UNIVERSIDADE DE LISBOA
FACULDADE DE CIÊNCIAS
DEPARTAMENTO DE BIOLOGIA

Bioinformatics toolbox for comparative clustering evaluation of Whole-Genome Sequencing (WGS) pipelines for bacteria routine surveillance

Joana Vanessa Gomes Pereira

Mestrado em Bioinformática e Biologia Computacional

Dissertação orientada por:
Doutora Verónica de Pinho Mixão
Professor Doutor Francisco José Moreira Couto

2025

Acknowledgements

The completion of this master's thesis would not have been possible without the support of several people and institutions who have accompanied me over the past three years. To all of them, I express my deepest gratitude.

First and foremost, I would like to thank my supervisor, Dr. Verónica Mixão, and Dr. Vítor Borges, for giving me the opportunity to join the research fellowship within the Centaur project, funded by the ISIDORe project (European Union's Horizon Europe Research & Innovation Programme, Grant Agreement no. 101046133). Thanks to this opportunity, I was also able to carry out this master's thesis at the Bioinformatics and Genomics Unit of the National Institute of Health Dr. Ricardo Jorge. I am equally grateful for the chance to participate in the 14th International Meeting on Microbial Epidemiological Markers (IMMEM XIV, Conference 2025), to present a poster, and to continue being part of the team over the coming months. Moreover, I thank them not only for their scientific guidance, but also for their constant encouragement, patience, and availability to clarify doubts, helping me to overcome the challenges I encountered along the way.

I would also like to thank my colleagues at the Unit for the warm way in which they welcomed me into the team, for their constructive suggestions during lab meetings, and for sharing their knowledge.

I am grateful to Professor Francisco Couto (Faculty of Sciences, University of Lisbon) for his suggestions, which contributed to my learning process. To my colleague Bruno, I am thankful for his help and encouragement in my first steps in bioinformatics. To my friend Inês, I am grateful for all her scientific advice.

I would also like to acknowledge the contribution of other professors who supported my personal development throughout this journey – Eduarda, Ana, André, Rodolfo, Isabel, and Dr. Dina.

Finally, I owe a very special thank you to my family and friends, especially to my sister, for her unconditional support and for always being there in the most demanding moments.

Abstract

Whole-Genome Sequencing (WGS) provides higher resolution than traditional typing to distinguish closely related isolates. As result, disease surveillance increasingly adopts WGS, with international agencies recommending its use in reference laboratories. However, the heterogeneity of workflows and unequal resources raise concerns about inter-laboratory result comparability and, consequently, data sharing and communication.

To address these issues, this thesis project developed EvalTree, a Python-based command-line tool to compare clustering results from two typing solutions, including traditional and genome-scale approaches, assessing their congruence at all possible resolution levels. EvalTree accepts two input folders or clustering files, processes them, and produces multiple outputs, including an user-friendly HTML report. When a folder generated by ReporTree, a tool to identify genetic clusters at all possible distance thresholds, is provided as input, EvalTree enables not only the inter-pipeline clustering comparison, but also detection of stable clustering regions, cluster characterization using metadata, and assessment of outbreak signal overlap.

EvalTree was validated and benchmarked using a large (2946 isolates) and diverse dataset of *Salmonella enterica*, showing it accurately reproduces a recently published large-scale evaluation of inter-pipeline congruence at the European level. Its running time was mainly affected by dataset diversity rather than size. To further demonstrate its applicability, EvalTree supported the implementation of the *S. enterica* genomic surveillance pipeline at the Portuguese National Institute of Health (INSA), by comparing its performance with that of the European Food Safety Authority (EFSA), revealing high cluster congruence and similar resolution power.

In summary, EvalTree is a novel bioinformatics tool (available through conda installation) that offers a practical, flexible solution to evaluate cluster congruence between the pipelines of different laboratories, supporting inter-laboratory communication in a One Health framework. It also promotes the long-term sustainability of any pipeline by enabling informed decision-making throughout its life-cycle (e.g., evaluating software updates).

Keywords: EvalTree, Clustering Congruence, Genomic Surveillance, Whole-Genome Sequencing, Outbreaks

Resumo

As doenças infecciosas são um grave problema de saúde pública, afetando milhões de pessoas anualmente. Portanto, o desenvolvimento e implementação de sistemas de vigilância que permitam rastrear e monitorizar os patógenos é essencial. Com as novas tecnologias de sequenciação os laboratórios têm estado a transitar dos métodos tradicionais, como a serotipagem e *Multi Locus Sequence Type (MLST)*, para a nova abordagem de *Whole-Genome Sequencing (WGS)*. Esta nova tecnologia tem maior poder de discriminação de patógenos quando comparada com os métodos traditionais. As abordagens mais comuns de análise de dados de *WGS* e diferenciação de isolados geneticamente muito próximos no âmbito da vigilância genómica de bactérias são o *core-genome* e *whole-genome Multilocus Sequence Type (cg/wgMLST)* ou *Single-Nucleotide Polymorphisms (SNPs)*.

O *cgMLST* baseia-se num conjunto de genes *core* de uma espécie bacteriana, enquanto o *wgMLST*, além do conjunto de genes *core*, contém genes acessórios, i.e. somente presentes em algumas estirpes da espécie. Estes métodos são bastante úteis e eficientes para a uma vigilância de rotina, para construção de bases de dados com perfis alélicos e para a identificação de *clusters* genéticos potencialmente associados a surtos. Por sua vez, a abordagem de *SNPs*, a qual se baseia na diferenciação de genomas ao nível do nucleótido usando um genoma de referência como comparação, é mais útil na análise de isolados geneticamente muito próximos, sendo frequentemente usada para confirmação de surtos determinados por *cgMLST*.

Na Europa, embora existam orientações para a implementação de sistemas de vigilância baseados em *WGS* num quadro de *One Health*, o qual considera a interconexão entre animais, humanos e ambiente, os países estão em diferentes ritmos de adopção de *WGS* e a seguir caminhos diferentes na escolha das *pipelines* bioinformáticas para análise destes dados. Por exemplo, enquanto alguns países se encontram a desenvolver os seus próprios *workflows* de análise de dados de *WGS*, outros optam por soluções comerciais ou por soluções *Open Source* já disponíveis no GitHub. Esta diversidade de *pipelines* implementadas levanta algumas questões, nomeadamente, ao nível da comparabilidade de resultados de *clustering* gerados por diferentes laboratórios e as consequências disto na sua comunicação e partilha de dados. Neste contexto, é fundamental o desenvolvimento de ferramentas que permitam avaliar a comparabilidade de resultados entre os diferentes países e, consequentemente, apoiar a sua comunicação. Assim, este projeto de tese teve como objetivo desenvolver o EvalTree, uma ferramenta bioinformática desenvolvida em Python e executada via terminal no sistema operativo Ubuntu, cujo objectivo é comparar e avaliar os dados de *clustering* de duas *pipelines* de tipagem microbiana (ex. métodos tradicionais ou *WGS*).

O EvalTree é uma ferramenta flexível que orquestra vários *scripts* para comparação de dados de *clustering* provenientes de diferentes *pipelines* de tipagem, como *SNPs*, alelos e de métodos tradicionais. Esta ferramenta processa e filtra dados de *clustering*, aceitando dois *inputs* (pastas ou ficheiros). Quando fornecida uma pasta gerada pelo ReporTree (ferramenta bioinformática que identifica *clusters* a todos os *thresholds* possíveis), uma maior gama de funcionalidades estão disponíveis, tais como a caracterização dos *clusters* com metadados ou a análise comparativa ao nível do surto. As principais análises do EvalTree são:

1. «Caracterização de *pipelines*», relatando o número de amostras e *thresholds* por *pipeline*, e também fornecendo visualização gráfica para a caracterização de *clusters* com metadados;

2. «Congruência entre *clusters de pipelines*», avaliando a congruência da composição dos *clusters* entre sistemas de tipagem, em todos os níveis de *thresholds* possíveis (para *WGS-based*) e também identificando regiões de estabilidade em cada *pipeline*;

3. «Análise de surtos», determinando a sobreposição entre os *clusters* identificados para cada *pipeline* em um nível de surto definido pelo utilizador (disponível apenas se uma pasta ReporTree for fornecida como entrada).

Os resultados de todas as análises são sumarizados num relatório em *HTML*, sendo que os ficheiros intermédios com os resultados mais detalhados são também fornecidos. Esse relatório ainda disponibiliza informações suplementares como legendas, sugestões de ficheiros que ajudam na interpretação dos dados, bem como a hiperligação para o GitHub. Para promover a usabilidade do EvalTree, esta ferramenta foi disponibilizada em várias plataformas como o GitHub e o PyPi. Um pacote *conda package* também foi criado tendo como base a ferramenta Poetry.

O EvalTree inclui um conjunto de testes unitários que cobrem os argumentos e os resultados presentes nos ficheiros e que permitem assegurar que o código está funcional. Além disso, EvalTree foi validado com um *dataset* de 2974 isolados de *Salmonella enterica*, previamente compilado no projeto BeOne, que foi testado em diversas *pipelines* de alelos.

Para avaliar o desempenho do EvalTree em termos de tempo de execução, um conjunto de três análises de *benchmarking* foram efetuadas utilizando o mesmo *dataset* de *S. enterica*. Para tal, foi avaliado i) o impacto do número de amostras do *dataset* no tempo de execução do EvalTree, ii) o efeito da diversidade do *dataset* utilizando dois serótipos de *S. enterica* (Typhi e Thompson) com diferentes diversidades genómicas, e, por fim, iii) a influência do número de amostras do *dataset* no tempo de execução do EvalTree independentemente da diversidade do grupo (ST11). Os resultados revelaram que a diversidade do *dataset* é o factor chave no desempenho do tempo de execução do EvalTree.

Para apoiar a implementação da *pipeline* de vigilância genómica de *Salmonella enterica* no Instituto Nacional de Saúde (INSA), um exercício comparativo de duas *pipelines* de alelos foi feito. Foi comparado o *workflow* da pipeline de *WGS* desenvolvido na Unidade de Genómica e Bioinformática do INSA, com a pipeline implementada pela *European Food Safety Authority (EFSA)*. As principais diferenças entre os *workflows* de WGS são relativas às *pipelines* para controlo de qualidade de *Reads* e *Assembly* e a versão do *allele caller* utilizada. Dessa forma, para determinar a congruência da composição de *clusters* genéticos identificados pelas duas *pipelines* no mesmo *dataset*, procedeu-se à análise dos respectivos dados com o EvalTree. Os resultados revelaram que a *pipeline* do INSA é mais permissiva

relativamente ao número de amostras que passam nos controlos de qualidade do que a *pipeline* da EFSA. Ambas as *pipelines* têm um número de *thresholds* muito similar, sendo identificados 3045 *thresholds* na *pipeline* do INSA e 3059 *thresholds* na *pipeline* da EFSA. Ambas as *pipelines* mostraram ter uma grande região de estabilidade com semelhante comprimento (aproximadamente 850 *thresholds* consecutivos) em que o *clustering* é muito similar. No que diz respeito à congruência de *clusters* entre ambas as *pipelines*, o EvalTree revelou a existência de uma alta congruência a todos os possíveis níveis de *threshold*. Isto é evidenciado pela análise de pontos de correspondência identificados entre as *pipelines* em praticamente todos os níveis de *thresholds*, com a respectiva linha de tendência linear a apresentar um $r^2 > 0{,}999$ e uma inclinação de 0,99 em ambas as direções, refletindo uma excelente concordância entre as *pipelines*.

Para investigar se ambas as *pipelines* detectam os mesmos sinais de surto, foi utilizado um *threshold* de 10 distância alélicas (AD), o qual é frequentemente utilizado em abordagens de *cg/wgMLST* para *datasets* de *Salmonella*. O número total de *clusters* detectados por ambas as *pipelines* foi muito similar, sendo identificados 232 *clusters* na *pipeline* do INSA e 235 *clusters* na *pipeline* do EFSA. Ambas as *pipelines* detetaram 209 *clusters* com exatamente a mesma composição. Os resultados revelaram que a um *threshold* de 10 AD, 90.09% dos *clusters* da *pipeline* do INSA são identificados pela *pipeline* da EFSA. Por sua vez, 88.94% dos *clusters* da *pipeline* do EFSA são identificados pela *pipeline* do INSA. Estes resultados sugerem que 9 em cada 10 sinais de surtos detetados por uma *pipeline* são identificados pela outra. Uma análise semelhante foi efetuada com um *threshold* mais estrito de 5 ADs, aplicando-se uma flexibilização do *threshold* em 1 AD, tal como recomendado em estudos anteriores para *pipelines* que utilizam o mesmo esquema, sendo esse o caso de estudo. A este nível foi demonstrado que 93.09% dos *clusters* do INSA foram detectados até 6 AD na *pipeline* da EFSA, enquanto que 92.31% dos *clusters* da EFSA a 5 AD foram identificados na *pipeline* do INSA até 6 AD. Em conjunto, estes resultados indicam alta concordância entre as *pipelines* do INSA e EFSA mesmo ao nível de surto.

Em conclusão, o EvalTree constitui uma solução prática e flexível para a comparação de dados de *clustering* entre laboratórios. Contribui para a avaliação contínua e para a sustentabilidade a longo prazo de qualquer *pipeline* (por exemplo, atualizações de *software*). A geração de um relatório interactivo com o sumário dos resultados facilita a comparabilidade dos dados de diferentes laboratórios, promovendo assim o estabelecimento de uma estrutura integrativa de vigilância genómica *One Health*.

Palavras Chave: EvalTree, Vigilância Genómica, Sequenciação de Genoma Inteiro, Congruência de *Pipelines*, Surtos

Contents

List of Figures

List of Tables

Acronyms

AD	Allelic Distance
AMR	Antimicrobial resistance
AR	Adjust Rand
AWC	Adjusted Wallace Coefficient
cgMLST	Core genome MLST
CS	Congruence score
ECDC	European Centre for Disease Prevention and Control
EFSA	European Food Safety Authority
EQAs	External Quality Assessments
HC	Hierarchical Clustering
INSA	National Institute of Health Doctor Ricardo Jorge
MLST	Multi-Locus Sequence Typing
MST	Minimum Spanning Tree
nAWC	neighborhood Adjusted Wallace Coefficient
SNP	Single-Nucleotide Polymorphisms
ST	Sequence Type
QC	Quality Control
wgMLST	Whole-genome MLST
WGS	Whole-Genome Sequencing
WHO	World Health Organization
WOAH	World Organisation for Animal Health (WOAH)

Chapter 1

Introduction

1.1 Motivation

Bacterial pathogens pose a serious threat to public health. Indeed, a global study on antimicrobial resistance revealed roughly 8.9 million deaths associated with bacterial infections in 2019 [49]. Therefore, routine bacterial surveillance is essential in preventing infections and protecting public health. Advances in Whole-Genome Sequencing (WGS) technologies have significantly improved the monitoring of these pathogens by providing high discriminatory power to distinguish between genetically related isolates (or strains) based on differences in their DNA sequences [65]. Consequently, international agencies, such as the European Centre for Disease Prevention and Control (ECDC), recommend the implementation of WGS-based surveillance systems to track pathogens, investigate potential outbreaks and predict phenotype (e.g., serotyping, antimicrobial resistance (AMR), biofilm formation, pathogenicity or virulence) [25, 65]. The accuracy and reliability of WGS has led to its adoption across various sectors, including human, veterinary, food, and environmental fields [47, 62].

WGS has opened new perspectives in microbial typing. For instance, Multi-Locus Sequence Typing (MLST) is a traditional typing method that consists of the analysis of seven highly conserved housekeeping genes to compare isolates. However, it has no discriminatory power for outbreak detection [65]. With WGS, traditional typing methods are being replaced by gene-by-gene approaches, including core genome MLST (cgMLST) and whole-genome MLST (wgMLST), since they offer higher discriminatory power and a more accurate identification of phylogenetic relationships [65]. While cgMLST uses a large set of loci present in most strains within a species, providing a stable genomic framework for comparison, wgMLST includes the core and accessory loci, being useful to refine the resolution for outbreak investigations and evolutionary studies [65]. Therefore, routine genomics-based surveillance pipelines are increasingly relying on cg/wgMLST approaches and, for this reason, there is a constant effort to develop novel typing schemas (i.e. set of loci used for cg/wgMLST and their alleles) for bacterial pathogens [2]. Noteworthy, although there is an increased tendency to adopt a similar strategy for the genomics analysis of a given bacterial pathogen, as different laboratories followed different paths into their capac-

itation, there is also a lack of harmonization between their implemented bioinformatics solutions, which hampers inter-laboratories communication and cooperation at international and intersectoral levels [1]. In this regard, a recent study performed on four important foodborne bacterial pathogens has opened good perspectives regarding inter-laboratory communication and cooperation when using cg/wgMLST approaches, especially when they rely on the same schema [45].

In this context, it is important to continue performing research and developing novel solutions that contribute to the establishment of WGS-based One Health surveillance frameworks, in which the human health, animal health and food safety sectors cooperate towards an enhanced surveillance and control of bacterial pathogens [62].

1.2 Problem

Given the relevance of WGS for pathogen surveillance, the scientific community is making significant efforts to be constantly developing and refining (novel) bioinformatics solutions. As different laboratories are following different and independent paths in the implementation of bacterial genomic surveillance systems, they end-up implementing different solutions according to their specific needs and operational setting. For instance, while some laboratories opted for commercial software, others opted to implement freely available solutions or even to develop their own customized tools tailored to specific institutional needs [2]. Therefore, as mentioned before, there is a lack of harmonization that is jeopardizing the inter-laboratory communication and cooperation necessary for a proper international and intersectoral monitoring of bacterial pathogens. Although forcing all laboratories to converge on a unified bioinformatic method would solve this problem, this is an unrealistic and utopic vision. Instead, there is a need for innovative solutions and research studies that inform about the comparability of the different bacterial typing methods, providing a "communication dictionary" that allows the inter-laboratory communication despite their different implemented genomic surveillance approaches. Furthermore, with the rapid advancement of technology and scientific knowledge, there is a clear need for solutions that ensure the long-term sustainability of pipelines, such as assessing the impact of software updates, parameter changes, and workflow modifications on the clustering results. In parallel, it is also important to support the development of new methods for bacterial surveillance, for instance, by facilitating the comparison of existing and new schemas to select the most appropriate type of pipeline based on the characteristics of the bacteria being studied.

1.3 Objectives

In this context, this thesis project aimed to develop a novel bioinformatics toolbox for comparative clustering evaluation of WGS pipelines for bacterial genomic surveillance, providing user-friendly reports with interactive graphical visualizations. To this end, the specific objectives are:

1. Develop a tool (EvalTree) with an automated workflow that efficiently processes the clustering information of two typing systems, orchestrates the methodology for clustering congruence analysis developed by [45], and generates a user-friendly report;

2. Validate and benchmark the tool;

3. Demonstrate its applicability in a real-case scenario: the implementation of a WGS-based surveillance system for *Salmonella enterica* in Portugal.

1.4 Methodology

To achieve the proposed objectives, this thesis project was organized into five tasks:

1. Script development: Development of a *Python* script that automates and orchestrates the previously developed methodology for inter-pipeline clustering congruence assessment, eliminating the need for manual file handling and the individual execution of scripts.

2. Report creation: Extension of the script described in Task 1 by designing and building a user-friendly and surveillance-oriented report. This includes the integration of interactive graphical visualizations, such as *Plotly* or similar libraries, and evaluating the most suitable format for report generation, such as HTML.

3. Validation and benchmarking: Validation and benchmarking of the toolbox (both its analytical and graphical components) using a publicly available *Salmonella enterica* dataset [42, 43].

4. Tool release: Creation of Conda and PyPI packages for installation and public release of the toolbox on GitHub.

5. Applicability in a real-case scenario: Application of the tool to compare the clustering results and performance of the genomic surveillance pipeline recently implemented at the National Institute of Health Doctor Ricardo Jorge (INSA) for *S. enterica* and the pipeline used by the European Food Safety Authority (EFSA) for the same species [25, 46].

1.5 Contributions

One of the main outputs of this thesis project is a novel bioinformatics toolbox for comparative clustering evaluation of any two bacterial typing systems (https://github.com/insapathogenomics/ACDTree/tree/main/EvalTree), providing relevant information not only regarding inter-pipeline clustering congruence at all possible levels, but also regarding the ability of two typing systems to detect similar outbreak signals. As such, it is expected to contribute to the genomics

surveillance field, facilitating future communication and cooperation between laboratories, countries and sectors, allowing the implementation of a real One Health surveillance.

Given the versatility of the developed toolbox, the impact of this thesis is expected to go beyond this context and also help ensure the long-term sustainability of WGS-based pipelines. For instance, the developed toolbox can be used to assess the impact of a new software version or parameter changes on clustering, thus supporting more informed decisions during the entire life-cycle of a genomic surveillance pipeline, from its initial development to continuous maintenance and updates.

By applying this tool to a real-case scenario, namely, to evaluate how the newly implemented *S. enterica* genomic surveillance pipeline at INSA compares to the one implemented by EFSA, this thesis demonstrates how these systems compare at all possible resolution levels and at potential outbreak level, paving the way for the future implementation of this approach in other laboratories for similar assessments, and demonstrating how it can be used to take External Quality Assessments (EQAs) to a larger-scale.

In addition to the development of this tool, the work also resulted in the acceptance of an abstract and the presentation of a poster at the 14th edition of the International Meeting on Microbial Epidemiological Markers (IMMEM XIV).

1.6 Document Structure

The thesis is organized in 5 chapters:

Chapter I - Introduction: This chapter provides a comprehensive overview of the thesis. It highlights the impact of infectious diseases on public health and describes the role of bacterial surveillance systems based on WGS technology in prevention and control of infectious diseases. Furthermore, it discusses the main problems in the implementation of WGS-based surveillance systems. Finally, it proposes a potential solution: the development of EvalTree, a toolbox designed to inform about the comparability of clustering data, thus promoting data sharing and contributing to enhance communication among laboratories.

Chapter II - Related Work: This chapter presents the scientific background that frames this thesis. It includes a brief literature review of the main bacterial typing methods used in WGS surveillance, such as cg/wgMLST and SNP analysis. Their advantages and disadvantages are underscored. The main WGS recommendations provided by European agencies (ECDC and EFSA) also are presented.

Chapter III - Methodology: This section presents the entire workflow used during the development of the EvalTree, and describes how the tool was validated and benchmarked. This chapter also indicates the methodology applied to evaluate of cluster congruence between the newly implemented *S. enterica* genomic surveillance pipeline at INSA and that of EFSA.

Chapter IV - Results and discussion: This section describes the implementation of EvalTree, including the processing of different types of input files and the main outputs provided by this tool. Furthermore, it outlines how the tool was validated and benchmarked and the developed installation solutions. In the end of this chapter, the results of the comparability assessment between INSA's and EFSA's genomic surveillance workflows for *S. enterica* are presented, illustrating how EvalTree can support the implementation of genomic surveillance systems in any laboratory.

Chapter V - Conclusion and Future Perspectives: This chapter summarizes the contributions of this thesis to genomic surveillance, discussing future perspectives for the application of EvalTree and its impact in promoting result comparability, data sharing, and in facilitating the exchange of information between countries.

Chapter 2

Related Work

2.1 Bacterial typing

Bacterial typing (i.e. classification) methods are used to monitor the emergence and spread of infectious diseases in humans. Method selection depends on the discrimination power, laboratory reproducibility [52], speed, cost, ease of use, and availability of equipment [67]. Typing methods can be classified as traditional (e.g., serotyping and classic MLST) and non-traditional (e.g., WGS).

2.1.1 Traditional typing (serotype and classic MLST)

Traditional typing methods have historically been employed to investigate the epidemiology of bacterial pathogens in defined geographic regions [64]. The choice of method is species-specific and may rely on either phenotypic traits or on DNA-based analyses.

A classical example of a phenotypic, non-molecular approach is serotyping, which differentiates bacterial strains within the same species based on the expression of cell surface antigens, most notably the O (somatic) and H (flagellar) antigens [52, 65]. This method is commonly applied in the classification of *Salmonella* isolates [67]. Although serotyping is straightforward to perform, it may not be sensitive enough to detect small genetic differences between strains, thus lacking the discriminatory power to differentiate closely related ones [51]. In addition, it is also time-consuming and labour-intensive [65]. Another critical limitation is that phenotypic methods do not necessarily reflect the genotype, since bacteria can exchange DNA without changing their phenotype. For this reason, it has been replaced by genotypic methods, which rely on bacterial DNA for strain discrimination [67]. An example of such a genotypic method is multilocus sequence type (MLST) [38].

MLST is a molecular-based approach that uses a defined schema of multiple internal fragments of housekeeping genes to detect DNA sequence changes between isolates, which are undetectable using phenotypic methods [65]. This technique [39, 40] involves the amplification and Sanger sequencing of seven core housekeeping genes, considered essential and conserved across all strains of the species.

Depending on the specific aims of the study, the number of loci used in an MLST schema can vary from six to ten, with seven genes being most commonly used [52]. Each unique sequence at a given locus is assigned an allele number, and the combination of alleles across all loci defines an allelic profile (e.g., 2-3-4-5-2-7-8). This allelic profile is then used to assign a sequence type (ST) to the isolate. All known allelic variants for each species are recorded in a MLST schema, which comprises a set of loci with their associated allele sequences. These are hosted in the PubMLST database [40].

Despite its utility, the limited number of loci restricts MLST's discriminatory power [59], making it less suitable for studies of single-clonal pathogens or in outbreak investigations requiring higher resolution.

With the emergence of WGS technologies, which provide higher resolution for microbial tracking than traditional typing methods, the community is evolving towards the implementation of WGS-based genomic surveillance. Despite the vast panoply of bioinformatics pipelines available for WGS data analysis, they can be roughly divided into Single-Nucleotide Polymorphisms (SNP)-based and Allele-based (also known as Gene-by-Gene) approaches [65].

2.1.2 Single-Nucleotide Polymorphisms (SNPs) pipelines

The SNP-based approach identifies single nucleotide polymorphism by comparing the genome sequence of an isolate against a reference genome [56]. SNP analysis examines both coding regions and intergenic regions. Careful selection of the reference genome is essential as it should be as closely as possible to the isolate under investigation. This minimizes alignment errors and reduces the risk of incorrectly calling SNPs that are absent in the reference genome but present in the isolates [63].

The SNP-based approach [13, 56] can involve several steps. First, the quality control of sequencing reads to remove low quality bases and adapters is carried out. The reads are then mapped to the reference genome. The coverage depth is calculated to ensure reliable data. Subsequently, SNPs are called and filtered to retrieve high-quality variants.

SNPs analysis is a widely used method in epidemiology, as it provides high resolution for differentiating closely related isolates. It is useful for discriminating outbreaks-related strains and sporadic clinical cases [16].

Despite its advantages the approach faces challenges, including the lack of consensus regarding bioinformatic pipelines, reference genome selection, quality filtering thresholds and handling recombinations [56]. In addition, SNP analysis requires high computational resources [54].

Several tools are available to support the SNP analysis, such as CSI Phylogeny [32] and Snippy [7].

2.1.3 Allele-based pipelines: core-genome and whole-genome MLST(cg/wgMLST)

The allele-based approach is based on the principles of classical MLST [40]. Instead of being restricted to a seven-loci MLST schema, it incorporates a much larger number of loci, resulting in two

variants: core genome MLST (cgMLST) and whole genome MLST (wgMLST) [39, 40]. A cgMLST schema includes hundreds to thousands of core gene loci, which are present in all strains of a given species, while wgMLST schema expands this set to include both core and accessory genes [40].

Both approaches are restricted to coding regions and require the development of loci schemas, which are species-specific and allow the differentiation of closely related isolates. The selection of schema with the target genes, is a first step for comparison across isolates [56]. A schema contains all known allele sequences for each locus present within a defined group [52]. However, currently available schemas remain limited for non-model organisms [65], highlighting the need for the development of new, well-curated schemas to expand surveillance to a broader range of species. Schema construction is labour-intensive and requires large, representative genome collections to capture species diversity and select informative loci [63, 65]. They can be hosted in online databases, including PubMLST (https://pubmlst.org/), the Institute Pasteur (https://bigsdb.pasteur.fr), Enterobase (https://enterobase.warwick.ac.uk/), cgmlst.org (https://cgmlst.org/ncs) from Ridom SeqSphere and Chewie-NS (https://chewie-ns.readthedocs.io/en/latest/). [41]

Within the allele-based pipelines, different approaches exist to determine the cg/wgMLST allelic profile of each bacterial isolate, namely assembly free mapping or assembly based mapping [63]. In the assembly free mapping, raw reads are directly mapped against the allele schema for gene calling, as implemented in pipelines such as MentaList [27]. This method avoids genome assembly but requires stringent quality filtering of sequencing reads [63]. In the assembly-based approach, filtered reads are first *de novo* assembled to generate a high quality draft genome (assembly), using a genome assembler such Shovill [55]. Assembly quality is then evaluated using metrics such as GC content, N50, and the number of contigs and scaffolds [56]. Additionally, tools such as Kraken2 [68] can be employed to detect potential contamination. Once assembled, it can be processed with chewBBACA [60], an allele caller selected by EFSA and ECDC for their pipelines [25]. This tool annotates assemblies to identify genes and their sequences are extracted. These coding sequences are compared and aligned against the reference allele schema. During allele calling is assigned a numerical allele identifier to each sequence detected in the schema. If the sequence is already present in the schema, the corresponding number is used [65]; if it is new, a new allele identifier number can be assigned to the sequence.

The output of this allele calling is an allelic profile for each isolate, defined as a set of allele numbers across all loci in the schema [65]. The allelic profiles of each isolate are combined in an allelic matrix. Each row corresponds to an isolate and each column to a locus. This matrix forms the basis for downstream clustering analyses.

It is important to note that, contrary to SNP-based approaches where each SNP is counted as a difference between two isolates, in cg/wgMLST approaches one difference corresponds to one locus with different alleles [56]. Moreover, cg/wgMLST is restricted to the genes present in the schema. As such, SNP-based analyses are expected to provide a higher resolution in discriminating between close-related isolates [65]. However, as cg/wgMLST approaches allow the direct comparison of sequences with predefined schema [59], eliminating the need for a close-related reference genome [17], it is not

only less computationally demanding but also more suitable for the analysis of large and diverse datasets (as a surveillance collection) and, consequently, for long-term pathogen surveillance.

2.2 WGS-based bacterial surveillance

2.2.1 Implementation of One Health genomic surveillance frameworks

International organisations such as the EFSA, ECDC, the World Health Organization (WHO), and the World Organisation for Animal Health (WOAH) have strongly recommended the implementation of WGS technologies in public health and food safety surveillance [25, 24, 48, 69, 70, 71, 72]. These recommendations highlight the importance of WGS to detect outbreaks, investigate multi-country clusters (groups of close-related isolates), identify sources of infection, and trace disease transmission routes.

Following these recommendations, in the European Union, national reference laboratories are progressively adopting WGS for bacteria genomic surveillance [30]. However, the diversity of methodological choices for analysing WGS data (SNP, cg/wgMLST), together with the wide range of software available, poses a challenge for laboratories. The selection of methods depends on laboratory experience, political commitment and financial resources [9, 58].

Allele-based pipelines (cgMLST) can be the first approach to monitor isolates and identify potential cluster outbreaks [35]. To confirm the genetic proximity of isolates in potential outbreak clusters, SNP based pipelines are often used in a subset of isolates [65]. Therefore, although some laboratories rely on SNP-based approaches [14, 15] for surveillance, the use of cgMLST is increasingly becoming the standard for longitudinal surveillance across many bacterial species [29], including foodborne pathogens (FWD) [30], with SNP-based and wgMLST approaches being reserved for in-depth investigation of potential cgMLST-determined outbreaks [34].

The lack of fully standardized procedures, variation of bioinformatic workflow and parameters, hinder interoperability among national and international laboratories [8, 53]. Consequently, each laboratory is following its own path, and therefore they are at different stages of implementation of WGS-based surveillance.

As result, this heterogeneity raises concerns about result comparability. When data are not comparable, inter-laboratory communication becomes problematic, particularly during outbreak situations, where the same cluster outbreak should be detected regardless of the method used. To address this issue, laboratories must ensure that results are both accurate and comparable across institutions [56]. To support these efforts, the ECDC conducts External Quality Assessments (EQAs), that evaluate laboratories capacity to detect and identify pathogens [22].

To further circumvent these communications barriers and ensure interoperability of genomic data, the European commission informed ECDC and EFSA to develop a fully structured and interoperable surveillance system One Health (Figure 2.1).

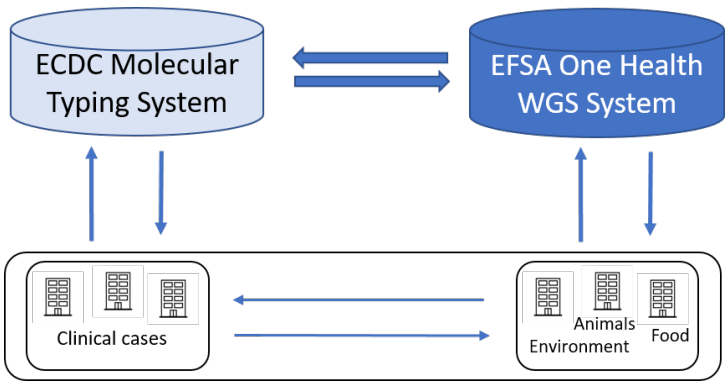


Figure 2.1: Schematic representation of the One Health system, illustrating its complexity. Laboratories within the same country must communicate with each other, but belong to different sectors and do not all integrate into the same database. Some laboratories report to the ECDC, while others report to the EFSA.

Within this framework, ECDC compiles and analyses clinical genomic data (allelic profiles of human isolates), while the EFSA compiles genomic data from food and environmental isolates (allelic profiles of non-human isolates). Priority pathogens include *Salmonella enterica*, *Listeria monocytogenes*, given their involvement in multi-country outbreaks [19] . This collaboration has resulted in a One Health WGS system database for bacteria surveillance, centralizing all information across sectors. The system enables the ECDC to query EFSA data in order to identify clusters containing both human and non-human isolates, thus supporting outbreak source attribution [25].

Integrating data from different sectors, such as human, animal, food and environment, enables the association of clinical cases with possible sources of contamination. This approach is at the core of One Health concept, which promotes coordinated genomic surveillance. Through the interoperability of both databases, it becomes possible to detect potential outbreaks at an early stage, thereby supporting the real-time investigation of multi-country food-borne outbreaks [25]

2.2.2 Detection of genetic clusters and signaling of potential outbreaks

The ultimate goal of WGS-based surveillance is the identification and monitoring of genetic clusters, i.e. groups of genetically closely-related isolates. For this purpose, to define these clusters, different clustering methods can be applied, with GrapeTree and Hierarchical Clustering (HC) being the most widely used approaches in bacterial genomic surveillance [65, 73]. Defining clustering groups requires calculating a distance matrix from an allelic or SNP matrix. Each isolate is represented by allelic or SNP values, and pairwise distances are calculated to quantify genetic differences. Once the distance matrix is generated, clustering methods can be applied to define the genetic groups.

In GrapeTree, can be used the novel minimum spanning tree algorithm (MSTree V2) [73], that it is able to handle missing data in large allelic profiles datasets. To mitigate the bias introduced by missing data, the algorithm calculates the asymmetric Hamming-like distances. This approach calibrates

the genetic distances according to the amount of missing data, reducing bias in cluster assignment [73]. This is particularly useful in outbreak investigations when datasets contain a high proportion of loci with missing data. In the resulting tree, each sample is represented by a node, with closely related samples grouped in the same node. Edges connect nodes to indicate allelic differences between sequenced samples [31]. The tree structure ensures that all nodes are linked with the minimum total edge weight possible [61, 73].

In the Hierarchical clustering - single linkage (HC-SL), first is calculated a distance matrix among samples using the Hamming distance, which measures the proportion of differing positions between two allelic profiles/aligned sequences [50]. In this calculation, the loci that are absent in either genome are ignored, and only loci that are shared are considered. The resulting distance matrix is then used to apply the single linkage hierarchical clustering algorithm, producing a dendrogram that represents the hierarchical relationships among the samples.

Figures 2.2 and 2.3 illustrate the previous clustering methods. Both used 90 *S.enterica* isolates of Thypi serotype [42, 43]. Isolates are colored by year, while uncolored isolates indicate that no metadata were available for those samples.

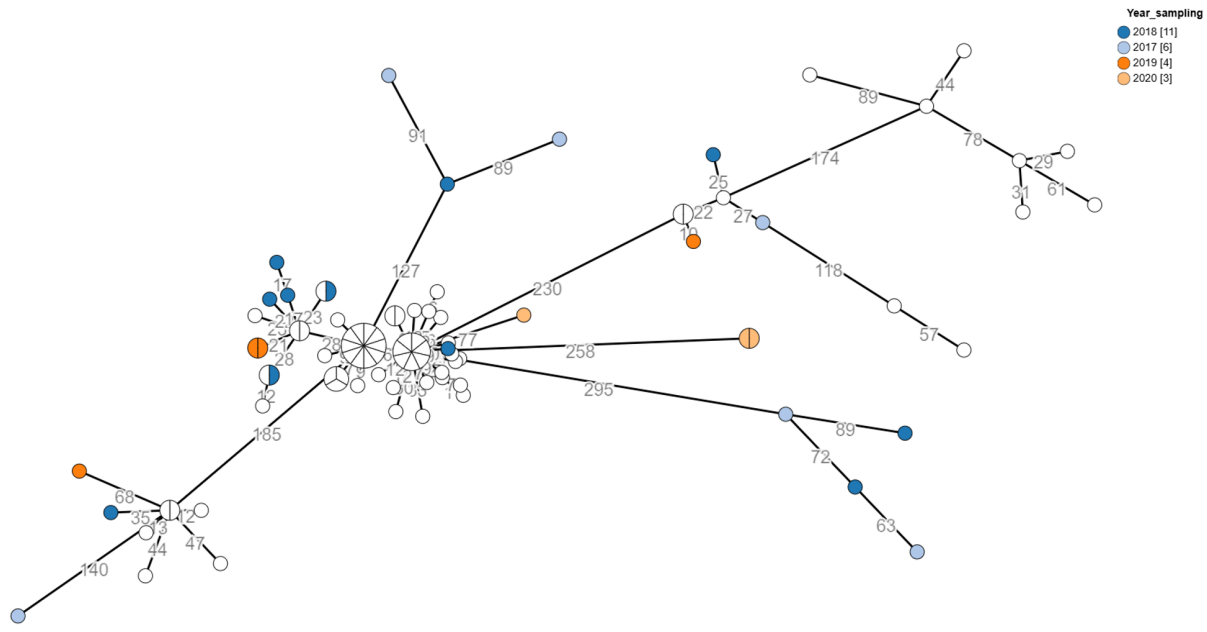


Figure 2.2: Minimum-spanning tree of *S.enterica* Typhi isolates. The tree was generated using ReporTree [44] and visualized with GrapeTree [73].

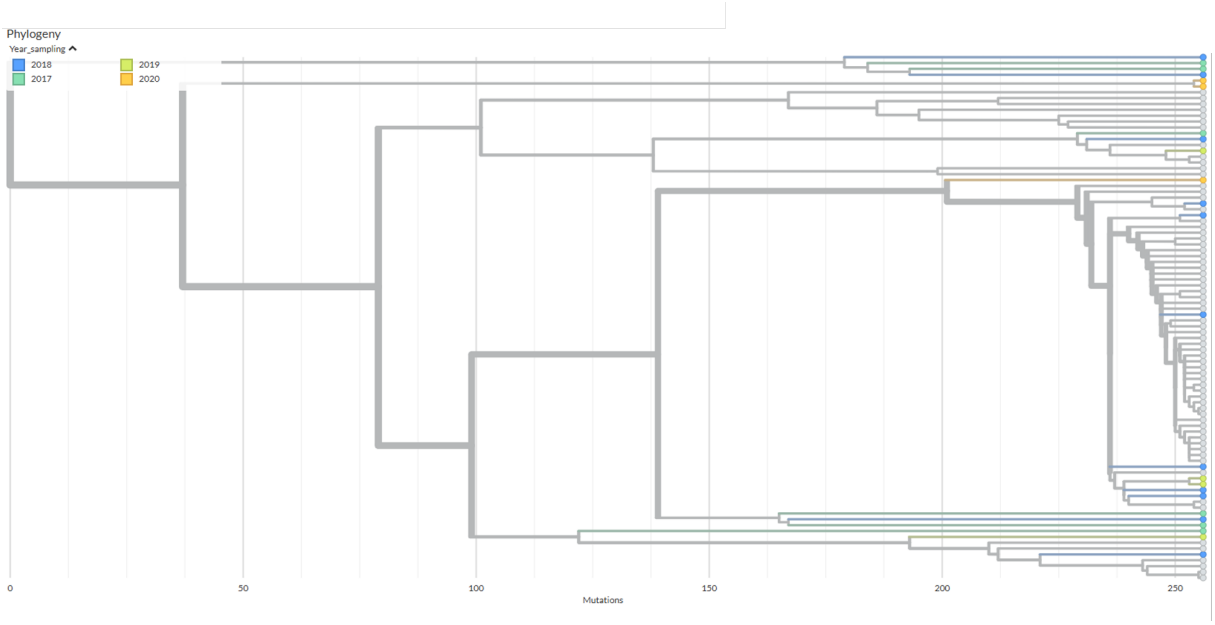


Figure 2.3: Single-linkage hierarchical clustering tree of *S.enterica* Typhi isolates. The tree was generated using ReporTree [44] and visualized with Auspice [5].

An example of a Python tool that identifies genetic clusters and characterize them according to the available metadata such as geotemporal spread and clinical information, is ReporTree [44]. It is able to identify genetic clusters at any or all possible distance thresholds or at the cluster stability regions, and generate reports with cluster characterization based on available metadata, such as, for example, timespan, geography, vaccination, and clinical status. Due to its flexibility, reproducibility, performance, low computational and time costs, and capability to handle large datasets efficiently, this tool has been used in routine laboratory surveillance. This tool is able to use either of the two previous clustering methods, and therefore respond to the laboratory needs. This determines the genetic cluster at any distance threshold and provides a partition table (**partitions.tsv*) containing the clustering results. In addition, it allows the application of a dynamic cgMLST + wgMLST approach, which can serve as an alternative to the commonly used cgMLST + SNP strategy for outbreak investigation.

The choice of threshold used to determine genetic clusters defines the maximum number of genetic differences allowed between isolates to group them within the same cluster. Thresholds play an important role in epidemiologic surveillance, enabling the identification of potential outbreak clusters, and form the basis for disease notification [66]. Thresholds are typically defined for each species based on knowledge of its evolutionary dynamics and historical outbreak data [6]. These fixed thresholds are widely used for outbreak detection, ensuring consistency in the identification of clusters across laboratories [6, 53].

However, the use of fixed thresholds has raised some concerns, regarding the tracking of pathogens. Some authors have suggested that using a fixed threshold can introduce bias into cluster identification,

as pathogen biology and the outbreak characteristics are not taken into account [53]. Moreover, the fixed thresholds do not consider the differences across the deployed pipelines. Due to these limitations, a dynamic threshold has been proposed, which considers factors such as time and mutation rate to define outbreak clusters [18]. Flexible thresholds of up to 2–3 allelic differences (AD) have been suggested to support the detection of the same cluster composition across different schema pipelines [45]. Pipelines that employ the same schemas can detect the same outbreaks sign when the threshold is adjusted by 1 AD [45]. Overall, threshold flexibility could benefit outbreak detection in inter-laboratory investigations, particularly in multi-country outbreaks.

2.3 Cluster congruence between different WGS-based surveillance pipelines

Laboratories have been facing challenges namely with the comparability of surveillance results across countries and sectors. This is particularly explored for foodborne bacterial pathogens, which are currently one of the priorities for implementation of genomic surveillance and require an integrative One Health approach for successful disease prevention and control. In this context, some research studies have tried to assess inter-pipeline comparability [13, 35, 45] and ECDC yearly performs EQAs [20, 21, 22]. Still, a recent study by [45] about inter-pipeline comparability used large datasets (with more than 2000 isolates each) of four important foodborne pathogens (*Listeria monocytogenes, Salmonella enterica, Escherichia coli*, and *Campylobacter jejuni*), which were provided to different European laboratories for application of their respective routine genomic surveillance pipeline and for assessment of cluster congruence. This study was performed in the frame of the BeONE project under the One Health European Joint Program (OHEJP) and involved laboratories from 10 different countries, including the food, animal, and human health sectors. In order to compare this large amount of data at all possible threshold levels (including in-depth outbreak-level analyses), as well as for comparison of the WGS-based surveillance results with other traditional typing solutions, the authors designed a methodology based on pairwise comparisons of clustering results [45]. Specifically, they relied on a previously developed tool ("ComparingPartitions", *comparing_partitions_v2.py* available at https://github.com/insapathogenomics/ComparingPartitions) [12] that calculates different coefficients (e.g., Adjusted Rand, Adjusted Wallace, etc.) between typing solutions [57] to compare:

1. the clustering obtained at progressively increasing distance thresholds of a cg/wgMLST typing method, by determining the neighborhood Adjusted Wallace coefficient (nAWC) at consecutive partitions:

$$\text{nAWC} = AWC_{n+1} \rightarrow AWC_n \tag{2.1}$$

2. the clustering obtained at all possible resolution levels between two typing methods, independently of whether they are cg/wgMLST or traditional, combining the AWC between method A and B (in

both directions) and the Adjusted Rand (AR) into a Congruence Score (CS) of each comparison. The AWC measures the probability that two samples that cluster together using one method (at a given threshold level) will also cluster together using another one (at a given threshold level). AR measures the overall agreement between the typing methods [57]. Therefore, the CS can range from 0 (low congruence) to 3 (absolute congruence).

$$\mathrm{CS} = \mathrm{AWC}_{A \to B} + \mathrm{AWC}_{B \to A} + \mathrm{AR} \tag{2.2}$$

While (2.1) allows the identification of stability regions, i.e., distance thresholds providing similar clustering results, (2.2) allows the determination of the congruence between any two typing approaches. Furthermore, they developed a script (*get_best_part_correspondence.py*, available at https://github.com/insapathogenomics/WGS_cluster_congruence) that based on the output of *comparing_partitions_v2.py* identifies the inter-pipeline corresponding points, i.e. the threshold of one pipeline that provides the most similar clustering results in the other, assessed as the highest CS above a user-defined minimum score. When a given threshold has a single valid corresponding point, this is assumed as the point of highest congruence. When a given threshold has several valid corresponding points (i.e., the highest CS was found in multiple comparisons), the closest threshold is reported as the best match. Another python-based script processes the information regarding the inter-pipeline corresponding points and determines the trend line fitting their distribution, as well as the respective r^2 and slope (*heatmap_final_score.py*, available at https://github.com/insapathogenomics/WGS_cluster_congruence). This information is output in a tabular format and can be used as a proxy to evaluate whether two cg/wgMLST schemas/pipelines yield highly concordant clustering at the exact same level, with slopes close to 1 (i.e., y = x scenario) reflecting high clustering concordance and similar discriminatory power across all levels. To determine the number of clusters identified in one pipeline at a given threshold that could be detected with the same composition by another pipeline at a (or up to a) similar or even higher threshold, for example, to identify the ability to provide similar outbreak signals, they developed the script *stats_outbreak_analysis.py* (available at: https://github.com/insapathogenomics/WGS_cluster_congruenc), which is tailored to analyze the clustering information provided by ReporTree [44].

By applying this methodological strategy, the authors were able to gain a comprehensive overview of the clustering performance of different pipelines (allele, SNP and traditional typing) used by various laboratories in their routine surveillance. For instance, allele-based pipelines showed high congruence in clustering results across all threshold levels, among all bacterial species except *C. jejuni*. Still, differences in outbreak cluster detection were observed and their results indicate that a threshold flexibilization by up to 2–3 ADs increases (1 AD if using the same cgMLST schema) increases the likelihood that a given pipeline detects clusters with the exact same composition as another pipeline by roughly 10% [45].

2.4 The CENTAUR project

The Genomics and Bioinformatics Unit of the Department of Infectious Diseases at INSA, has a laboratory dedicated to, among other topics, tracking the emergence and spread of pathogens and supporting the investigation of infectious disease outbreaks. The team has been adopting international guidelines for the implementation of WGS-based routine surveillance, being also actively involved in research projects dedicated to the development of novel methods for surveillance of pathogens. This thesis project is the scientific continuation of the work by [45], being carried out at INSA in the frame of the CENTAUR research grant, an international project that aims to address the current limitations in the rapid development and wider adoption of wg/cgMLST schemas for bacterial surveillance and outbreak detection. The main goal of this initiative is the creation of tools that enable the rapid and efficient creation and validation of high-quality wg/cgMLST schemas for relevant bacterial pathogens, including any bacterial pathogen X.

Chapter 3

Methodology

3.1 Code development

With the main goal of developing novel bioinformatics toolbox for comparative clustering evaluation of WGS pipelines for bacterial genomic surveillance (EvalTree) this thesis project started by the development of a python-based script (using python 3.8) (EvalTree.py, available at https://github.com/insapathogenomics/ACDTree/tree/main/EvalTree) that orchestrates the methodology for inter-pipeline cluster congruence assessment developed by [45]. This script manages user input specifications through the *argparse module*, which parses command-line arguments and stores them in variables and objects for further validation. This module was used to generate an integrated help message (Appendix A.1). The paths of all provided inputs are evaluated using the *os.path* submodule of *os* module. When instead of a file, a folder is provided as input, all contained files must share the same prefix to ensure their origin from the same run. If any difference exists in the filename prefix (e.g. pipeline name), the program stops. In cases where two individual input files are provided for pairwise clustering comparisons, their prefixes must differ, ensuring that they correspond to distinct pipelines.

Further validations are required when a ReporTree output folder is provided as input to EvalTree. The *fnmatch* module is used to identify the correct files required for the analysis. Clustering visualization and outbreak analysis require arguments with specific structures, which are validated. Stability analysis is only carried out when the corresponding file is not present in the input folder. Beyond that, correct argument combinations are validated using the *sys* module, preventing the execution with incompatible arguments. Data files are subsequently processed using the *pandas* library.

Once inputs are validated, clustering comparisons are performed through the automatic orchestration of the methodology reported by [45], available on GitHub (https://github.com/insapathogenomics/ComparingPartitions; https://github.com/insapathogenomics/WGS_cluster_congruence).
The *subprocess* module is used to execute *comparing_partitions_v2.py*, *get_best_correspondence.py* and

stats_outbreak_analysis.py, thereby avoiding manual CLI execution of each of these scripts and reducing the risk of user error.

Graphical visualizations are generated with the *Plotly* library, a high-level open-source Python library for building quick, interactive plots [33]. Specifically, the plots make use of the *plotly.express* and the *plotly.graph_object* modules. The *plotly.io* was used for reading, writing and converting figures into an HTML string. This allows the integration of the plots into an interactive HTML report.

Execution monitoring and reporting are implemented through a logging function. The log file records the progress of the script, software version, executed commands (*sys* library), timestamps for start and end times (*datetime* library) and any warnings encountered. Errors are displayed on the screen. Additionally, all functions include a docstrings, along with a descriptive text about the tool itself, using the *textwrap* library.

3.2 Validation and benchmarking

For validation and benchmarking analyses, a previously compiled *Salmonella enterica* dataset comprising 2974 genomes assemblies was used [42, 43].

For EvalTree toolbox validation, the publicly available files with clustering information at all possible levels (partitions tables) for each *S. enterica* allele-based pipeline analysed by [45] were downloaded (Table 3.1), as well as all metadata associated to the dataset, including *in-silico* prediction of traditional typing (serotype and ST) [42, 43, 45].

Table 3.1: Number of samples and thresholds retrieved from each partition file, using different allele-based pipelines and clustering methods (HC and GT).

Pipeline	**Clustering method**	**File name**	**Number of Samples**	**Number of Thresholds**
Bionumerics	HC	*Bionumerics_HC_partitions.tsv*	2954	2844
chewieSnake	GT	*chewieSnake_GT_partitions.tsv*	2959	2801
chewieSnake	HC	*chewieSnake_HC_partitions.tsv*	2959	2615
INNUENDO-like	GT	*INNUENDO-like-EFSA_GT_parititons.tsv*	2947	3066
INNUENDO-like	HC	*INNUENDO-like-EFSA_HC_parititons.tsv*	2947	2948
INNUENDO-like	GT	*INNUENDO-like-Enterobase_GT_partitions.tsv*	2956	2895
INNUENDO-like	HC	*INNUENDO-like-Enterobase_HC_partitions.tsv*	2956	2800
INNUENDO-like	GT	*INNUENDO-like-INNUENDO99_GT_partitions.tsv*	2962	2998
INNUENDO-like	HC	*INNUENDO-like-INNUENDO99_HC_partitions.tsv*	2962	2881
SeqSphere	HC	*SeqSphere_HC_partitions.tsv*	2960	2602

EvalTree was used to perform two types of pairwise comparisons: (i) between the cgMLST partition tables and the chewieSnake partition table, using both clustering algorithms (GrapeTree and Hierarchical Clustering) (Figure 3.1); and (ii) between the cgMLST partition tables and traditional typing data (serotype and MLST table) (Figure 3.2). For these comparisons and as illustrated in figures below, EvalTree received as arguments the respective two input files (or two cgMLST partition table or one traditional methods table and cgMLST table) and an output folder to save the results, while the remaining parameters were kept as default.

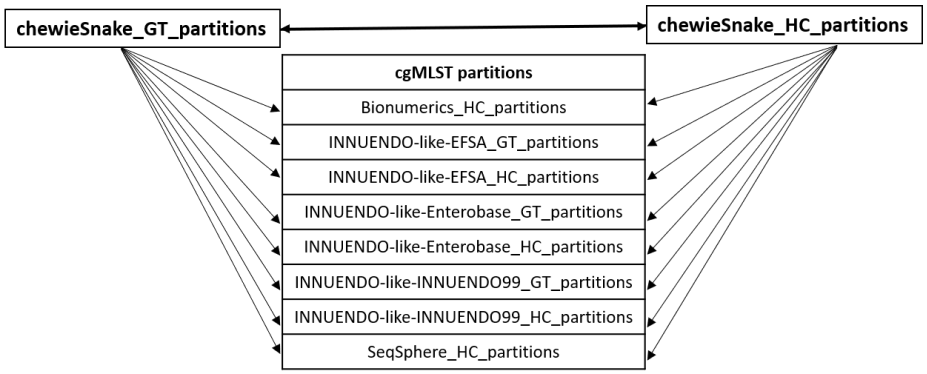


Figure 3.1: Pairwise comparison between allele-based pipelines carried out in EvalTree validation.

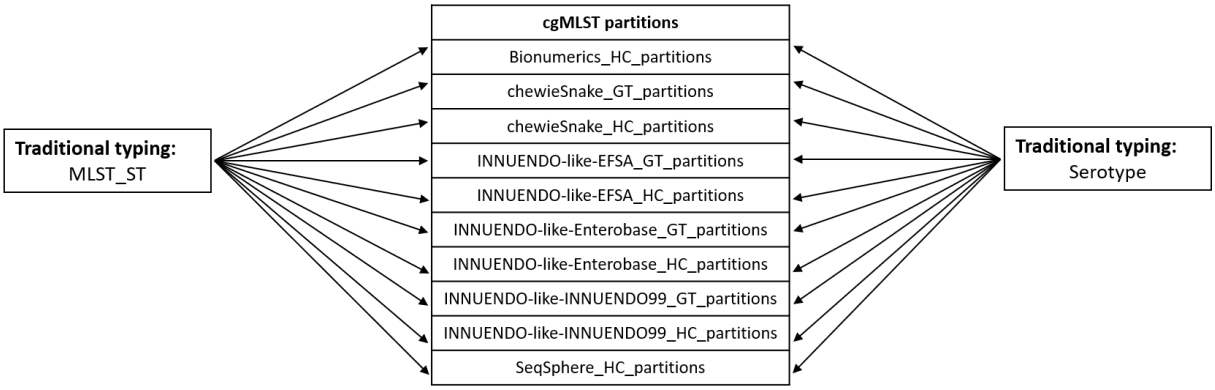


Figure 3.2: Pairwise comparison between traditional typing and allele-based pipelines carried out in EvalTree validation.

The results obtained in each of these analyses were compared with those reported by [45]. Given the large amount of data, for this validation process, a python script was developed to verify whether the results matched (Appendix A.2). The evaluated metrics included the reported number of thresholds and samples per pipeline, the comparison of congruence scores, and the threshold identified as of highest congruence between each allele-based pipeline and traditional typing classification (serotype and ST).

EvalTree benchmarking was performed on a desktop [Intel Core iX with 16GB RAM] using the same *S. enterica* dataset [42, 43]. Briefly, the raw sequencing reads were downloaded and PathoGen-Surveil v1.0.0 pipeline (the routine genomic surveillance pipeline recently implemented at INSA, available at https://github.com/insapathogenomics/PathoGenSurveil) was applied. This pipeline relies on INNUca v4.2.2 (GitHub https://github.com/B-UMMI/INNUca) for read quality control with FastQC [3], trimming with Trimmomatic [10], and contamination check with Kraken2 [68], as well as for genome assembly using SPAdes [4]. Moreover, it uses chewBBACA 3.3.5 [60] for allele calling using the *S. enterica* INNUENDO wgMLST schema with 8558 loci available at chewie-NS [41], yielding an allelic table (2974 samples). Finally, it was used ReporTree v2.5.4 [44] to remove samples with more than 5% of missing loci (–loci-called 0.95) and perform genetic clustering with the 3255 *S. enterica* cgMLST core loci using GrapeTree MSTreeV2 algorithm [73]. This yielded a partitions table and an allelic filtered matrix with 3255 loci and 2946 *S. enterica* samples, which served as the basis for all subsequent analyses.

To evaluate the runtime performance of EvalTree, three benchmarking tests were conducted:

1. To test the impact of the number of samples in EvalTree's running time, six subdatasets were generated containing 400, 800, 1200, 1600, 2000 and 2400 samples. The full dataset (2946 samples) was also included for comparison. In order to ensure that the genetic diversity was similar in each of the subdatasets, samples were not randomly selected, but instead obtained with DownTree v1.0.0 (https://github.com/jg-pereira/CENTAUR/tree/main/DownTree), providing the final GrapeTree partitions table as input and requesting one representative sample per cluster and a final number of assemblies similar to the targeted subdataset size. The allele matrix was filtered according to these results, generating the subdatasets necessary for downstream analyses.

2. To test the effect of the dataset diversity in EvalTree's running time, two *S. enterica* serotypes with similar representation in the dataset (n = 90) but different genomic diversity, namely Typhi and Thompson [45], were used. The allele matrix was filtered according to the available metadata, generating the subdatasets necessary for downstream analyses.

3. To test the influence of the number of samples in EvalTree's running time independently of the diversity, the allele matrix was filtered according to the available metadata for ST11, one of the most frequent STs causing human disease (n = 724). From this dataset, four different subdatasets with different sample sizes 150, 300, 450 and 600 were generated through random sample selection. The entire dataset was also tested.

The datasets generated (allelix matrix) for each benchmarking were then used as input for two independent ReporTree v2.5.4 [44] runs, using default parameters, but requesting clustering information at all possible threshold levels with GrapeTree (MSTreeV2) and HC (single-linkage) clustering algorithms. For each benchmarking, EvalTree was run with two input folders, each containing a partition table generated from the same set of samples but using different clustering methods (either GrapeTree

or Hierarchical Clustering), an output folder to store the results, while the remaining parameters were kept as default. EvalTree analyses were performed in triplicate, and the runtime for each replicate was recorded.

3.3 Conda package and installation tests

To facilitate environment sharing and ensure the reproducibility of this tool, a PyPI package was first built using the Poetry (version 2.1.3) tool as base. In order to create a Conda package, a Conda recipe was generated from the PyPI package using Grayskull (version 2.7.3; https://github.com/conda/grayskull). EvalTree is available on the GitHub repository and a dedicated page including all installation options is provided (https://github.com/insapathogenomics/ACDTree/tree/main/EvalTree). For constant validation of the functionality of all components within EvalTree, and also to provide a resource that verifies the success of installation, a set of unit tests were created. These tests were designed to identify potential coding errors and verify the correct handling of input arguments. Several tests were implemented to cover the EvalTree code, including the verification of inputs, the content of output files (e.g., stable region file, final score, All_correspondence), and the counting of expected file types (e.g., .tsv, .txt, .html). Tests were performed under different input conditions, such as providing two folders, or two files, or one file and folder. According to these inputs, different arguments were specified, and the number and content of the output were verified. All tests can be executed using Pytest.

3.4 Comparison of two genomic surveillance pipelines

The *S. enterica* dataset containing 2974 samples [42, 43] explained in dataset section 3.2, was used to compare the performance of two genomic surveillance pipelines: INSA and EFSA allele-based pipelines. INSA's pipeline (available at https://github.com/insapathogenomics/PathoGenSurveil) was run as described in section 3.2. The EFSA pipeline [25] was run using a script that facilitates its local deployment (available at https://github.com/insapathogenomics/cml_efsa.wgs.onehealth_facilitator). The allelic matrix obtained from each pipeline was filtered for the samples that passed quality control in both (n = 2821 samples). For each filtered allelic matrix obtained (INSA and EFSA), a simplified ReportTree [44] analysis was performed, with default parameters, and requesting GrapeTree [73] MSTreeV2 clustering algorithm, as it is the clustering method implemented in both institutions.

Finally, EvalTree was used to carry out the comparative congruence analyses and assess the performance of the two pipelines. It received two input folders, each containing the output data from the ReportTree analysis performed for the INSA and EFSA pipelines, respectively, as well as an output folder. To make the outbreak analyses it was used the threshold outbreak argument as well as the repeat threshold argument. All other parameters were kept as default. The evaluated metrics include the reported number of thresholds per pipeline, the comparison of congruence scores, the corresponding threshold points, the

length of stability regions, and the percentage of clusters with identical composition.

Chapter 4

Results and Discussion

4.1 EvalTree implementation

4.1.1 Input processing

EvalTree, Figure 4.1, is a flexible bioinformatic tool for the assessment of the comparability between any two typing systems. Designed to accept as input two files with the typing results of two systems (traditional typing or WGS-based clustering), EvalTree is also able to process the output folders of independent ReporTree [44] runs. These input files are processed to perform three different analyses (details in section 4.1.2):

1. "Pipeline characterization", reporting the number of samples and thresholds per pipeline, and also providing graphical visualization for the characterization of clusters with metadata;

2. "Inter-pipeline cluster congruence", assessing the congruence of cluster composition between typing systems, across all possible threshold levels (for WGS-based) and also identifying clustering stability regions per pipeline;

3. "Outbreak analysis", determining the overlap between the clusters identified by each pipeline at a user-defined outbreak level (only available if a ReporTree [44] folder is provided as input).

The main file processed by EvalTree is the one with typing information (equivalent to the *partitions.tsv* table of ReporTree, with clustering information at all possible threshold levels), which can be provided directly as an input or within a ReporTree folder. This file is used for pipeline characterization, including the identification of stability regions when more than one classification is available (e.g. multiple distance thresholds in WGS-based pipelines), and for inter-pipeline cluster congruence analysis.

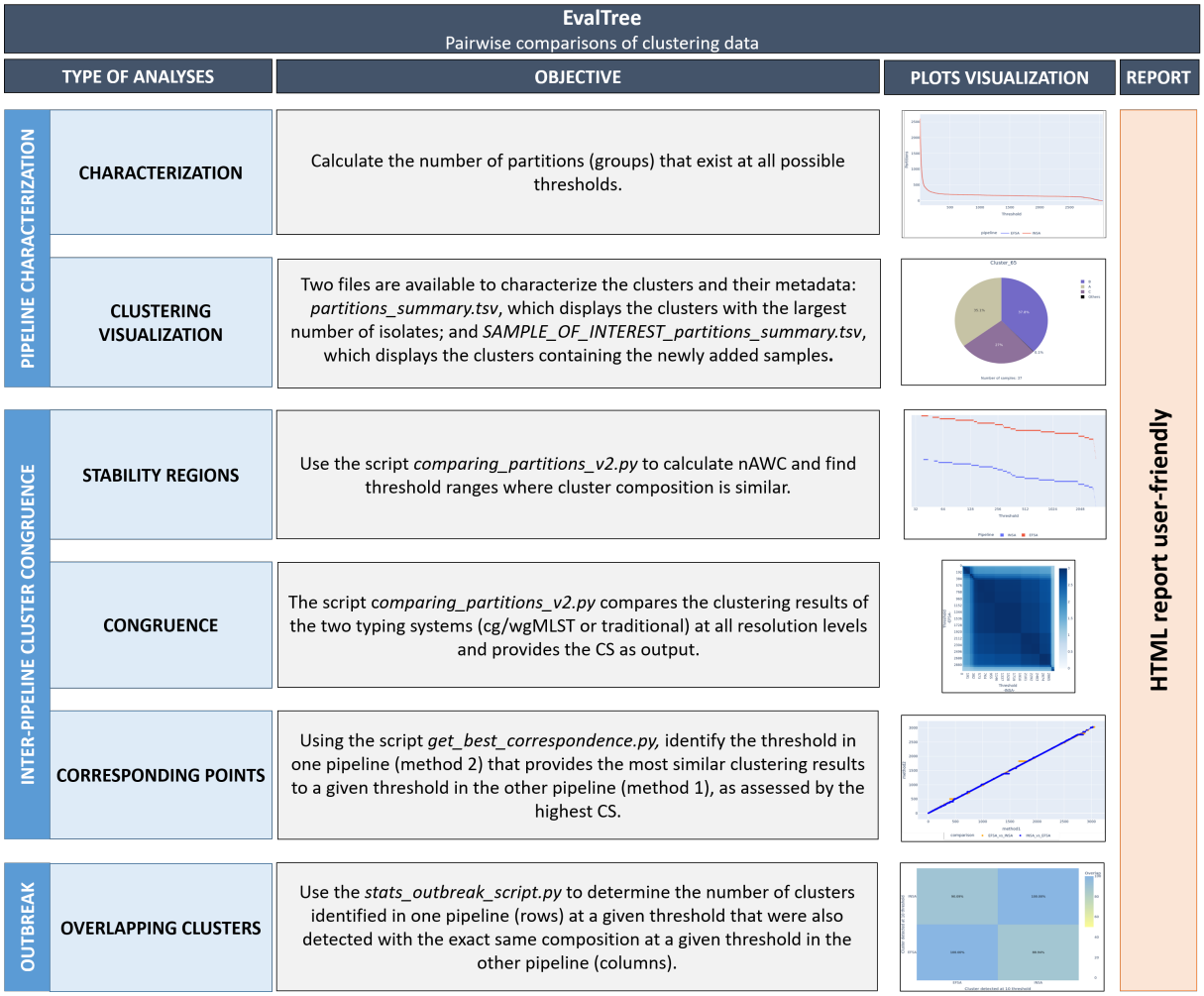


Figure 4.1: Types of analyses performed by EvalTree, with their objectives described and accompanied by the corresponding visualizations. Depending on the analyses selected by the user, the results are compiled into an HTML report.

Although being designed to compare any files with typing information, in reality, the user can take much more advantage of EvalTree's functionalities if providing a ReporTree folder. Indeed, in this case, EvalTree is able to also process the *partitions_summary.tsv* or (depending on the user-specifications) the *SAMPLE_OF_INTEREST_partitions_summary.tsv*, which provide cluster characterization information at each distance threshold. EvalTree uses this information to generate pie-charts according to the user-specified parameters (Table 4.1). If such information is available, it also reports whether the cluster composition was kept or not since the last ReporTree run of the pipeline under analysis and, if not, what is the number of new samples added to the cluster. Furthermore, if a *stableRegions.tsv* file is available within ReporTree folder(s), EvalTree processes its information to report stable regions, instead of performing

an additional step for their identification.

Finally, the "Outbreak analysis" is only available when a ReporTree folder is provided as input for the two typing systems, as it requires the information contained in ReporTree's *clusterComposition.tsv* file to assess the overlap between cluster composition at user-defined thresholds (details in section 4.1.2).

The handling of all these files can be adjusted through a set of arguments that enable the customization of the analyses (Table 4.1).

Table 4.1: Overview of arguments available in EvalTree, organized by analysis, category, argument name, and function.

Analyses	Category	Argument name	Function
Pipeline characterization	Starting point	`input (--i)`	Partitions files or other type of classification
		`output (--o)`	Directory to save the results
		`list (--list)`	List the names of the categories in the `partitions_summary` file
		`version (--v)`	EvalTree version
		`help (-h)`	Show the description of all arguments
	ReporTree clustering visualization	`plot_summary (-ps)`	Selection of the type of file to visualize the plots (`partitions_summary` or `SAMPLE_OF_INTEREST_partitions_summary`)
		`plot_threshold (-pt)`	Thresholds for visualizing the plots
		`column_plot (-cp)`	Categories to visualize in the plots
		`n_cluster (-n)`	Number of plots to display
		`plot_percentage (-pcp)`	Percentage of segments that the user intends to visualize in the plot
		`plot_number (-pcn)`	Number of segments represented in the plot for the selected category
Inter-pipeline cluster congruence	Stability regions	`n_stability (-n_stab)`	Range of thresholds where the cluster composition is similar
		`thr_stability (-thr_stab)`	The neighborhood Adjusted Wallace Coefficient (nAWC) threshold used to determine if a clustering threshold is considered stable
	Congruence	`threshold (-t)`	Filter the partition table
		`score (-s)`	Select the values for corresponding points
Outbreak analysis	Outbreaks	`threshold_outbreak (-to)`	Thresholds for pairwise comparison (fixed or dynamic)
		`repeat_thr_outbreak (-rto)`	Repeat threshold outbreak analysis

4.1.2 Inter-pipeline cluster congruence analysis and overlap of outbreak signals

One of the main outcomes of EvalTree is the information about pipeline comparability. Once the input files have been received, EvalTree forwards the processed clustering data inputs to the script *com-*

paring_partitions_v2.py, configured for pairwise comparisons (between methods). This script performs genetic clustering comparisons across all possible thresholds. The remaining arguments of this script are filled with default values according to the instructions of [45]. The output directory resulting from step above is then passed to *get_best_correspondence.py*, which identifies the corresponding threshold points between pipelines. The output of this script is then optimized, using the *remove_hyphen.py* script developed in this thesis project, which automatically removes rows without threshold correspondence between typing methods, thus avoiding the need for manual editing before subsequent analyses.

EvalTree also includes an optional outbreak analysis, with a facultative script orchestration of *stats_outbreak_script.py*, when specified by the user. This step is only executed if a cluster composition file, generated from a ReporTree folder, is present in the input directory. The analysis can be configured to use either a fixed or a dynamic threshold. A fixed threshold enables a direct cluster comparison between pipelines, whereas flexible thresholds define a range for cluster comparison, allowing the detection of clusters with similar compositions. Additionally, EvalTree can optionally repeat this analysis and generate a new report.

4.1.3 Main outputs and HTML report

EvalTree produces multiple output files allowing the user to explore data and share it. The main output file of *EvalTree.py* culminates with a production of a user-friendly HTML report containing interactive plot visualizations and respective legends of all the results (Figures 4.2, 4.3, 4.4 and 4.5). This report presents an initial overview section showing the runtime, version and the command line used for the run, and another two main sections, including pipeline characterization and inter-pipeline cluster congruence, and a facultative section of outbreak analysis.

The pipeline characterization section (Figure 4.2) is provided for each typing method under comparison, and displays a summary with the name of pipelines, number of samples and thresholds, and includes a line plot illustrating the number of partitions (clusters) per threshold. This information is saved in the **cluster_partitions.tsv* file. When the input file only has a single column (e.g. traditional typing), a bar plot is provided instead.

A subsection in the pipeline characterization can be produced, when the user selects the ReporTree clustering visualization (Figure 4.2), to characterize the clusters alongside their metadata files. Two files are available: **partitions_summary.tsv* or **SAMPLE_OF_INTEREST_partitions_summary.tsv* files. The cluster plots are organised by threshold (e.g., MST-7x1.0) and by the selected category (e.g., country). For a given threshold in a particular category (one or several), the tool generates one or multiple plots. Each cluster plot can contain one or multiple segments, where the size of each segment reflects the count/relative frequency of samples that exist within the analysed category. Additionally, the user can choose to display the segments by absolute number or percentage. The number of plots shown depends on the file used. When the **partitions_summary.tsv* report is provided, the number of plots displayed will follow the user's or default specification (Table 4.1). When the **SAMPLE_OF_INTEREST_parti-*

tions_summary.tsv file is used, the tool only displays those plots for clusters with the samples of interest.

In the case of inter-pipeline cluster congruence (Figures 4.3 and 4.4) analysis, a barplot is generated representing the stability regions identified in each pipeline, showing the threshold ranges where the clustering results are stable. The data is transformed into logarithmic data to facilitate the block visualization. Additionally, EvalTree presents a heatmap showing the congruence score obtained from the pairwise comparisons at all possible threshold levels. In addition, a scatterplot is produced indicating the inter-pipeline corresponding threshold points between the pipelines, also providing the linear tendency line fitting those points.

Optionally, when a ReporTree folder is provided as input and when specified by the user, the HTML report may also include a section reporting the overlap of outbreak signals between pipelines (Figure 4.5). This section includes a heatmap for each analysis requested by the user, with the percentage of overlap clusters that exist in one pipeline and also exist with the exact same cluster composition in the other.

Besides the HTML report, this tool provides a wide variety of files that are generated throughout the inter-pipeline cluster congruence analysis, such as the **stableRegions.tsv* which lists the block names, their respective threshold range, and the length of each block; the **metrics.tsv* that summarizes all comparisons between consecutive pairs of thresholds ("n + 1" → "n"); the **final_score.tsv* containing the congruence score for each pairwise threshold comparison, as well as additional files with the Adjusted Rand and Adjusted Wallace coefficients, corresponding to a decomposition of the CS presented in the heatmap; the **All_correspondence.tsv* file, which reports the inter-pipeline corresponding thresholds between the two pipelines, or even the **slope.tsv* file that includes statistics values such as the r^2 and *p-value* of the tendency line plot. Moreover, if the user requested the "outbreak analysis" the **stats_outbreak_summary.tsv*, which presents the total number of clusters detected at a certain level by each pipeline, as well as the number of clusters with exact composition between the two and exclusive of each of them is also provided. All these files are available in EvalTree's output folder, being available for consultation by the user in order to obtain more details than those summarized in the HTML report.

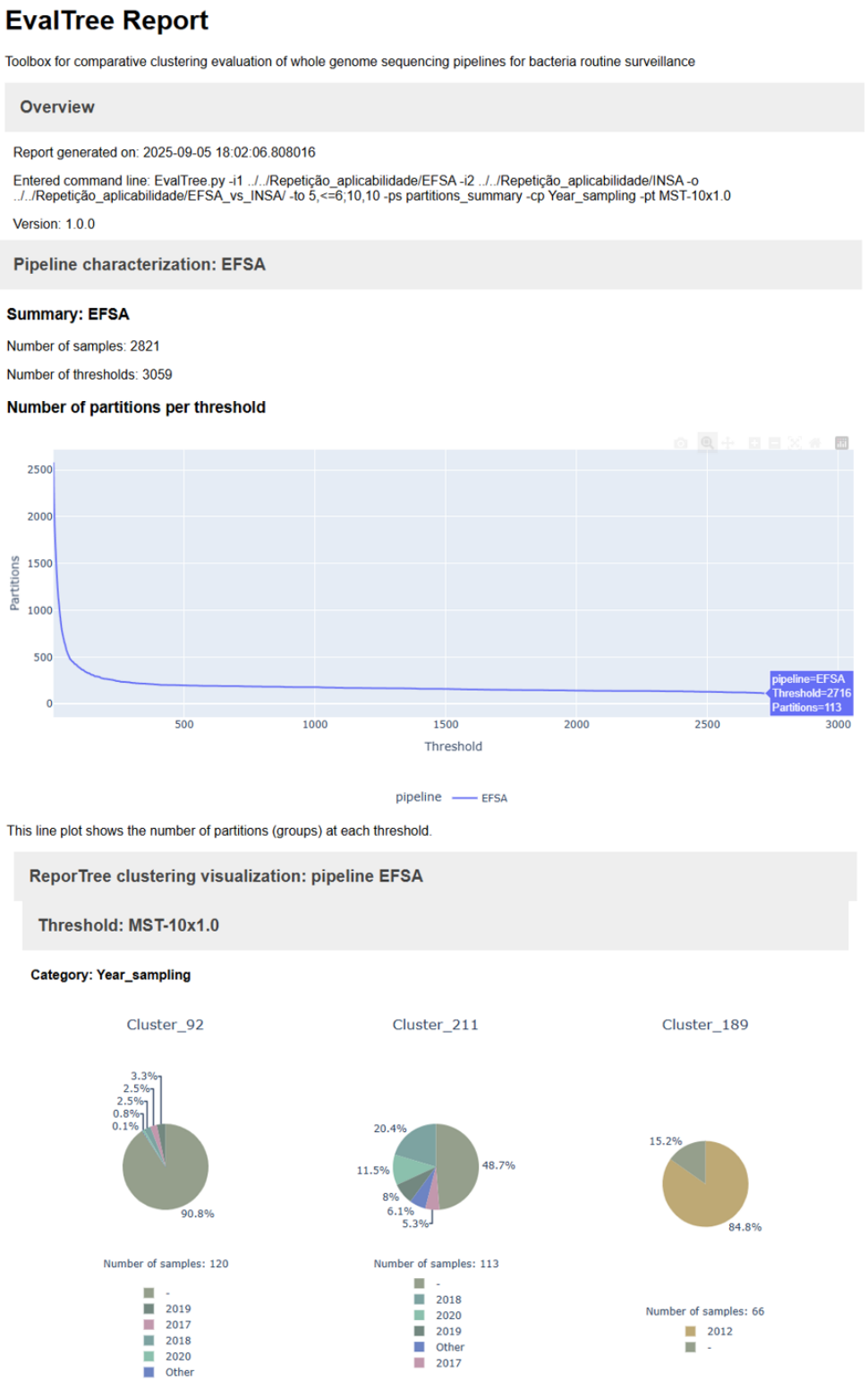


Figure 4.2: Illustration of an HTML report example: Pipeline characterization and ReporTree clustering visualization.

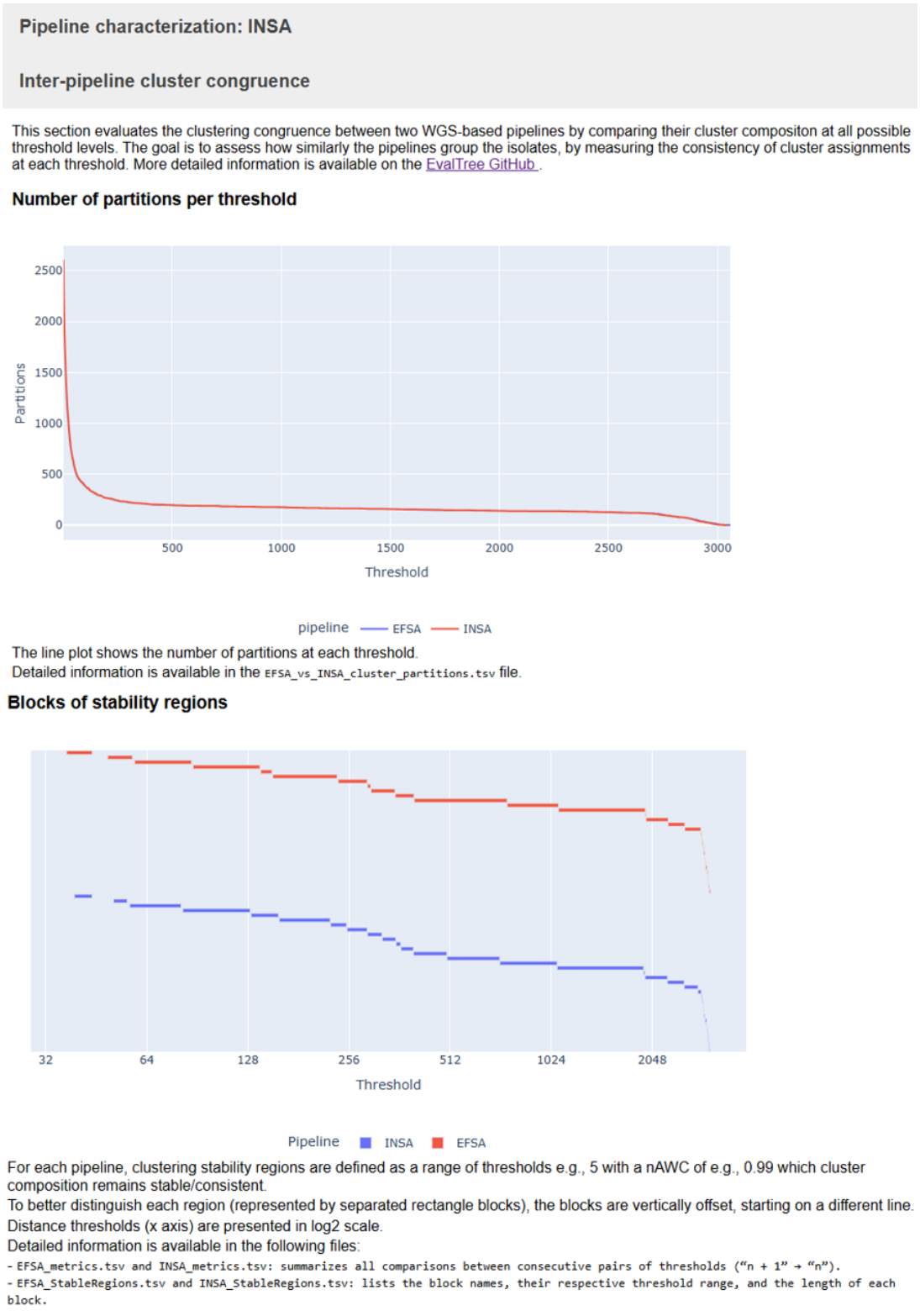
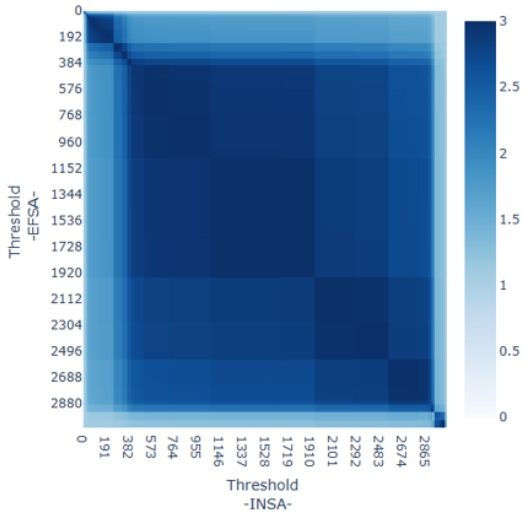


Figure 4.3: Continuation of the illustration 4.2 of an example HTML report: Inter-pipeline cluster congruence section (stability regions).

Congruence score



The heatmap presents a pairwise comparison of clustering results obtained from two pipelines, EFSA and INSA, at all possible distance thresholds.
For each threshold, the cluster composition generated by two pipelines are compared, and a congruence score is computed (CS).
The CS is based on Adjusted Wallace coefficient (AWC) and Adjusted Rand (AR).
The AWC measures the probability that two samples that cluster together by one method (at a given threshold level) will also cluster together by another one (at a given threshold level). This calculation is performed in both directions (method A → method B and method B → method A) for all thresholds.
AR evaluates the overall agreement between the typing methods. The congruence score (CS= AWC A→B + AWC B→A + AR) ranges from 0 indicating low congruence, to 3 indicating absolute congruence.
Detailed information is available in the `EFSA_vs_INSA_final_score.tsv` file.

Corresponding points

This graph shows the corresponding points (thresholds) between the two pipelines in both directions above (CS >= 2.85).
When comparing a set of samples between two pipelines, the probability of two sample clustering together in one method/pipeline in a given threshold may not to be the same in the other method/pipeline. Therefore:
- Orange line, the threshold in the INSA pipeline (method 2) that produces clustering results most similar to those in the EFSA pipeline (method 1) is identified.
- Blue line, the threshold in the EFSA pipeline (method 2) that produces clustering results most similar to those in the INSA pipeline (method 1) is identified.
Both methods produce similar clustering results when the tendency line has a slope near 1.
A linear tendency line supported by 2952 (blue) and 2918 (orange) points is presented.
Detailed information is available in the EFSA_vs_INSA_All_correspondence.tsv file.

Figure 4.4: Continuation of the illustration 4.3 of an example HTML report: Inter-pipeline cluster congruence section (congruence score and corresponding point)

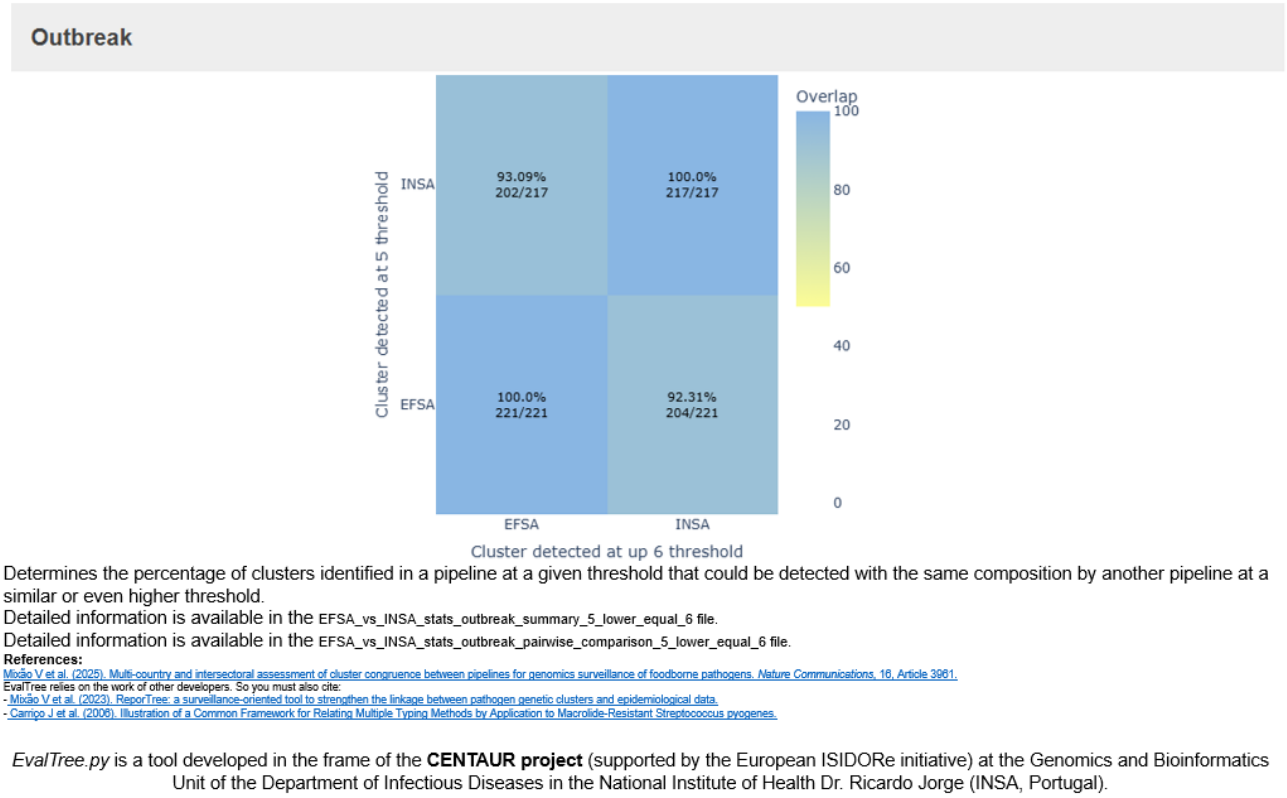


Figure 4.5: Continuation of the illustration 4.4 of an example HTML report: Outbreak analyses section.

4.1.4 Installation and code availability

The installation and code availability of EvalTree can be found in the GitHub repository (https://github.com/insapathogenomics/ACDTree/tree/main/EvalTree). The tool can be installed through the conda package using insapathogenomics channel (https://anaconda.org/insapathogenomics/evaltree) or alternatively through PyPi using pip (https://pypi.org/project/evaltree/).

4.2 EvalTree validation and benchmarking

In order to validate EvalTree, the analyses done by [45] for *S. enterica* were partially performed during this thesis project. Specifically, two types of validation were performed, the first one to compare traditional typing (serotype and ST) with each allele-based pipeline, as illustrated in Figure 3.2, resulting in 20 pairwise comparisons, and the second one to compare the chewieSnake [17] pipeline with all the remaining allele-based pipelines (details in the Methodology 3.2 section), yielding 17 pairwise comparisons (Figure 3.1). The results showed that EvalTree is able to report the same number of samples and thresholds for each pipeline as [45], and, more importantly, to identify the exact same distance threshold

(for each pipeline) with highest congruence with the traditional typing systems. Furthermore, when comparing the congruence score obtained for each pairwise comparison of allele-based pipelines, the only difference encountered was in the number of decimal places reported. For comparison purposes, outputs of [45] and the outputs of EvalTree were rounded to four decimal places. Altogether these results validate EvalTree's results, showcasing its readiness for release and implementation.

An important aspect to take into consideration for the implementation of any bioinformatics tool is its runtime. Although the number of samples was a factor expected to contribute to this, the dataset diversity was hypothesized as having a much bigger impact. Therefore, EvalTree's benchmarking was designed to evaluate these two factors using a publicly available dataset of *S. enterica* comprising 2974 isolates [42, 43] and comparing the clustering results obtained by INSA's *S. enterica* genomic surveillance pipeline (PathoGenSurveil, available at https://github.com/insapathogenomics/PathoGenSurveil) when using GrapeTree MSTreeV2 [73] and single-linkage hierarchical clustering (details in the Methodology section). As shown in Figure 4.6, when the runtime was evaluated for increasing dataset sizes with similar genetic diversity, the runtime increased. Indeed, on average, a dataset with 400 *S. enterica* samples took 111296,04 seconds (1 day and 6 hours), while a dataset of 2946 samples took 153312,69 seconds (1 day and 18 hours) (Figure 4.6, Table A.1), roughly corresponding to 38% more time for a dataset that is 6 times bigger.

At the same time, when assessing the runtimes considering two datasets with similar size (n = 90) but very different genetic diversity, as the Typhi and Thompson *S. enterica* serotypes [45], the most diverse dataset required, on average, 3593 sec (+/- 1 h), while the most clonal one required only 48,441026 sec (Figure 4.7, Table A.2). This suggests that, as initially hypothesized, the dataset diversity is the key factor in EvaTree's runtime performance. The runtime evaluation using subdatasets with different sizes and resulting from random selection (i.e. not ensuring that the dataset diversity was kept between datasets) further demonstrated this by revealing that, in this scenario, there is no tendency in EvalTree's runtime, and that actually the subdataset with 450 isolates required less than half of the time of the subdataset with 600 isolates (Figure 4.8, Table A.3).

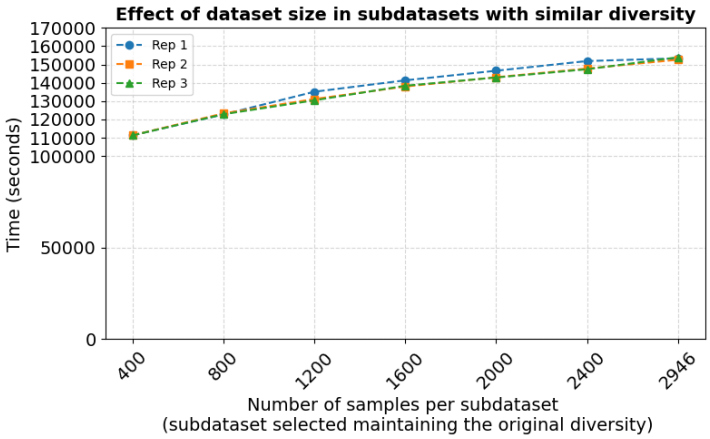


Figure 4.6: First EvalTree benchmarking, to test the impact of the number of samples in EvalTree running time. Seven subdatasets of *S. enterica* with different sizes (400, 800, 1200, 1600, 2000, 2400 and 2946 samples respectively) were used. Each dataset was run with GrapeTree and HC clustering analyses. Datasets with same set of samples were compared using EvalTree.

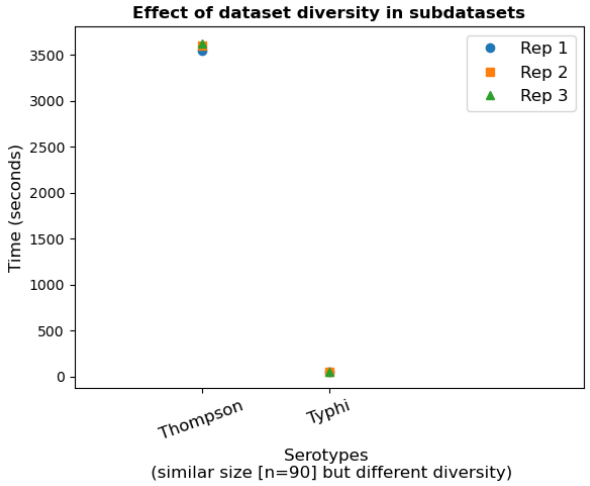


Figure 4.7: Second EvalTree benchmarking, to test the effect of diversity in EvalTree runtime for two *S. enterica* serotypes: Thompson and Typhi. Both datasets have same number of samples (n=90). Each dataset was run with GrapeTree and HC clustering analyses. For each serotype was carried out pairwise comparison between different clustering methods using EvalTree.

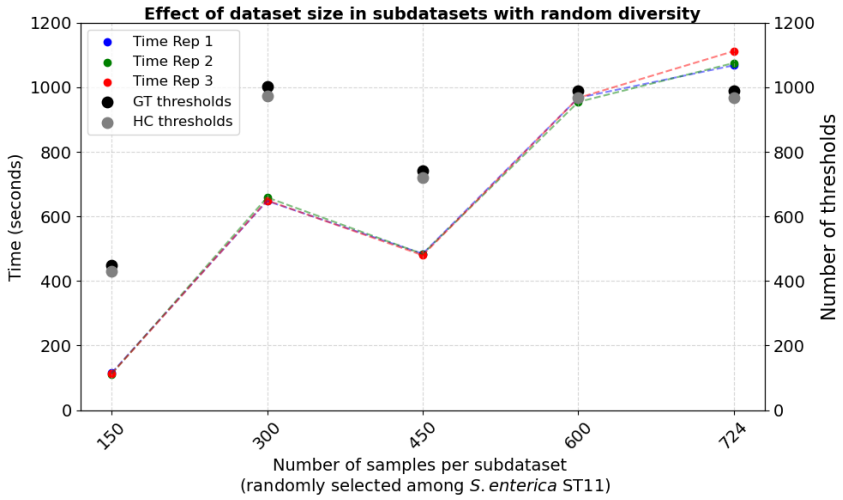


Figure 4.8: Third EvalTree benchmarking, to test the influence of the number of samples in EvalTree runtime, independently of the diversity. Five subdatasets with different sizes (150, 300, 450, 600 and 724) of *S. enterica* sequence type eleven (ST11) were created. For each dataset was run with GrapeTree and HC clustering analyses. Datasets with same set of samples were compared using EvalTree. To complement this information, the number of thresholds generated by the dataset with GT and HC methods was added per each size dataset.

4.3 Supporting the implementation of the *Salmonella enterica* genomic surveillance pipeline at INSA

S. enterica is the second most important zoonotic disease in the EU, with 77,486 confirmed human cases of *salmonellosis* reported in 2023 [26]. It was also the most frequently identified pathogen in large multi-country outbreaks mainly linked to contaminated eggs and egg-based products [23]. Human *salmonellosis* can be effectively controlled through the One Health approach, by reducing prevalence in animals and food products and facilitating outbreak source attribution. To prevent its spread, *Salmonella* is considered as a prioritary pathogen for surveillance under EU Decision 1082/2013/EU3 on serious cross-border threats to health [22].

In order to better control and prevent *salmonellosis* cases in Portugal, and, simultaneously, comply with the new European regulations and guidelines, INSA developed its own allele-based pipeline for routine genomic surveillance of *S. enterica* (PathoGenSurveil, available at https://github.com/insapathogenomics/PathoGenSurveil). With this advance, since January 2025, INSA is performing real-time genomic surveillance of *S. enterica*, being also responsible for the centralization of the One Health *S. enterica* database in the country.

In this context, it was of utmost importance to evaluate how this new pipeline compares with that of

European agencies, as, although INSA followed international guidelines when designing the surveillance workflow, each pipeline has its own particularities (Table 4.2).

Table 4.2: Summary of the main specificities of INSA and EFSA pipelines for *S. enterica* genomic surveillance and details about the dataset samples passing quality control (QC), as well as about the method used for the dataset clustering analysis.

Steps	**INSA**	**EFSA**
Pipeline for quality control and assembly	INNUca v4.2.2 [28, 37]	Aquamis [11]
Allele caller	chewBBACA v3.3.5 [60]	chewBBACA v2.8.5 [60]
Schema used for allele calling	wgMLST INNUENDO (8522 loci) [37]	cgMLST INNUENDO (3255 loci) [37]
Schema used for clustering	cgMLST INNUENDO (3255 loci) [37]	cgMLST INNUENDO (3255 loci) [37]
Clustering pipeline	ReporTree v2.5.4 [44]	-
DATASET ANALYSIS		
Samples passing loci-called filter	2946	2831
Samples passing QC in both	2821	
Clustering	ReporTree v2.5.4 with GrapeTree [44, 73]	

Taking the chance that, contrary to the ECDC pipeline, the EFSA pipeline is publicly available and a facilitator of its local deployment was recently released (https://github.com/insapathogenomics/cml_efsa.wgs.onehealth_facilitator), in the frame of this thesis project, EvalTree was used to perform such analysis, thus supporting the implementation of this pipeline in the national surveillance system. To this end, each pipeline (INSA and EFSA) were run over the publicly available *S. enterica* dataset with 2974 isolates [42, 43, 45]. While 2946 of the isolates passed INSA's pipeline quality control, 2831 passed EFSA's pipeline quality control. This corresponds to 0.94% and 4.80% isolates being filtered out by the INSA and EFSA pipelines, respectively, indicating that the INSA pipeline applies less stringent quality control parameters. Therefore, for further evaluation, both allelic matrices were filtered to have the same set of samples, resulting in 2821 common samples (Table 4.2). For this reason, before EvalTree's evaluation, genetic clustering was repeated for each pipeline, using the filtered matrices. This additional step relied on ReporTree [44] using GrapeTree MSTreeV2 [73] algorithm, the standard clustering method in both pipelines [25].

EvalTree was then used to compare the clustering results of both pipelines and evaluate their congruence. Notably, despite the differences in the pipeline workflow (Table 4.2), INSA and EFSA pipelines produced a very similar number of possible threshold levels (3045 and 3059 thresholds, respectively) and partitions per threshold (Figure 4.9).

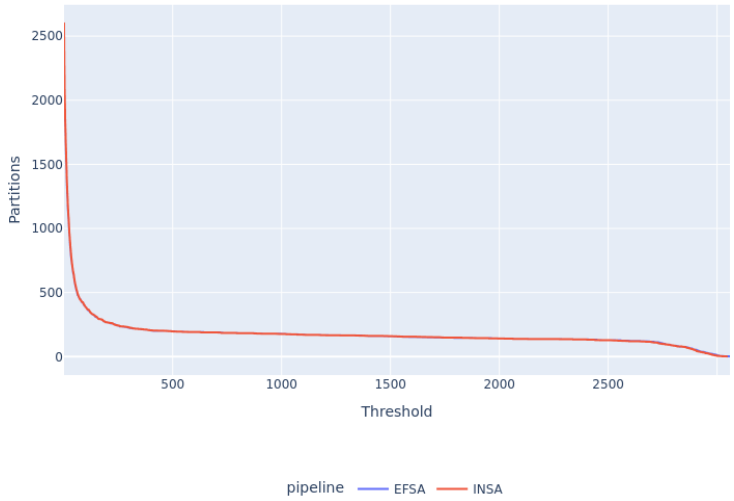


Figure 4.9: Screenshot obtained from EvalTree's HTML report. Number of partitions in each pipeline at each possible threshold.

Consistently with this, both pipelines revealed similar clustering stability regions (Figure 4.10), with the largest region in the INSA pipeline lying within 1068 - 1930 ADs and EFSA's in 1078 - 1953 ADs. Altogether, these results suggest that both pipelines capture similar populational structure. Interestingly, the stability blocks identified by EvalTree for INSA and EFSA pipelines are consistent with the study by [36]. These authors analysed large volumes of *in silico* cgMLST profiles for *Salmonella* lineage to identify stability regions using the nAWC. The stability regions identified ranged between thresholds 1192 and 1885, with a size 693 ADs, showing high concordance with the results retrieved by EvalTree. The largest stability regions identified in our study also correspond closely to those detected in [45].

Regarding the inter-pipeline cluster congruence, EvalTree results revealed a very high congruence at all possible threshold levels (Figure 4.11), with maximum CS being achieved at 3, observed across several threshold levels (for example in the INSA pipeline between 1069 and 1096, and in the EFSA pipeline from 1078 to 1104). This is also clearly demonstrated by the existence of inter-pipeline corresponding points (CS ≥ 2.85) across almost all thresholds levels, with their linear tendency line presenting an $r^2 > 0.999$ and a slope of 0.99 in both directions (Figure 4.12), reflecting excellent agreement between pipelines.

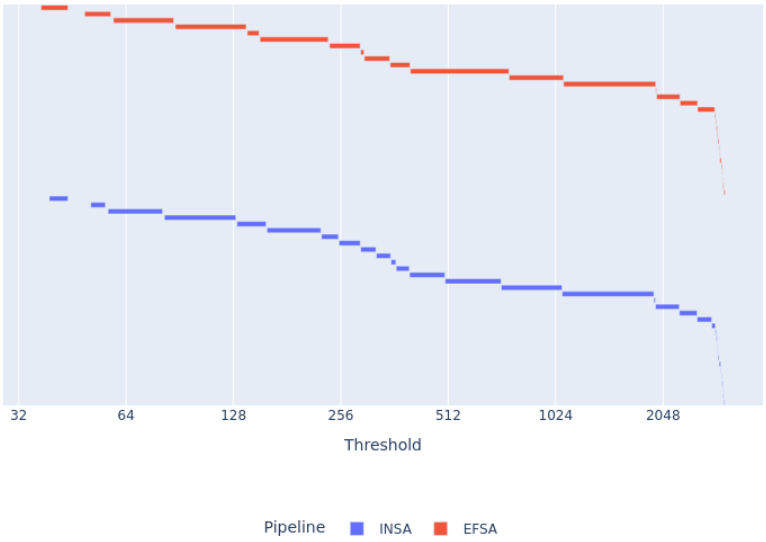


Figure 4.10: Screenshot obtained from EvalTree's HTML report. Clustering stability regions determined for each pipeline. To understand the stability regions block, they are vertically phased, beginning in a different line. Thresholds (x axis) are converted in $\log_2$ scale.

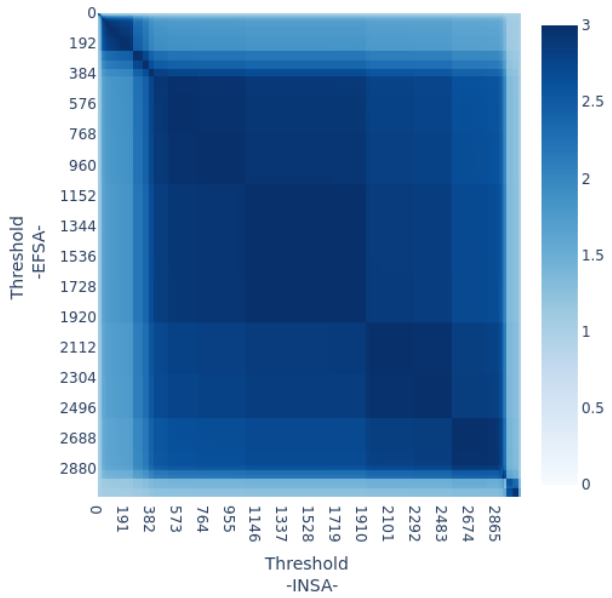


Figure 4.11: Screenshot obtained from EvalTree's HTML report. Cluster congruence at all possible distance thresholds. The congruence score is defined based on the adjusted Wallace coefficient (AWC) and an Adjust Rand (AR), and ranges from 0 (low congruence - blank color) to 3 (total congruence - dark blue color) [57].

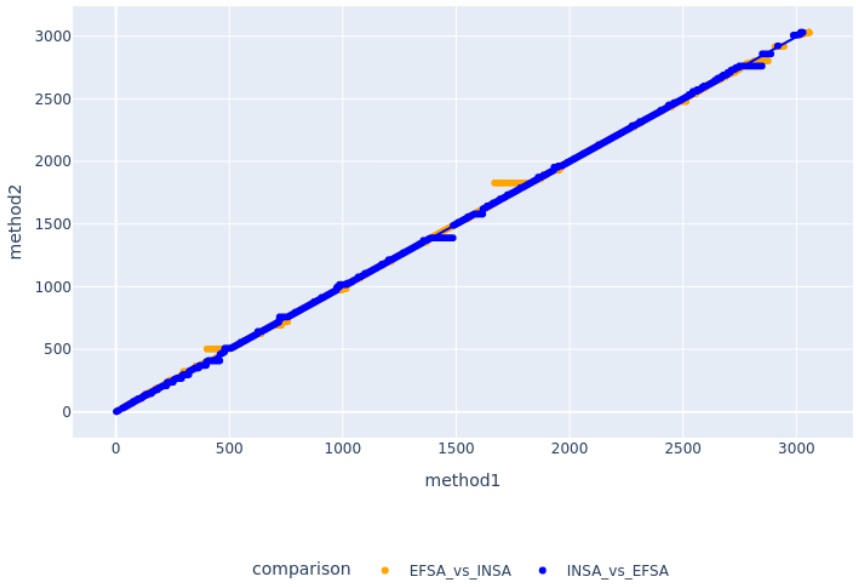


Figure 4.12: Screenshot obtained from EvalTree's HTML report. Identification of corresponding points producing similar clusters across both pipelines. Orange line: the threshold in the INSA pipeline (method 2) that provides the most similar clustering results to a given threshold in the EFSA pipeline (method 1). Blue line: the threshold in the EFSA pipeline (method 2) that yields clustering most similar to the INSA pipeline (method 1).

Having determined that INSA's and EFSA's pipelines present an overall high congruence at all resolution levels, it was important to clarify whether both were able to report similar outbreak signals. To this end, EvalTree's initial run requested an "outbreak analysis" using a 10 ADs threshold, a commonly used threshold to detect and report potential outbreak clusters [6], and the one routinely used by both surveillance systems.

EvalTree's results show that the INSA and EFSA pipelines identified 232 and 235 clusters, respectively, using a fixed threshold of 10 ADs (Table 4.3).

Table 4.3: Pairwise comparison of cluster overlap between the INSA and EFSA pipelines using a fixed threshold (Thr 10 = Thr 10).

Pipelines	**Total clusters**	**Common (exact composition)**	**Exclusives**
INSA	232	209 (90.09% = 209/232)	23 (9.91% = 23/232)
EFSA	235	209 (88.94% = 209/235)	26 (11.06% = 26/235)

INSA and EFSA pipelines identified 23 and 26 exclusive clusters, respectively. A total of 209 clusters were commonly detected with exact composition by both pipelines at a fixed threshold of 10

ADs, corresponding to 90.09% (209/232) and 88.94% (209/235) of the clusters detected by INSA and EFSA pipelines, respectively (Figure 4.13).

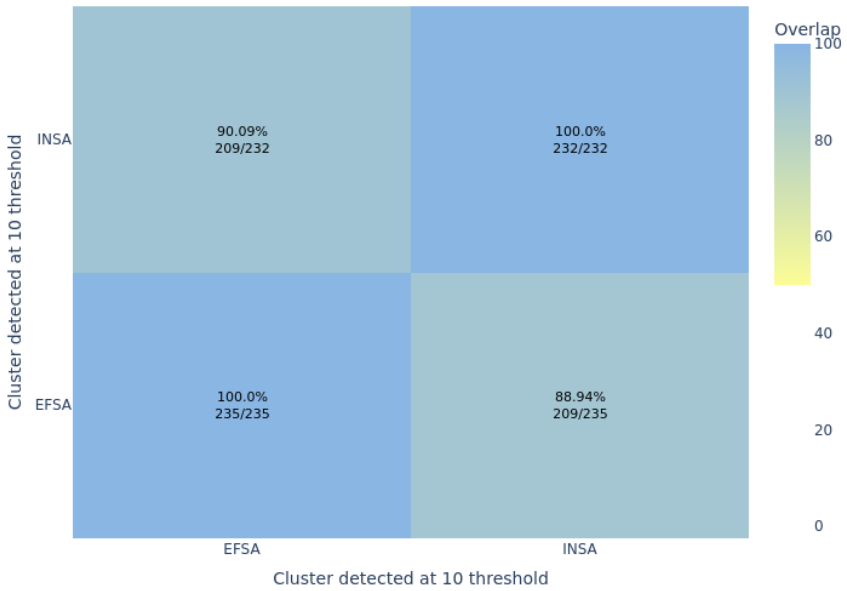


Figure 4.13: Screenshot obtained from EvalTree's HTML report. Percentage the clusters detected at 10 allelic distance threshold in one pipeline with the exact same genetic composition in another pipeline.

Although this value points that 1 in each 10 outbreak signals provided by one pipeline is not captured by the other, in reality this interpretation should take into consideration that this is a very strict approach in which only clusters with the exact same composition (i.e. exact same set of samples) at a fixed threshold are considered as similar signals.

As demonstrated by [45], relaxing the 1 AD threshold favors the detection of outbreak signals when pipelines rely on the same cgMLST scheme. Given this scenario, a new outbreak analysis was subsequently performed.

Therefore, taking advantage of EvalTree's argument '-rto', which allows repeating just the "outbreak analysis" of a previous EvalTree run (Table 4.1), the overlap between clusters detected at 10 ADs by one pipeline and those detected at up to 11 ADs by the other was assessed. This approach showed that 93.97% (218/232) of INSA clusters were identified by the EFSA pipeline at up 11 AD, and 93.62% (220/235) of the EFSA clusters were identified by INSA pipeline at up 11 AD (Table 4.4, Figure 4.14).

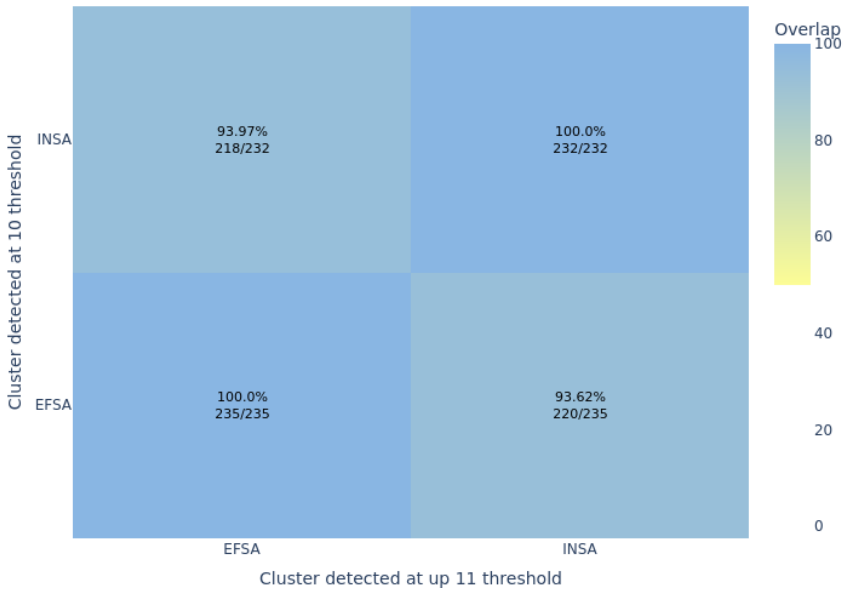


Figure 4.14: Screenshot obtained from EvalTree's HTML report. Percentage the clusters detected at 10 allelic distance threshold in one pipeline with the exact same genetic composition in another pipeline at up 11 ADs.

Table 4.4: Pairwise comparison of cluster overlap between the INSA and EFSA pipelines using a dynamic threshold (Thr 10 <= Thr 11).

Pipelines	**Total clusters**	**Common (exact composition)**	**Exclusives**
INSA	232	218 (93.97% = 218/232)	14 (6.03% = 14/232)
EFSA	235	220 (93.62% = 220/235)	15 (6.38% = 25/235)

A similar exercise using a commonly used threshold at a higher level of resolution, i.e. 5 ADs [45, 6], demonstrated that 93.09% of INSA clusters at 5 AD threshold were identified in the EFSA pipeline up to 6 AD threshold, while 92.31% of EFSA clusters were detected up to 6 AD threshold in the INSA pipeline (Figure 4.15, Table 4.5).

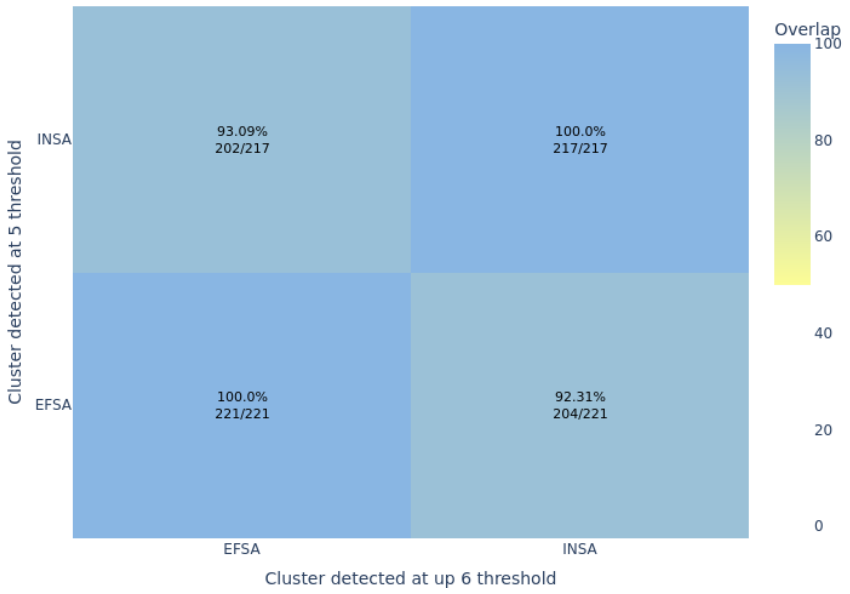


Figure 4.15: Screenshot obtained from EvalTree's HTML report. Percentage the clusters detected at 5 allelic distance threshold in one pipeline with the exact same genetic composition in another pipeline at up 6 ADs.

Table 4.5: Pairwise comparison of cluster overlap between the INSA and EFSA pipelines using a dynamic threshold (Thr 5 <= Thr 6).

Pipelines	**Total clusters**	**Common (exact composition)**	**Exclusives**
INSA	217	202 (93.09% = 202/217)	15 (6.91% = 15/217)
EFSA	221	204 (92.31% = 204/221)	17 (7.69% = 17/221)

Altogether, these results showcase a very high concordance between the outbreak signals reported by INSA and EFSA pipelines, even at outbreak level. This is particularly evident when comparing these values with the results found by [45], where, for example, the average overlap in outbreak signals in tested allele-based pipelines up to 10 ADs was 85%.

In Figure 4.16, a zoom-in of the cluster congruence at AD = 10 is presented, showing a high cluster congruence score (CS=2.894261) between pipelines. A list of corresponding points/thresholds is presented in Figure 4.17. The results shown at AD 10 in the INSA pipeline corresponds to 10 AD in the EFSA pipeline, with very similar clustering outcomes. This indicates that surveillance results will be comparable between the two laboratories.

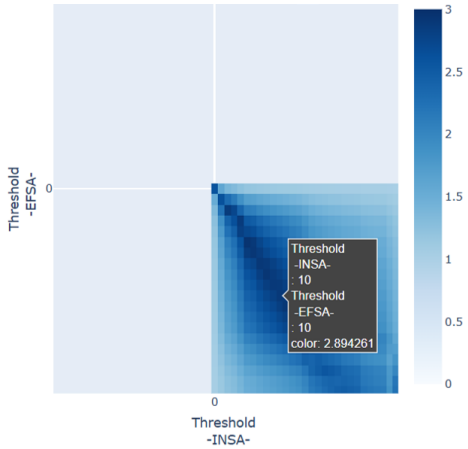


Figure 4.16: Screenshot obtained from EvalTree's HTML report. Cluster congruence at 10 allelic distance thresholds.

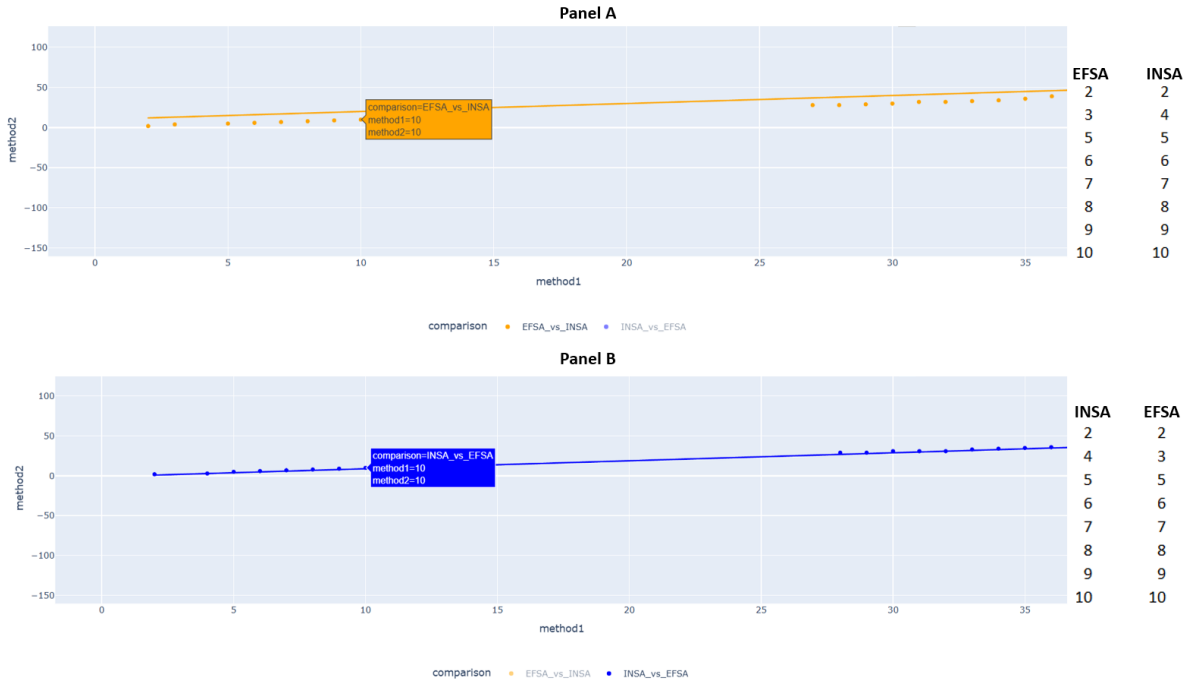


Figure 4.17: Screenshot obtained from EvalTree's HTML report. Identification of corresponding points (CS $\geq$ 2.85) producing similar clusters at 10 allelic distance thresholds in both pipelines. Panel A: threshold in the INSA pipeline (method 2) that provides the most similar clustering results in the EFSA pipeline (method 1); Panel B: threshold in the EFSA pipeline (method 2) that yields clustering results most similar to the INSA pipeline (method 1).

Chapter 5

Conclusion

EvalTree is an innovative tool for analysing clustering data from two typing pipelines (traditional typing and WGS), enabling the assessment of congruence between pipelines. This tool is fully automated and the inter-pipeline clustering congruence analysis can be performed through a single command line. The tool is designed to accept one or two inputs (folders or clustering files). When an input folder is provided, it offers additional features, including cluster characterization with associated metadata and detection of potential outbreak signals. A user-friendly HTML report with dynamic and interactive plots is generated according to the analyses defined by the user. To promote usability and reproducibility, EvalTree is distributed both as a conda package and on GitHub.

The development of this tool aims to support laboratories, for example, in the evaluation and comparison of the performance of different pipelines, in the assessment of new versus old typing schemas, and monitoring the impact of software modifications. Ensuring result comparability among laboratories is particularly important in multi-country outbreak scenarios, where efficient communication between countries is critical for an effective public health response.

In this context, this thesis supports the implementation of *S. enterica* genomic surveillance in Portugal through a comparison between the INSA pipeline and the EFSA pipeline used in Europe. The results revealed high congruence between pipelines, demonstrating that both workflows produce comparable surveillance outcomes.

As future perspectives, EvalTree could benefit from improved execution time, as the long runtime remains a limitation for large and diverse datasets. Future work could explore performance optimizations, such as parallelization of the pairwise comparisons. Additional improvements include implementing a new argument to dynamically select serotypes or sequence type with the highest number of samples; enabling clustering visualization to be executed as an independent module; and extending the tool for multiple pairwise comparisons, enabling the tool to handle more than two inputs. For instance, three input datasets (A, B and C), all pairwise comparisons could be performed (A vs B, A vs C and B vs. C). This can be generalised to any number of inputs (n), with all pairwise combinations being computed automatically, thus further broadening the tool's applicability.

In conclusion, this thesis not only provides a practical solution to the current challenges faced by laboratories in implementing WGS but also fosters interoperability and supports inter-laboratory collaborations, while ensuring the sustainability and long-term maintenance of pipelines. By enabling comparable analyses over time, it contributes to stronger integration of genomic data into public health decision-making within a One Health framework.

References

[1] Afolayan, A. O., Bernal, J. F., Gayeta, J. M., Masim, M. L., Shamanna, V., Abrudan, M., Abudahab, K., Argimón, S., Carlos, C. C., Sia, S., et al. (2021). Overcoming data bottlenecks in genomic pathogen surveillance. *Clinical Infectious Diseases*, 73(Supplement_4):S267–S274. Available at: https://doi.org/10.1093/cid/ciab785. 2

[2] Atxaerandio-Landa, A., Arrieta-Gisasola, A., Laorden, L., Bikandi, J., Garaizar, J., Martinez-Malaxetxebarria, I., and Martinez-Ballesteros, I. (2022). A practical bioinformatics workflow for routine analysis of bacterial wgs data. *Microorganisms*, 10(12):2364. Available at: https://doi.org/10.3390/microorganisms10122364. 1, 2

[3] Babraham Bioinformatics (2010). Fastqc: A quality control tool for high throughput sequence data. Available at: https://www.bioinformatics.babraham.ac.uk/projects/fastqc. 20

[4] Bankevich, A., Nurk, S., Antipov, D., Gurevich, A. A., Dvorkin, M., Kulikov, A. S., Lesin, V. M., Nikolenko, S. I., Pham, D., Prjibelski, A. D., Pyshkin, A. V., Sirotkin, A. V., Vyahhi, N., Tesler, G., Alekseyev, M. A., and Pevzner, P. A. (2012). Spades: a new genome assembly algorithm and its applications to single-cell sequencing. *Journal of Computational Biology*, 19(5):455–477. Available at: https://doi.org/10.1089/cmb.2012.0021. 20

[5] Bedford, T., Neher, R., Hadfield, J., and et al. (2018). Auspice: An open-source interactive tool for visualising phylogenomic data. Available at: https://auspice.us/. XI, 13

[6] Besser, J. M., Carleton, H. A., Trees, E., Stroika, S. G., Hise, K., Wise, M., and Gerner-Smidt, P. (2019). Interpretation of whole-genome sequencing for enteric disease surveillance and outbreak investigation. *Foodborne Pathogens and Disease*, 17(8):507–517. Available at: https://doi.org/10.1089/fpd.2019.2650. 13, 39, 41

[7] bfr_bioinformatics (2025). snippysnake - variant calling pipeline with snippy. Available at: https://gitlab.com/bfr_bioinformatics/snippySnake. 8

[8] Bogaerts, B., Nouws, S., Verhaegen, B., Denayer, S., Van Braekel, J., Winand, R., Fu, Q., Crombé, F., Piérard, D., Marchal, K., Roosens, N. H. C., De Keersmaecker, S. C. J., and Vanneste, K. (2021).

Validation strategy of a bioinformatics whole genome sequencing workflow for shiga toxin-producing *Escherichia coli* using a reference collection extensively characterized with conventional methods. *Microbial Genomics*, 7(3):mgen000531. Available at: https://doi.org/10.1099/mgen.0.000531. 10

[9] Bogaerts, B., Winand, R., Fu, Q., Van Braekel, J., Ceyssens, P.-J., Mattheus, W., Bertrand, S., De Keersmaecker, S. C. J., Roosens, N. H. C., and Vanneste, K. (2019). Validation of a bioinformatics workflow for routine analysis of whole-genome sequencing data and related challenges for pathogen typing in a european national reference center: *Neisseria meningitidis* as a proof-of-concept. *Frontiers in Microbiology*, 10:362. Available at: https://doi.org/10.3389/fmicb.2019.00362. 10

[10] Bolger, A. M., Lohse, M., and Usadel, B. (2014). Trimmomatic: a flexible trimmer for illumina sequence data. *Bioinformatics*, 30(15):2114–2120. Available at: https://doi.org/10.1093/bioinformatics/btu170. 20

[11] Carlus Deneke, Holger Brendebach (2023). Aquamis. Available at: https://gitlab.com/bfr_bioinformatics/AQUAMIS. 36

[12] Carriço, J. A., Silva-Costa, C., Melo-Cristino, J., Pinto, F. R., de Lencastre, H., Almeida, J. S., and Ramirez, M. (2006). Illustration of a common framework for relating multiple typing methods by application to macrolide-resistant *Streptococcus pyogenes*. *Journal of Clinical Microbiology*, 44(7):2524–2532. Available at: https://doi.org/10.1128/JCM.02536-05. 14

[13] Chattaway, M. A., Painset, A., Godbole, G., Gharbia, S., and Jenkins, C. (2023). Evaluation of genomic typing methods in the *Salmonella* reference laboratory in public health, england, 2012–2020. *Pathogens*, 12(2):223. Available at: https://doi.org/10.3390/pathogens12020223. 8, 14

[14] Coipan, C. E., Friesema, I. H., van den Beld, M. J. C., Bosch, T., Schlager, S., van der Voort, M., Frank, C., Lang, C., Fruth, A., and Franz, E. (2022). Sporadic occurrence of enteroaggregative shiga toxin-producing *Escherichia coli* o104:h4 similar to 2011 outbreak strain. *Emerging Infectious Diseases*, 28(9):1890–1894. Available at: https://pmc.ncbi.nlm.nih.gov/articles/PMC9423916/. 10

[15] Dallman, T. J., Byrne, L., Ashton, P. M., Cowley, L. A., Perry, N. T., Adak, G., Petrovska, L., Ellis, R. J., Elson, R., Underwood, A., Green, J., Hanage, W. P., Jenkins, C., Grant, K., and Wain, J. (2015). Whole-genome sequencing for national surveillance of shiga toxin–producing *Escherichia coli* o157. *Clinical Infectious Diseases*, 61(3):305–312. Available at: https://academic.oup.com/cid/article-abstract/61/3/305/491349?redirectedFrom=fulltext. 10

[16] den Bakker, H. C., Moreno Switt, A. I., Cummings, C. A., Hoelzer, K., Degoricija, L., Rodriguez-Rivera, L. D., Wright, E. M., Fang, R., Davis, M., Root, T., Schoonmaker-Bopp, D., Musser, K. A.,

Villamil, E., Waechter, H. N., Kornstein, L., Furtado, M. R., and Wiedmann, M. (2011). A whole-genome single nucleotide polymorphism-based approach to trace and identify outbreaks linked to a common *Salmonella enterica* subsp. *enterica* serovar montevideo pulsed-field gel electrophoresis type. *Applied and Environmental Microbiology*, 77(24):8648–8655. Available at: https://pmc.ncbi.nlm.nih.gov/articles/PMC3233099/. 8

[17] Deneke, C., Brendebach, H., Uelze, L., Borowiak, M., Malorny, B., and Tausch, S. H. (2021). Species-specific quality control, assembly and contamination detection in microbial isolate sequences with aquamis. *Genes*, 12(5):644. Available at: https://doi.org/10.3390/genes12050644. 9, 32

[18] Duval, A., Opatowski, L., and Brisse, S. (2023). Defining genomic epidemiology thresholds for common-source bacterial outbreaks: a modelling study. *Lancet Microbe*, 4(5):e349–e357. Available at: https://doi.org/10.1016/S2666-5247(22)00380-9. 14

[19] European Centre for Disease Prevention and Control (2019). Ecdc strategic framework for the integration of molecular and genomic typing into european surveillance and multi-country outbreak investigations – 2019–2021. Technical report, ECDC. Available at: https://www.ecdc.europa.eu/sites/default/files/documents/framework-for-genomic-surveillance.pdf. 11

[20] European Centre for Disease Prevention and Control (2024). External quality assessment schemes to support european surveillance of legionnaires' disease in eu/eea countries, 2022–2023. Technical report, ECDC, Stockholm. Available at: https://www.ecdc.europa.eu/sites/default/files/documents/legionnaires-external-quality-assessment-2022-2023.pdf. 14

[21] European Centre for Disease Prevention and Control (2025). Twelfth external quality assessment scheme for typing of shiga toxin-producing *Escherichia coli*. Technical report, ECDC, Stockholm. Available at: https://www.ecdc.europa.eu/sites/default/files/documents/EQA-20250128-88.pdf. 14

[22] European Centre for Disease Prevention and Control (ECDC) (2024). Thirteenth external quality assessment for *Salmonella* typing. Technical report, European Centre for Disease Prevention and Control, Stockholm. Available at: https://www.ecdc.europa.eu/sites/default/files/documents/EQA-13-Salmonella-typing.pdf. 10, 14, 35

[23] European Centre for Disease Prevention and Control (ECDC) and European Food Safety Authority (EFSA) (2017). Multi-country outbreak of *Salmonella* enteritidis infections linked to polish eggs. Technical report, ECDC/EFSA, Stockholm/Parma. Available at: https://ecdc.europa.eu/sites/portal/files/documents/12-12-2017-RRA-UPDATE-4-Salmonella-Enteritidis_0.pdf. 35

[24] European Centre for Disease Prevention and Control (ECDC) and European Food Safety Authority (EFSA) (2019). Efsa and ecdc technical report on the collection and analysis of whole genome sequencing data from food-borne pathogens and other relevant microorganisms isolated from human, animal, food, feed and food/feed environmental samples in the joint ecdc-efsa molecular typing database. Technical Report EN-1337, EFSA Supporting Publications. Available at: https://doi.org/10.2903/sp.efsa.2019.EN-1337. 10

[25] European Food Safety Authority (EFSA) (2022). Guidelines for reporting Whole Genome Sequencing-based typing data through the EFSA One Health WGS System. Technical Report EN-7413, European Food Safety Authority (EFSA). Available at: https://doi.org/10.2903/sp.efsa.2022.EN-7413. 1, 3, 9, 10, 11, 21, 36

[26] European Food Safety Authority (EFSA) and European Centre for Disease Prevention and Control (ECDC) (2024). The european union one health 2023 zoonoses report. *EFSA Journal*, 22:e09106. Available at: https://doi.org/10.2903/j.efsa.2024.9106. 35

[27] Feijao, P., Yao, H.-T., Fornika, D., Gardy, J., Hsiao, W., Chauve, C., and Chindelevitch, L. (2018). Mentalist - a fast mlst caller for large mlst schemes. *Microbial Genomics*, 4(2):e000146. Available at: https://doi.org/10.1099/mgen.0.000146. 9

[28] GitHub-B-UMMI (2023). INNUca: Innuendo quality control of reads, *de novo* assembly and contigs quality assessment, and possible contamination search. Available at: https://github.com/B-UMMI/INNUca. 36

[29] Glasgow, H. L., Zheng, Y., Brazelton, J. N., Tang, L., and Hayden, R. T. (2025). Comparison of core genome multi-locus sequencing typing pipelines for hospital outbreak detection of common bacterial pathogens. *Journal of Clinical Microbiology*. Available at: https://journals.asm.org/doi/10.1128/jcm.00646-25. 10

[30] Gomes, E., Araújo, D., Nogueira, T., Oliveira, R., Silva, S., Oliveira, L. V. N., Azevedo, N. F., Almeida, C., and Castro, J. (2025). Advances in whole genome sequencing for foodborne pathogens: implications for clinical infectious disease surveillance and public health. *Frontiers in Cellular and Infection Microbiology*, 15:1593219. Available at: https://doi.org/10.3389/fcimb.2025.1593219. 10

[31] Graham, R. L. and Hell, P. (1985). On the history of the minimum spanning tree problem. *Annals of the History of Computing*, 7:43–57. 12

[32] Kaas, R. S., Leekitcharoenphon, P., Aarestrup, F. M., and Lund, O. (2014). Solving the problem of comparing whole bacterial genomes across different sequencing platforms. *PloS One*, 9(8):e104984. Available at: https://doi.org/10.1371/journal.pone.0104984. 8

[33] Lavanya, A., Gaurav, L., Sindhuja, S., Seam, H., Joydeep, M., Uppalapati, V., Ali, W., and VidyaSagar (2023). Assessing the performance of python data visualization libraries: A review. *International Journal of Computer Engineering in Research Trends*, 10(1):28–39. Available at: https://doi.org/10.22362/ijcert/2023/v10/i01/v10i0104. 18

[34] Leeper, M. M., Schroeder, M. N., Griswold, T., Thakur, M., Krishnan, K., Katz, L. S., Hise, K. B., Williams, G. M., Stroika, S. G., Im, S. B., Lindsey, R. L., Smith, P. A., Huffman, J., Kelley, A., Cleland, S., Collins, A. J., Gautam, S., Tyagi, E., Park, S., and Carriço, J. A. (2025). Validation of core and whole-genome multi-locus sequence typing schemes for shiga-toxin-producing *Escherichia coli* (stec) outbreak detection in a national surveillance network, pulsenet 2.0, usa. *Microorganisms*, 13(6):1310. Available at: https://www.mdpi.com/2076-2607/13/6/1310. 10

[35] Leeper, M. M., Tolar, B. M., Griswold, T., Vidyaprakash, E., Hise, K. B., Williams, G. M., Im, S. B., Chen, J. C., Pouseele, H., and Carleton, H. A. (2023). Evaluation of whole and core genome multilocus sequence typing allele schemes for *Salmonella enterica* outbreak detection in a national surveillance network, pulsenet usa. *Frontiers in Microbiology*, 14:1254777. Available at: https://doi.org/10.3389/fmicb.2023.1254777. 10, 14

[36] Liu, C. C. and Hsiao, W. W. L. (2022). Large-scale comparative genomics to refine the organization of the global *Salmonella enterica* population structure. *Microbial Genomics*, 8(12):000906. Available at: https://doi.org/10.1099/mgen.0.000906. 37

[37] Llarena, A.-K. et al. (2018). Innuendo: A cross-sectoral platform for the integration of genomics in the surveillance of food-borne pathogens. *EFSA Supporting Publications*, 15:EN–1498. Available at: https://efsa.onlinelibrary.wiley.com/doi/abs/10.2903/sp.efsa.2018.EN-1498. 36

[38] Maiden, M. C., Bygraves, J. A., Feil, E., Morelli, G., Russell, J. E., Urwin, R., Zhang, Q., Zhou, J., Zurth, K., Caugant, D. A., Feavers, I. M., Achtman, M., and Spratt, B. G. (1998). Multilocus sequence typing: a portable approach to the identification of clones within populations of pathogenic microorganisms. *Proceedings of the National Academy of Sciences of the United States of America*, 95(6):3140–3145. Available at: https://doi.org/10.1073/pnas.95.6.3140. 7

[39] Maiden, M. C. J. (2006). Multilocus sequence typing of bacteria. *Annual Review of Microbiology*, 60(1):561–588. Available at: https://doi.org/10.1146/annurev.micro.59.030804.121325. 7, 9

[40] Maiden, M. C. J., Jansen van Rensburg, M. J., Bray, J. E., Earle, S. G., Ford, S. A., Jolley, K. A., and McCarthy, N. D. (2013). Mlst revisited: the gene-by-gene approach to bacterial genomics. *Nature Reviews Microbiology*, 11(10):728–736. Available at: https://doi.org/10.1038/nrmicro3093. 7, 8, 9

[41] Mamede, R., Vila-Cerqueira, P., Silva, M., Carriço, J. A., and Ramirez, M. (2021). Chewie nomenclature server (chewie-ns): a deployable nomenclature server for easy sharing of core and whole genome mlst schemas. *Nucleic Acids Research*, 49(D1):D660–D666. Available at: https://doi.org/10.1093/nar/gkaa889. 9, 20

[42] Mixão, V., Pinto, M., Gomes, J. a. P., Sobral, D., Brendebach, H., Deneke, C., Tausch, S., Di Pasquale, A., Swart-Coipan, C., Iwan, E., Linde, J., Lagesen, K., Petrovska, L., Umaer Naseer, M., Sommer Kaas, R., Simon, S., Joensen, K., Kiil, K., Nielsen, S., and Borges, V. (2023). The OHEJP BeONE Project – *Salmonella enterica* genome assembly dataset. Zenodo, April 5, 2023, version 2. Available at: https://doi.org/10.5281/zenodo.7267785. 3, 12, 18, 20, 21, 33, 36

[43] Mixao, V. (2022). Genome assemblies and respective wg/cgmlst profiles of a diverse dataset comprising 1,434 *Salmonella enterica* isolates. *Zenodo*. Available at: https://doi.org/10.5281/ZENODO.7119735. 3, 12, 18, 20, 21, 33, 36

[44] Mixão, V., Pinto, M., Sobral, D., Di Pasquale, A., Gomes, J. P., and Borges, V. (2023). Reportree: a surveillance-oriented tool to strengthen the linkage between pathogen genetic clusters and epidemiological data. *Genome Medicine*, 15(1):43. Available at: https://doi.org/10.1186/s13073-023-01196-1. XI, 12, 13, 15, 20, 21, 23, 36

[45] Mixão, V., Pinto, M., Brendebach, H., Sobral, D., Dourado Santos, J., Radomski, N., Majgaard Uldall, A. S., Bomba, A., Pietsch, M., Bucciacchio, A., de Ruvo, A., Castelli, P., Iwan, E., Simon, S., Coipan, C. E., Linde, J., Petrovska, L., Kaas, R. S., Grimstrup Joensen, K., Borges, V., et al. (2025). Multi-country and intersectoral assessment of cluster congruence between pipelines for genomics surveillance of foodborne pathogens. *Nature Communications*, 16(1):3961. Available at: https://doi.org/10.1038/s41467-025-59246-8. 2, 3, 14, 15, 16, 17, 18, 19, 20, 27, 32, 33, 36, 37, 40, 41, 42

[46] Mixão, V., Santos, J. D., and Borges, V. (2024). Facilitator of the local deployment of the EFSA One Health Whole-Genome Sequencing analytical pipeline. GitHub repository, https://github.com/insapathogenomics/cml_efsa.wgs.onehealth_facilitator. Accessed on October 16, 2025. 3

[47] Mustafa, A. S. (2024). Whole genome sequencing: Applications in clinical bacteriology. *Medical Principles and Practice*, 33(3):185–197. Available at: https://doi.org/10.1159/000538002. 1

[48] Nadon, C., Van Walle, I., Gerner-Smidt, P., Campos, J., Chinen, I., Concepcion-Acevedo, J., Gilpin, B., Smith, A. M., Kam, K. M., Perez, E., Trees, E., Kubota, K., Takkinen, J., Nielsen, E. M., Carleton, H., and Panel, F.-N. E. (2017). Pulsenet international: Vision for the implementation of whole genome sequencing (wgs) for global food-borne disease surveillance. *Eurosurveillance*, 22(23):30544. Available at: http://dx.doi.org/10.2807/1560-7917.ES.2017.22.23.30544. 10

[49] Naghavi, M., Vollset, S. E., Ikuta, K. S., Swetschinski, L. R., Gray, A. P., Wool, E. E., Aguilar, G. R., Mestrovic, T., Smith, G., Han, C., et al. (2024). Global burden of bacterial antimicrobial resistance 1990–2021: a systematic analysis with forecasts to 2050. *The Lancet*, 404(10459):1199–1226. Available at: http://doi.org/10.1016/S0140-6736(24)01867-1. 1

[50] Norouzi, M., Fleet, D. J., and Salakhutdinov, R. R. (2012). Hamming distance metric learning. In *Advances in Neural Information Processing Systems*, pages 1061–1069. 12

[51] Oakley, B. B., Gonzalez-Escalona, N., and Molina, M. (2015). Molecular typing and differentiation. In *Compendium of Methods for the Microbiological Examination of Foods*, chapter 12. American Public Health Association. Available at: https://doi.org/10.2105/mbef.0222.017. 7

[52] Ramadan, A. A. (2022). Bacterial typing methods from past to present: A comprehensive overview. *Gene Reports*, 29:101675. Available at: https://doi.org/10.1016/j.genrep.2022.101675. 7, 8, 9

[53] Schadron, T., van den Beld, M., Mughini-Gras, L., and Franz, E. (2024). Use of whole genome sequencing for surveillance and control of foodborne diseases: status quo and quo vadis. *Frontiers in Microbiology*, 15:1460335. Available at: https://doi.org/10.3389/fmicb.2024.1460335. 10, 13, 14

[54] Schürch, A. C., Arredondo-Alonso, S., Willems, R. J. L., and Goering, R. V. (2018). Whole genome sequencing options for bacterial strain typing and epidemiologic analysis based on single nucleotide polymorphism versus gene-by-gene–based approaches. *Clinical Microbiology and Infection*, 24(4):350–354. Available at: https://doi.org/10.1016/j.cmi.2017.12.016. 8

[55] Seemann, T. (2017). Shovill: faster spades assembly of illumina reads. Available at: https://github.com/tseemann/shovill. 9

[56] Segerman, B., Skarin, H., and Ástvaldsson, Á. (2024). Guidance document for cluster analysis of whole genome sequence data. Technical report, European Union Reference Laboratory for *Campylobacter*, Swedish Veterinary Agency, Uppsala, Sweden. Available at: https://zenodo.org/records/14536255. 8, 9, 10

[57] Severiano, A., Pinto, F. R., Ramirez, M., and Carriço, J. A. (2011). Adjusted wallace coefficient as a measure of congruence between typing methods. *Journal of Clinical Microbiology*, 49(11):3997–4000. Available at: https://doi.org/10.1128/JCM.00624-11. XII, 14, 15, 38

[58] Shen, Y., Liu, Y., Krafft, T., and Wang, Q. (2025). Progress and challenges in infectious disease surveillance and early warning. *Medicine Plus*, 2(1):100071. Available at: https://doi.org/10.1016/j.medp.2025.100071. 10

[59] Sheppard, S. K., Jolley, K. A., and Maiden, M. C. J. (2012). A gene-by-gene approach to bacterial population genomics: Whole genome mlst of *Campylobacter*. *Genes*, 3(2):261–277. Available at: https://doi.org/10.3390/genes3020261. 8, 9

[60] Silva, M., Machado, M. P., Silva, D. N., Rossi, M., Moran-Gilad, J., Santos, S., Ramirez, M., and Carriço, J. A. (2018). chewbbaca: A complete suite for gene-by-gene schema creation and strain identification. *Microbial Genomics*, 4(3). Available at: https://doi.org/10.1099/mgen.0.000166. 9, 20, 36

[61] Spada, E., Sagliocca, L., Sourdis, J., Garbuglia, A. R., Poggi, V., De Fusco, C., and Mele, A. (2004). Use of the minimum spanning tree model for molecular epidemiological investigation of a nosocomial outbreak of hepatitis c virus infection. *Journal of Clinical Microbiology*, 42:4230–4236. Available at: https://doi.org/10.1128/JCM.42.9.4230-4236.2004. 12

[62] Struelens, M. J., Ludden, C., Werner, G., Sintchenko, V., Jokelainen, P., and Ip, M. (2024). Real-time genomic surveillance for enhanced control of infectious diseases and antimicrobial resistance. *Frontiers in Science*, 2:1298248. Available at: https://doi.org/10.3389/fsci.2024.1298248. 1, 2

[63] Tang, S., Orsi, R. H., Luo, H., Ge, C., Zhang, G., Baker, R. C., Stevenson, A., and Wiedmann, M. (2019). Assessment and comparison of molecular subtyping and characterization methods for *Salmonella*. *Frontiers in Microbiology*, 10:1591. Available at: https://doi.org/10.3389/fmicb.2019.01591. 8, 9

[64] Tenover, F. C., Arbeit, R. D., Goering, R. V., and Molecular Typing Working Group of the Society for Healthcare Epidemiology of America (1997). How to select and interpret molecular strain typing methods for epidemiological studies of bacterial infections: A review for healthcare epidemiologists. *Infection Control and Hospital Epidemiology*, 18(6):426–439. Available at: https://doi.org/10.2307/30141252. 7

[65] Uelze, L., Grützke, J., Borowiak, M., Hammerl, J. A., Juraschek, K., Deneke, C., Tausch, S. H., and Malorny, B. (2020). Typing methods based on whole genome sequencing data. *One Health Outlook*, 2:1–19. Available at: https://doi.org/https://doi.org/10.1186/s42522-020-0010-1. 1, 7, 8, 9, 10, 11

[66] UNHCR (2020). Emergence handbook – disease surveillance threshold. https://www.unhcr.org/emergency-handbook. Accessed: 2025-09-01. 13

[67] van Belkum, A., Tassios, P. T., Dijkshoorn, L., Haeggman, S., Cookson, B., Fry, N. K., Fussing, V., Green, J., Feil, E., Gerner-Smidt, P., Brisse, S., Struelens, M., and European Society of Clinical Microbiology and Infectious Diseases (ESCMID) Study Group on Epidemiological Markers (ESGEM) (2007). Guidelines for the validation and application of typing methods for use

in bacterial epidemiology. *Clinical Microbiology and Infection*, 13(Suppl 3):1–46. Available at: https://doi.org/10.1111/j.1469-0691.2007.01786.x. 7

[68] Wood, D. E. and Salzberg, S. L. (2014). Kraken: ultrafast metagenomic sequence classification using exact alignments. *Genome Biology*, 15(3):R46. Available at: https://doi.org/10.1186/gb-2014-15-3-r46. 9, 20

[69] World Health Organization (WHO) (2023a). Whole genome sequencing as a tool to strengthen foodborne disease surveillance and response – module 1: Introductory module. Technical report. Available at: https://iris.who.int/bitstream/handle/10665/373459/9789240021228-eng.pdf?sequence=1. 10

[70] World Health Organization (WHO) (2023b). Whole genome sequencing as a tool to strengthen foodborne disease surveillance and response – module 2: Whole genome sequencing in foodborne disease outbreak investigations. Technical report. Available at: https://iris.who.int/bitstream/handle/10665/373460/9789240021242-eng.pdf?sequence=1. 10

[71] World Health Organization (WHO) (2023c). Whole genome sequencing as a tool to strengthen foodborne disease surveillance and response – module 3: Whole genome sequencing in foodborne disease routine surveillance. Technical report. Available at: https://iris.who.int/bitstream/handle/10665/373522/9789240021266-eng.pdf?sequence=1. 10

[72] World Organisation for Animal Health (WOAH) (2018). Standards for high throughput sequencing, bioinformatics and computational genomics. Technical report. Available at: https://www.woah.org/fileadmin/Home/fr/Health_standards/tahm/1.01.07_HTS_BGC.pdf. 10

[73] Zhou, Z., Alikhan, N.-F., Sergeant, M. J., Luhmann, N., Vaz, C., Francisco, A. P., Carriço, J. A., and Achtman, M. (2018). Grapetree: visualization of core genomic relationships among 100,000 bacterial pathogens. *Genome Research*, 28(9):1395–1404. Available at: https://doi.org/10.1101/gr.232397.117. XI, 11, 12, 20, 21, 33, 36

Appendix A

Extra Information

This chapter provides supplementary information. It describes the usage arguments of the EvalTree.py script, as well as the script developed for final score validation.

A.1 Description of the usage arguments of the EvalTree.py

-h, –help Show this help message and exit.

-v, –version [OPTIONAL] Specify the version number of EvalTree.

-i1 INPUT1, –input1 INPUT1
[MANDATORY] Specify the first input type (folder or file) and requires the full path. If a folder is provided, it must contain the partition table with clustering data at all possible threshold levels. Alternatively, individual clustering table files from traditional methods or WGS-based methods can be supplied.

-i2 INPUT2, –input2 INPUT2
[OPTIONAL] Specify the second input type (folder or file) and requires the full path. If a folder is provided, it must contain the partition table with clustering data at all possible threshold levels. Alternatively, individual clustering table files from traditional methods or WGS-based methods can be supplied.

-o OUTPUT, –output OUTPUT
[OPTIONAL] Specify the output directory for storing all analyses results. If no folder is provided, the program will automatically create one based on the prefix of the files.

-s SCORE, –score SCORE
[OPTIONAL] Define a minimum score to consider two partitions (one from each pipeline) as corresponding. The score accepts values between 0 (low congruence) and 3 (high congruence).

-t THRESHOLD, –threshold THRESHOLD

[OPTIONAL] Define an integer threshold range to select or filter threshold columns from the partition table file. A filtered partition table, containing only the selected columns (thresholds), will be created and used for subsequent analysis. Threshold ranges are specified using a hyphen to separate the minimum and maximum values (e.g., 10-20). If this option is not set, the script will perform clustering for all possible thresholds in the range 0 to the maximum threshold.

-ps partitions_summary, sample_of_interest, –plots_summary partitions_summary, sample_of_interest

[OPTIONAL] Specify the type of cluster characterization file (**partitions_summary.tsv* or *SAMPLES_OF_INTEREST_partitions_summary.tsv*). Both files are expected to be located within a Reportree folder. Using the partition_summary option, the largest clusters present in the file will be characterized. Alternatively, the samples_of_interest option will characterize all clusters, including those resulting from the addition of new samples (kept increase, new, new (increase), new (merge_increase), new (split_increase), new (split_merge_increase)).

-n N_CLUSTER, –n_cluster N_CLUSTER

[OPTIONAL] Specify the number of top clusters to be displayed from the partitions_summary.tsv file, which must be located within a Reportree folder. This argument is not applicable when using the samples_of_interest option.

-cp COLUMNS_PLOTS, –columns_plots COLUMNS_PLOTS

[OPTIONAL] Name(s) of the column(s) to process the characterization of the clustering data in the selected file (specified by the plots_summary argument). For multiple column names, indicate them separated by commas without spaces (e.g., column1,column2).

-pt PLOTS_THRESHOLD, –plots_threshold PLOTS_THRESHOLD

[OPTIONAL] Identify the integer threshold(s) to be applied to the file specified by the plots_summary argument. For multiple thresholds, indicate them separated by commas without spaces (e.g., X,Y,Z). This generates a pie chart showing the clustering data for the specified threshold(s), according to the columns_plot argument.

-pcn PLOTS_CATEGORY_NUMBER, –plots_category_number PLOTS_CATEGORY_NUMBER

[OPTIONAL] Determines the number of plot categories in the **partitions_summary.tsv* or **SAMPLES_OF_INTEREST_partitions_summary.tsv* file that are intended to be collapsed into an unique slice, named 'Other', for visualization in the cluster plots. When there are more than 5 slices (default), they

will be combined into one slice, named 'Other'.

-pcp PLOTS_CATEGORY_PERCENTAGE, –plots_category_percentage PLOTS_CATEGORY_PERCENTAGE

[OPTIONAL] Determines the percentage of plot categories in the **partitions_summary.tsv* or **SAMPLES_OF_INTEREST_partitions_summary.tsv* file that are intended to be collapse into the slice called 'Other' for visualization in the cluster plots. Slices plots with a lower percentage than the entered plots_category_percentage will be combined into one slice named 'Other'.

-to THRESHOLD_OUTBREAK, –threshold_outbreak THRESHOLD_OUTBREAK

[OPTIONAL] Determine the number of clusters identified in one pipeline (rows) at a given threshold that were also detected with the exact same cluster composition at a (or up to a) given threshold in the other pipeline (columns).

This argument has a specific structure: it requires two thresholds and the type of comparison (fixed or dynamic threshold).

Threshold1: Threshold at which the genetic clusters are identified in the pipeline of interest.

Threshold2: Threshold at which the genetic clusters must be searched in the other pipeline.

Comparison (fixed or dynamic):

- *fixed*: Direct cluster comparison at specific threshold between two pipelines. Use a comma to separate threshold1 and threshold2. Example: 7,7.

- *dynamic*: Range of threshold to find genetic clusters with similar composition in the other pipeline. Use <= to indicate a dynamic threshold range. Example of expression: 7,<=9.

For multiple threshold pairs, separate them with a semicolon. Example: "7,7;<=7,10" represents two threshold pair.

-list partitions_summary,sample_of_interest, –list partitions_summary,sample_of_interest

[OPTIONAL] Specify the names of the columns present in the *partitions_summary.tsv* or *SAMPLES_OF_INTEREST_partitions_summary.tsv* file.

-rto, –repeat_threshold_outbreak

[OPTIONAL] This argument can only be used after of a previous analysis of threshold_outbreak and generate a second HTML report

-n_stab N_STABILITY, –n_stability N_STABILITY

[OPTIONAL] Range of threshold in which the cluster composition can be similar.

-thr_stab THR_STABILITY, –thr_stability THR_STABILITY

[OPTIONAL] The neighborhood Adjusted Wallace Coefficient (nAWC) threshold used to determine if a clustering threshold is considered stable.

-ttc TRADITIONAL_TYPING_CATEGORY, –traditional_typing_category TRADITIONAL_TYPING_CATEGORY

[OPTIONAL] Select the most frequent values of the traditional typing category (e.g., serotype, sequence type), i.e. those with the highest number of samples.

A.2 Script for validating final score files

```
import pandas as pd
import argparse
import os
import fnmatch
import sys
import textwrap

def compare_consistent_files(elem1, elem2, input1, input2):

    """
    Compare two score files that already follow the expected naming convention.
    """

    small_path1=f'{input1}/{elem1}'
    small_path2=f'{input2}/{elem2}'
    file1=os.path.abspath(small_path1)
    file2=os.path.abspath(small_path2)

    df1 = pd.read_csv(file1, sep="\t")
    df2 = pd.read_csv(file2, sep="\t")

    df1_processed = df1.drop(index=0)
    df2_processed = df2.iloc[:-1, :-1]
    df1_processed = df1_processed.reset_index(drop=True).round(4)
    df2_processed = df2_processed.reset_index(drop=True).round(4)

    print(f"{file1}\n")
    print(df1_processed)
    print(f"{file2}\n")
    print(df2_processed)

    if df1_processed.columns.equals(df2_processed.columns):

        cols_to_compare = df1_processed.columns
        for col in cols_to_compare:
            iguais = df1_processed[col] == df2_processed[col]
            if iguais.all():
                print(f"Column {col} : True")

            else:
                sys.exit(f"Colums '{col}': False, there are differences in {(~iguais).sum()} lines.")
```

```
def comparison_reversed_files(elem1,elem2,input1,input2):

    """
    Compare two score files when one of them has reversed partition prefixes.
    """

    filename1 = os.path.basename(elem1)
    prefixes = filename1.replace('_final_score.tsv', '').split('_partitions_vs_')
    prefixes = [p.replace('_partitions', '') for p in prefixes]
    reversed_name1 = f"{prefixes[1]}_partitions_vs_{prefixes[0]}_partitions"

    small_path1=f'{input1}/{reversed_name1}_final_score.tsv'
    small_path2=f'{input2}/{elem2}'
    file1=os.path.abspath(small_path1)
    file2=os.path.abspath(small_path2)

    df1 = pd.read_csv(file1, sep="\t")
    df2 = pd.read_csv(file2, sep="\t")

    df1_processed=df1.drop(df1.columns[1], axis=1)
    df2_processed = df2.iloc[:-1, :-1]

    df1_processed = df1_processed.reset_index(drop=True).round(4)
    df2_processed = df2_processed.reset_index(drop=True).round(4)


    i = 0
    j = 1

    max_rows = df1_processed.shape[0]
    max_cols = df2_processed.shape[1]

    print(f'DF1: {file1}')
    print(df1_processed)
    print(f'DF2: {file2}')
    print(df2_processed)


    while i < max_rows and j < max_cols:
        row = df1_processed.iloc[i, 1:]
        column = df2_processed.iloc[:, j]

        row = row.reset_index(drop=True)
        column = column.reset_index(drop=True)

        if len(row) == len(column):
            are_equal = (row.values == column.values).all()
        else:
            sys.exit(f"Error: Different sizes in line {i} and column {j}")
            #are_equal = False

        print(f"Line {i} of df1 processed and colum {j-1} of df2 processed are equal? {are_equal}")
```

```
        i += 1
        j += 1

def collect_input1_files(input1):

    """
    Collect and classify files from input1 folder:
    - valid_files: files with expected prefixes
    - renamed_files: files requiring renaming due to reversed partitions
    """

    valid_files = []
    renamed_files = []
    files = os.listdir(input1)

    for elem in files:

        if elem.startswith('Bionumerics_') or elem.startswith('chewieSnake_GT_partitions_vs_INNUENDO-like-INNUENDO99_HC_partitions') \
              or elem.startswith ('chewieSnake_HC_partitions_vs_INNUENDO-like-INNUENDO99_HC_'):
            valid_files.append(elem)
        else:
            if "_vs_" in elem and elem.endswith("_final_score.tsv"):
                parte1, parte2 = elem.split("_vs_")
                parte2 = parte2.replace("_final_score.tsv", "")
                new_name = f"{parte2}_vs_{parte1}_final_score.tsv"
                renamed_files.append(new_name)

    valid_files1=sorted(valid_files)
    renamed_files1=sorted(renamed_files)

    return valid_files1, renamed_files1


def collect_input2_files(input2):

    valid_files=[]
    files=[]

    for elem in  os.listdir(input2):

        if elem.startswith('Bionumerics_') or elem.startswith('chewieSnake_GT_vs_INNUENDO-like-INNUENDO99_HC_final_score') \
            or elem.startswith('chewieSnake_HC_vs_INNUENDO-like-INNUENDO99_HC_final_score'):
            valid_files.append(elem)

        else:
            files.append(elem)

    valid_files2=sorted(valid_files)
    files2=sorted(files)
```

```
    return valid_files2, files2


def main():

    parser = argparse.ArgumentParser(description=textwrap.dedent("""Script for validation of the EvalTree tool
            It compares final score files from two input folders, ensuring consistency
            and detecting differences between files"""))

    parser.add_argument("-i1", "--input1",
        action = "store",
        required = True,
        help = 'Folder containing all final scores files to be compare run by myself.')

    parser.add_argument("-i2", "--input2",
        action = "store",
        required = True,
        help='Folder with final_scores files from BeOne project.')

    #-----------------------------------------------------------

    args = parser.parse_args()
    input1 = args.input1
    input2 = args.input2

    #-----------------------------------------------------------

    valid_files1, renamed_files1 = collect_input1_files(input1)
    valid_files2, files2 = collect_input2_files(input2)

    #-----------------------------------------------------------
    j=1
    for elem1,elem2 in zip(valid_files1, valid_files2):
        print(f'{j}----------------------------------------------------------------------------------------------------------------')
        print(elem1,'--->', elem2,'\n')
        compare_consistent_files(elem1,elem2,input1,input2)
        j +=1
    #-----------------------------------------------------------
    K=1
    for elem1,elem2 in zip (renamed_files1, files2):

        print(f'{K}----------------------------------------------------------------------------------------------------------------')
        print(elem1,'--->', elem2, '\n')
        comparison_reversed_files(elem1,elem2,input1,input2)
        K +=1

if __name__ == "__main__":
    main()
```

A.3 EvalTree runtime in benchmark tests

Table A.1: Times of the three repetitions for each dataset size in the benchmarking 1.

Dataset size	Rep 1 (s)	Rep 2 (s)	Rep 3 (s)
400	111313.07	111312.12	111262.92
800	122689.86	123284.65	122815.79
1200	135104.82	131135.14	130410.69
1600	141348.56	137956.22	138395.75
2000	146599.07	143059.97	142895.15
2400	151884.81	147797.22	147400.41
2946	153358.53	152618.26	153961.28

Table A.2: Times of the three repetitions for each serotype in the benchmarking 2.

Serotype	Rep 1 (s)	Rep 2 (s)	Rep 3 (s)
Thompson	3547.301	3603.804	3628.652
Typhi	48.392	48.856	48.075

Table A.3: Times of the three repetitions and thresholds for each dataset size in the benchmarking 3.

Dataset size	Rep 1 (s)	Rep 2 (s)	Rep 3 (s)	Threshold GT	Threshold HC
150	114.971	111.945	112.457	449	430
300	647.847	658.738	649.338	1001	973
450	484.137	483.425	479.528	741	719
600	967.010	954.183	966.959	988	968
724	1067.708	1074.916	1111.687	988	968