees and simple taxon labels in the HGT-score calculation step, the HGT-score method shows promise in real-time applications. It could function as a descriptive diagnostic tool alerting for HGT occurrence, for example in a water treatment facility.

Bibliography

[1] Berglund F, Österlund T, Boulund F, Marathe NP, Larsson DGJ, Kristiansson E. Identification and reconstruction of novel antibiotic resistance genes from metagenomes. Microbiome. 2019;7(1):1–14.

[2] Price MN, Dehal PS, Arkin AP. FastTree 2 – Approximately Maximum-Likelihood Trees for Large Alignments. PLOS ONE. 2010;5(3):1–10. Available from: https://doi.org/10.1371/journal.pone.0009490.

[3] Sherrill-Mix S. taxonomizr: Functions to Work with NCBI Accessions and Taxonomy; 2022. R package version 0.9.3. Available from: https://CRAN.R-project.org/package=taxonomizr.

[4] Murray CJ, Ikuta KS, Sharara F, Swetschinski L, Robles Aguilar G, Gray A, et al. Global burden of bacterial antimicrobial resistance in 2019: a systematic analysis. The Lancet. 2022;399(10325):629–655.

[5] WHO;.

[6] Soucy SM, Huang J, Gogarten JP. Horizontal gene transfer: Building the web of life. Nature Reviews Genetics. 2015;16(8):472–482. Available from: http://dx.doi.org/10.1038/nrg3962.

[7] Andam CP, Gogarten JP. Biased gene transfer in microbial evolution. Nature Reviews Microbiology. 2011;9(7):543–555.

[8] Shoemaker NB, Vlamakis H, Hayes K, Salyers AA. Evidence for extensive resistance gene transfer among Bacteroides spp. and among Bacteroides and other genera in the human colon. Applied and Environmental Microbiology. 2001;67(2):561–568.

[9] Bengtsson-Palme J, Kristiansson E, Larsson DGJ. Environmental factors influencing the development and spread of antibiotic resistance. FEMS Microbiology Reviews. 2018;42(1):68–80.

[10] Prüfer K, Munch K, Hellmann I, Akagi K, Miller JR, Walenz B, et al. The bonobo genome compared with the chimpanzee and human genomes. Nature. 2012;486(7404):527–531.

[11] Widespread of horizontal gene transfer in the human genome. BMC Genomics. 2017;18(1):1–11.

[12] What is FASTA format?;. Available from: https://zhanggroup.org/FASTA/.

[13] Clark K, Karsch-Mizrachi I, Lipman DJ, Ostell J, Sayers EW. GenBank. Nucleic Acids Research. 2016;44(D1):D67–D72.

[14] Naas T, Oueslati S, Bonnin RA, Dabos ML, Zavala A, Dortet L, et al. Beta-lactamase database (BLDB)–structure and function. Journal of Enzyme Inhi-

bition and Medicinal Chemistry. 2017;32(1):917–919. Available from: https://doi.org/10.1080/14756366.2017.1344235.

[15] Bush K, Jacoby GA. Updated functional classification of β-lactamases. Antimicrobial Agents and Chemotherapy. 2010;54(3):969–976.

[16] Katoh K, Standley DM. MAFFT multiple sequence alignment software version 7: Improvements in performance and usability. Molecular Biology and Evolution. 2013;30(4):772–780.

[17] The Newick tree format;. Available from: https://evolution.genetics.washington.edu/phylip/newicktree.html.

[18] Cock PJA, Antao T, Chang JT, Chapman BA, Cox CJ, Dalke A, et al. Biopython: Freely available Python tools for computational molecular biology and bioinformatics. Bioinformatics. 2009;25(11):1422–1423.

[19] Huerta-Cepas J, Serra F, Bork P. ETE 3: Reconstruction, Analysis, and Visualization of Phylogenomic Data. Molecular Biology and Evolution. 2016;33(6):1635–1638.

[20] Van Hoek AHAM, Mevius D, Guerra B, Mullany P, Roberts AP, Aarts HJM. Acquired antibiotic resistance genes: An overview. Frontiers in Microbiology. 2011;2(SEP):1–27.

[21] Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ. Basic local alignment search tool. Journal of Molecular Biology. 1990;215(3):403–410.

[22] Camacho C, Coulouris G, Avagyan V, Ma N, Papadopoulos J, Bealer K, et al. BLAST+: Architecture and applications. BMC Bioinformatics. 2009;10:1–9.

[23] Abdelsalam NA, Hegazy SM, Aziz RK. The curious case of Prevotella copri. Gut microbes. 2023 dec;15(2):2249152.

[24] Fultz R, Ticer T, Ihekweazu FD, Horvath TD, Haidacher SJ, Hoch KM, et al. Unraveling the Metabolic Requirements of the Gut Commensal Bacteroides ovatus. Frontiers in microbiology. 2021;12:745469.

[25] García N, Gutiérrez G, Lorenzo M, García JE, Píriz S, Quesada A. Genetic determinants for cfxA expression in Bacteroides strains isolated from human infections. Journal of Antimicrobial Chemotherapy. 2008 09;62(5):942–947. Available from: https://doi.org/10.1093/jac/dkn347.

[26] Berger SA, Krompass D, Stamatakis A. Performance, accuracy, and Web server for evolutionary placement of short sequence reads under maximum likelihood. Systematic biology. 2011 may;60(3):291–302.

A

Appendix

A.1 fARGene options

When predicting ARGs, fARGene was used on genomic data from NCBI GenBank. The tool was run using the following settings, per fasta file and per HMM:

```
fargene -i [fasta filepath].fna --hmm-model [hmm filename].hmm
    --score [threshold] -sp --force
    --rerun --amino-dir [tmp path]
```

A.2 Predicted β-lactamase class A ORFs from the host GCA_00178275.1 and corresponding BLAST results

The *Bacteroides ovatus* host with accession accession id GCA_001178275.1 was used with fARGene and had the output of 2 predicted ARGs when utilizing a β-lactamase class A HMM

```
>orfs-translated\_ADMO01000015.1\_seq1\_1
MRSFIVFLCLVPTLLFARQTQLETQLKEAIKGKKAEIGIAVIIDGKD
TITVNNDIPYPLMSVFKFHQALALADYMGKQRQSLETRLPIKKSDLK
PDTYSPLRDKYPQGGIEMSIADLLKYTLQQSDNNACDILFDYQGGPD
AVNKYIHSLGIRECAIAGTETAMHEDLNLCYENWTTPLAAAELVEIF
RKKPLFPKVYKDFIFQTMVECQTGQDRLVAPLLDKKVTVGHKTGTGD
LNAKGQQIGCNDIGFVLLPGGRTYSIAVFVKDSEENNQANNKIIADI
SRIVYEYVMQH*

>orfs-translated\_ADMO01000001.1\_seq1\_1
VFSLSHKSATKGSANPPLTDVLTDSISQIVSACPGEIGVAVIINNTD
TVSVNNKSIYPMMSVFKVHQALALCNDFDKKGLSLDTLVKINREKLD
PKTWSPMMKDYSAPVISLTVRDLLRYTLSQSDNNASNIMFKNMLNTA
QTDSFIAKLIPHSSFQIAYTEEEMSADHDKAYSNYTSPLGAAMLMNR
LFTESLISNEKQDFIKNALKECKTGIDRIVAPLLDKEGVVIAHKTGS
GDVNENGILAAQNDVAYICLPNKVCYTLAVFVKDFKGNESQASQFVA
HISAVVYSLLINTALN*
```

Both genes were BLAST:ed using the UniProt tool, yielding matches with 99.7% and 99.3% identity, respectively. The first match was a β-lactamase class A gene known as *BSGG_1114* with predicted origins from a *Bacteroides sp.* bacterium. The second match was a β-lactamase class A gene known as *cfxA* with predicted origins from a *Prevotella sp.* bacterium.

```
>tr|E5C4M8|E5C4M8_9BACE Beta-lactamase class A catalytic domain-
   containing protein OS=Bacteroides sp. D2 OX=556259 GN=BSGG_1114 PE
   =4 SV=2
MRSFIVFLCLIPTLLFARQTQLETQLKEAIKGKKAEIGIAVIIDGKDTVTVNNDIHYPLM
SVFKFHQALALADYMGKQKQSLETRLPIKKSDLKPDTYSPLRDKYPQGGIEMSIADLLRY
TLQQSDNNACDILFNYQGGPDAVNKYIHSLGIRECAIVGTETAMHEDLNLCYENWTTPLA
AAELVEIFRKKPLFPKVYKDFIFQTMVECQTGQDRLVAPLLDKKVTVGHKTGTGDLNAKG
QQIGCNDIGFVLLPGGRTYSIAVFVKDSEENNQANSKIIADISRIVYEYVMQH

>tr|A0A374T6L6|A0A374T6L6_9BACT Beta-lactamase OS=Prevotella sp. TF12
   -30 OX=2292365 GN=cfxA PE=3 SV=1
MKKNRKKQIVVLCIALVCIFILVFSLSHKSATKGSANPPLTDVLTDSISQIVSACPGEIG
VAVIINNTDTVSVNNKSIYPMMSVFKVHQALALCNDFDKKGLSLDTLVKINREKLDPKTW
SPMMKDYSAPVISLTVRDLLRYTLSQSDNNASNIMFKNMLNTAQTDSFIAKLIPRSSFQI
AYTEEEMSADHDKAYSNYTSPLGAAMLMNRLFTESLISNEKQDFIKNALKECKTGIDRIV
APLLDKEGVVIAHKTGSGNVNENGILAAQNDVAYICLPNKVCYTLAVFVKDFKGNESQAS
QFVAHISAVVYSLLINTALN
```

A.3 Cluster count behavior

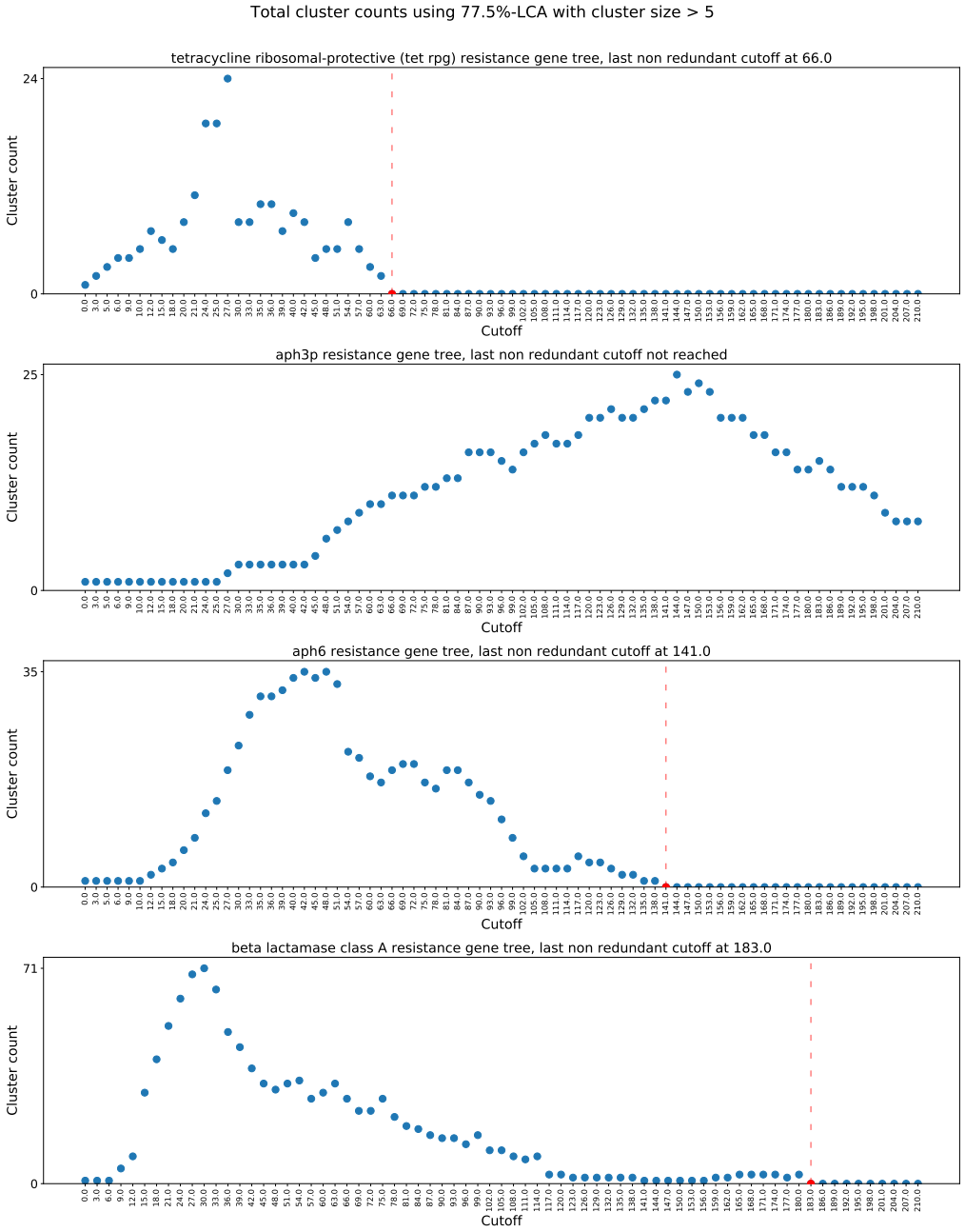


Figure A.1: Count of HGT score-valid clusters plotted against cutoff values for four chosen ARG trees: tet rpg, aph3p, aph6 and β-lactamase classA. LCA of 77.5% was used, so clusters of size greater than 5 were included in the count. The cutoff for each tree which produced a valid cluster count of 0 is indicated with a red dot and a dashed vertical line. Concurrent cutoffs after the line cannot yield valid clusters.

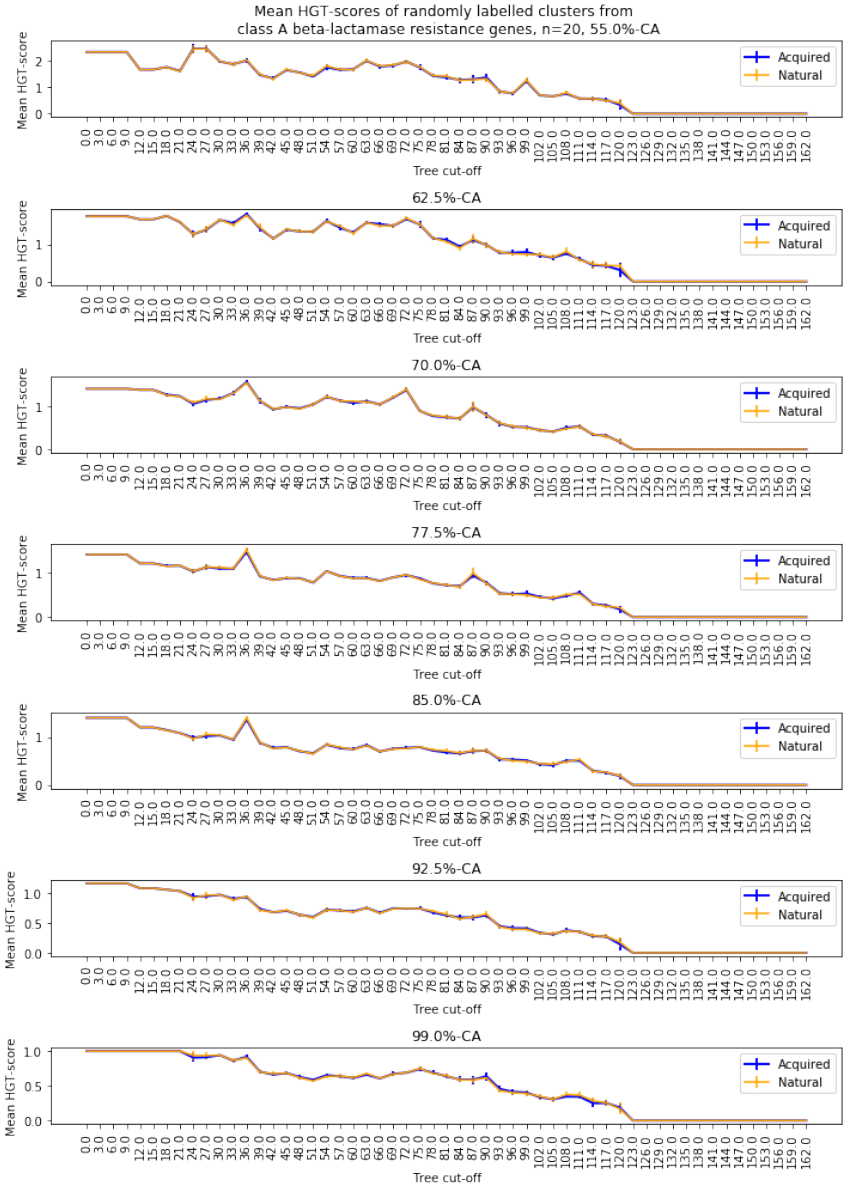


Figure A.2: Mean HGT-scores (n=20) at different tree cut-offs of randomly labelled (Acquired/Natural) class A beta-lactamase resistance genes. There are seven subplots in total, each subplot resulting from HGT-scoring on different %-CA values, from 55%-CA up to 99%-CA. Error bars representing one standard deviation at the points at each tree cut-off are shown in the same color as the points in question.

DEPARTMENT OF MATHEMATICAL SCIENCES
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY