Saarland University
Center for Bioinformatics
Master's Program in Bioinformatics

Master Thesis

Alignment-free tools for reconstructing phylogenetic relationships in the *Myxococcus/Pyxidicoccus* clade

Submitted by
Guangyi Chen
on April 12, 2022

Supervisor
Prof. Dr. Olga Kalinina

Reviewers
Prof. Dr. Olga Kalinina
Prof. Dr. Tobias Marschall

HIPS HELMHOLTZ
Institut für Pharmazeutische Forschung Saarland

Statement

I hereby confirm that this thesis is my own work and that I have documented all sources used.

Saarbrücken, April 12, 2022

SAARLAND UNIVERSITY
Center for Bioinformatics

Abstract

Alignment-free tools for reconstructing phylogenetic relationships in the *Myxococcus/Pyxidicoccus* clade

Submitted by Guangyi Chen

Computational phylogeny reconstruction utilizes the power of sequence alignment to estimate the distances among a group of target organisms and construct phylogenetic trees to indicate relative relationships to each other. We define sequence alignment as the problem of arranging molecular sequences (DNA, RNA, amino acids) from different target organisms and identifying their similarities. The current major drawback of sequence alignment is that when comparing divergent sequences it is usually hard or expensive to find the optimal alignment. To overcome this, alignment-free methods are becoming more and more popular and some of them show great potential in sequence comparison accuracy and run time. In this thesis, we explored three alignment-free methods Prot-SpaM, Multi-SpaM and SANS-Serif to assess their utility for reconstructing phylogenetic relationships between 71 collected *Myxococcus* and *Pyxidicoccus* strains in order to assess an earlier proposed hypothesis that these species actually form a single genus. Our resulting phylogenetic trees showed that *Pyxidicoccus* strains formed a separate group, distinguishing them from the other *Myxococcus* strains. It may not support the hypothesis that *Pyxidicoccus* and *Myxococcus* belong to the same genus. We also used a reference project data to perform benchmarking tests, whose results showed that SANS-Serif slightly outperformed Prot-SpaM and Multi-SpaM in the accuracy of the phylogeny reconstruction.

Keywords: Sequence alignment, Alignment-free, Computational phylogeny reconstruction, *Myxobacteria*

Acknowledgements

Perhaps it was not very common to decide to complete a second Master's degree, yet I enjoyed this journey.

Foremost, I would like to thank my supervisor Prof. Kalinina for your inspirational and meticulous guidance. I really look forward to our future projects together. Also, I would like to thank Fawaz, my daily supervisor who brought me into the group and project. It's a huge pleasure to get to know you both professionally and privately, I'm proud to call you a friend.

Then I would like to express my gratitude to my colleagues in WIBI, you're all amazing people and I can't wait to continue working with you all.

I would also like to give my applause to my friends who shared this journey with me and you are always providing huge supports: Anthony, Ilya, Maciej, Alper, Joel, Luba, Jeff, Ola, Swarna, Momal, Siwen, Julien, Matias ... It was a pleasure.

Content

Declaration of Authorship **iii**

Abstract **v**

Acknowledgements **vii**

List of Figures **xii**

List of Tables **xvi**

1 Introduction **1**

1.1 *Myxobacteria*: *Myxococcus* and *Pyxidicoccus* 1

1.2 Computational phylogeny reconstruction 2

1.2.1 Methodologies 3

1.2.2 Input data 4

1.3 Alignment-free sequence comparison 6

1.3.1 A brief history 6

1.3.2 Categories of alignment-free sequence comparison 8

1.4 Focus of this thesis 11

1.5 Thesis structure 11

2 Preliminaries **13**

2.1 Sequence alignment 13

2.1.1 Definition 13

2.2 Spaced-word match 14

2.2.1 Definition 14

2.2.2 Biological inference 16

2.3 *de Bruijn Graph* 17

2.3.1 Definition 17

2.3.2 Variants of *de Bruijn Graphs* 18
2.4 Phylogenetic tree 19

3 Motivation **21**

4 Methods **23**
4.1 Pipeline design 23
4.1.1 Workflow 23
4.1.2 Data collection and filtering 25
4.1.2.1 Curated Data 25
4.1.2.2 AFproject Data 26
4.1.3 Data pre-processing 26
4.1.3.1 Curated Data: Protein sequence extraction 26
4.1.3.2 AFproject Data: Protein annotation 26
4.1.3.3 Identification of core genes and corresponding proteins 27
4.1.4 Use of alignment-free tools 29
4.1.4.1 Prot-SpaM 29
4.1.4.2 Multi-SpaM 31
4.1.4.3 Sans-Serif 33
4.1.5 Phylogenetic tree construction and visualisation 35
4.1.6 Tree comparisons 35
4.2 Implementation 37

5 Results **39**
5.1 Data summary 39
5.1.1 Curated Data 39
5.1.2 AFproject Data 40
5.2 Protein clustering 41
5.2.1 Establishing protein sequence identity threshold 41
5.2.2 Determining core and auxiliary genes 41
5.3 Phylogenetic trees 46
5.4 Tree comparisons 48
5.4.1 Curated Data 48
5.4.2 AFproject Data 48
5.4.2.1 Comparison against the reference tree 48
5.4.2.2 Comparison against each other 49

6 Conclusion **51**

Bibliography **53**

Appendix **63**

List of Figures

Figure 1.1 An example U-shape plot from Colquhoun *et al* [1]. It shows the frequency distribution of genes in 10 genomes of 6 bacterial species. The plot has the characteristic U-shaped curve, indicating most genes are either rare or common. 5

Figure 2.1 An example of a *spaced-word match* between DNA sequences S_1 and S_2 with respect to a binary pattern $P = \{1100101\}$ of length 7 from Lau *et al* [2]. This *spaced-word match* represents the two sequences match at the positions of 1s in P (match positions) and mismatches are allowed at the positions of 0 in P (don't-care positions). 15

Figure 2.2 Spaced-word histograms for comparing *Saccharomyces mikatae* with *Saccharomyces cerevisiae* sequences from Leimeister *et al* [3]. There are two peaks separated by the red line. The left peak represents the random spaced-word matches and the right one represents the homologous spaced-word matches. 17

Figure 2.3 An example *de Bruijn Graph* with each node as a 4-mer [4]. Nodes with k-mer $\{AGAC\}$ and $\{GACA\}$ (upper figure) can be collapsed into node with k-mer $\{AGACA\}$ (downside figure). 18

Figure 2.4 An example *colored de Bruijn Graph* using the same topology as Figure 2.3 [4]. In total it contains four color tags, each represents a source genome to which the corresponding node belongs. 19

Figure 2.5 An example rooted phylogenetic tree from Dencker *et al* [5]. Here it concludes 14 plant species, with closer species been annotated with the same colors. 20

Figure 4.1 Workflow of this thesis. It involves two branches. The bottom branch focuses on performing phylogenetic reconstruction on the collected *Myxococcus* and *Pyxidicoccus* strains. The top branch represents a comprehensive benchmarking test using data from a public project [6]. 23
Figure 4.2 Workflow of MMseqs2 clustering function from Steinegger *et al* [7]. It involves a three-step cascade clustering procedure, which repeatedly performs sequence alignment using the `Smith-Waterman` algorithm and clustering from high to low sequence identity in order to accelerate the process. In the last step, all clustering results are merged. 27
Figure 4.3 Workflow of Prot-SpaM [8]. Firstly the amino acid sequences in FASTA format are inputted. Then the program searched for all the *spaced-word match* for each pair of sequences. In the next step. these *spaced-word match* are filtered in order to select the homologous ones. In the last step, the scores of the *filtered spaced-word match* is converted into distance scores. 29
Figure 4.4 Workflow of Multi-SpaM [5]. It involves four steps: data input, searching for *P-blocks*, constructing *quartet tree* and constructing *super tree*. 31
Figure 4.5 Workflow of SAN-Serif [9]. It involves six steps: data input, constructing *colored de Bruijn Graph*, collapsing *colored de Bruijn Graph*, calculating weights of ordered splits, generating unordered splits and phylogenetic tree inference. 33
Figure 4.6 An example of Robinson-Foulds distance calculation created by Yuancheng [10]. Tree 1 and Tree 2 have in total 2 different bipartition cases ($\{AB, CD\}$ for Tree 1 and $\{AC, BD\}$ for Tree 2), hence their Robinson-Foulds distance is 2. 36

Figure 5.1 Length distribution of collected 71 *Pyxidicoccus* and *Myxococcus* strains as **Curated Data**. 40
Figure 5.2 Length distribution of 29 fully assembled *E. coli* genomes from **AFproject Data**. 41

Figure 5.3 Histogram of the number of unique strains from the **Curated Data** protein clusters by MMseqs2 using clustering sequence identity threshold 50%. The histogram shows a clear *U*-shape curve. The far-left part indicates proteins from auxiliary genes and the far-right part indicates proteins from core genes. . . . 42
Figure 5.4 Histogram of the number of unique strains from the **Curated Data** protein clusters by MMseqs2 using clustering sequence identity threshold 60%. The histogram shows a clear *U*-shape curve. The far-left part indicates proteins from auxiliary genes and the far-right part indicates proteins from core genes. . . . 43
Figure 5.5 Histogram of the number of unique strains from the **Curated Data** protein clusters by MMseqs2 using clustering sequence identity threshold 70%. The histogram shows a clear *U*-shape curve. The far-left part indicates proteins from auxiliary genes and the far-right part indicates proteins from core genes. . . . 43
Figure 5.6 Histogram of the number of unique strains from the **AF-project Data** protein clusters by MMseqs2 using clustering sequence identity threshold 50%. The histogram shows a clear *U*-shape curve. The far-left part indicates proteins from auxiliary genes and the far-right part indicates proteins from core genes. . . . 44
Figure 5.7 Histogram of the number of unique strains from the **AF-project Data** protein clusters by MMseqs2 using clustering sequence identity threshold 60%. The histogram shows a clear *U*-shape curve. The far-left part indicates proteins from auxiliary genes and the far-right part indicates proteins from core genes. . . . 44
Figure 5.8 Histogram of the number of unique strains from the **AF-project Data** protein clusters by MMseqs2 using clustering sequence identity threshold 70%. The histogram shows a clear *U*-shape curve. The far-left part indicates proteins from auxiliary genes and the far-right part indicates proteins from core genes. . . . 45
Figure 5.9 Phylogenetic tree of mixed 71 *Pyxidicoccus* and *Myxococcus* strains by Prot-SpaM using the whole proteome. The red block indicates the *Pyxidicoccus* strains. Description of each strain can be found in Appendix A.2. 47

Figure 5.10 Phylogenetic tree of mixed 71 *Pyxidicoccus* and *Myxococcus* strains by SANS-Serif using the whole proteome. The red block indicates the *Pyxidicoccus* strains. Description of each strain can be found in Appendix A.2. 47

Figure A.1 Phylogenetic tree of mixed 71 *Pyxidicoccus* and *Myxococcus* strains by Prot-SpaM using the proteins from core genes under protein clustering sequence identity 0.5. Description of each strain can be found in Appendix A.2. 63

Figure A.2 Phylogenetic tree of mixed 71 *Pyxidicoccus* and *Myxococcus* strains by Prot-SpaM using the proteins from core genes under protein clustering sequence identity 0.6. Description of each strain can be found in Appendix A.2. 64

Figure A.3 Phylogenetic tree of mixed 71 *Pyxidicoccus* and *Myxococcus* strains by Prot-SpaM using the proteins from core genes under protein clustering sequence identity 0.7. Description of each strain can be found in Appendix A.2. 64

Figure A.4 Phylogenetic tree of mixed 71 *Pyxidicoccus* and *Myxococcus* strains by SANS-Serif using the proteins from core genes under protein clustering sequence identity 0.5. Description of each strain can be found in Appendix A.2. 65

Figure A.5 Phylogenetic Tree of mixed 71 *Pyxidicoccus* and *Myxococcus* strains by SANS-Serif using the proteins from core genes under protein clustering sequence identity 0.6. Description of each strain can be found in Appendix A.2. 65

Figure A.6 Phylogenetic tree of mixed 71 *Pyxidicoccus* and *Myxococcus* strains by SANS-Serif using the proteins from core genes under protein clustering sequence identity 0.7. Description of each strain can be found in Appendix A.2. 66

List of Tables

Table 1.1 Taxonomy of *Myxococcus*, currently 8 species have been identified according to NCBI Taxonomy [11]. NCBI = National Center for Biotechnology Information. 2
Table 1.2 Taxonomy of *Pyxidicoccus*, currently 3 species have been identified according to NCBI Taxonomy [11]. NCBI = National Center for Biotechnology Information. 2

Table 5.1 Summary of Curated Data. 39
Table 5.2 Proportions of chosen proteins for **Curated Data** and **AFproject Data** under different sequence identities 50% to 70% for protein clustering by MMseqs2. 45
Table 5.3 normalized Robinson Foulds comparison between Prot-SpaM and SANS-Serif on **Curated Data** using different clustering levels of proteins from core genes. 48
Table 5.4 normalized Robinson Foulds score between Prot-SpaM and reference tree on **AFproject Data** using different clustering levels of proteins from core genes. 49
Table 5.5 Normalized Robinson Foulds score between SANS-Serif and reference tree on **AFproject Data** using different clustering levels of proteins from core genes. 49
Table 5.6 Normalized Robinson Foulds score between Prot-SpaM and SANS-Serif on **AFproject Data** using different clustering levels of proteins from core genes. 50
Table 5.7 Normalized Robinson Foulds score between Prot-SpaM and Multi-SpaM on **AFproject Data** using different clustering levels of proteins from core genes. 50
Table 5.8 Normalized Robinson Foulds score between SANS-Serif and Multi-SpaM on **AFproject Data** using different clustering levels of proteins from core genes. 50

Table A.1 A list of main alignment-free sequence comparison tools based on the summary from Zielezinski *et al* [6]. It lists the tools with their category, current version, released year and source. The tools marked with ⋄ are used in this thesis, NA = Not Available. 67
Table A.2 A summary of the 71 strains within the **Curate Data** of this thesis. For each strain, the description, name used in the phylogenetic tree, Genbank ID, source and link are provided. nuccore = NCBI nucleotide database, assembly = NCBI assembly database, NA = Not Available, HIPS = Helmholtz Institute for Pharmaceutical Research Saarland. 69

Chapter 1

Introduction

1.1 *Myxobacteria*: *Myxococcus* and *Pyxidicoccus*

Myxobacteria (*Myxococcales*) is an order from class *Deltaproteobacteria*. It is also known as fruiting gliding bacteria and consists of a group of predatory bacteria which mainly live in the soil [12, 13]. The members of *Myxobacteria* usually have large genomes (9-14 million base pairs (Mbp) [14]), which helps to produce very diverse secondary metabolites that can be used as drug leads to develop antibiotics, antivirals and cytotoxins [15]. Many compounds isolated from *Myxobacteria* have recently been found to have impressive antiviral activity [16]. To this end, *Myxobacteria* serves as a good source of lead compounds for novel drug discovery.

Myxococcus and *Pyxidicoccus* are the two most studied genera in *Myxobacteria* under family *Myxococcaceae*, their taxonomy hierarchies can be seen in Table 1.1, 1.2. Previous studies have identified 8 *Myxococcus* species and 3 *Pyxidicoccus* species [11]. Interestingly, a phylogenetic analysis based on 16S rRNA genes showed that *Myxococcus* is split into two groups of species, one group contains *Myxococcus stipitatus* and *Myxococcus fulvus* and it bifurcates with *Pyxidicoccus fallax* [17]. Hence there is a hypothesis that *Myxococcus* and *Pyxidicoccus* may be considered as a single genus. In this work, we set out to use alignment-free methods on whole bacterial genomes and/or proteomes to test this hypothesis.

Category	**Taxonomic Name**
Domain	*Bacteria*
Phylum	*Proteobacteria*
Class	*Delta Proteobacteria*
Order	*Myxococcales* (*Myxobacteria*)
Family	*Myxococcaceae*
Genus	*Myxococcus*
Species	*Myxococcus eversor*
	Myxococcus fulvus
	Myxococcus hansupus
	Myxococcus llanfairpwllgwyngyll-gogerychwyrndrobwllllantysilio-gogogochensis
	Myxococcus stipitatus
	Myxococcus vastator
	Myxococcus virescens
	Myxococcus xanthus

Table 1.1: Taxonomy of *Myxococcus*, currently 8 species have been identified according to NCBI Taxonomy [11]. NCBI = National Center for Biotechnology Information.

Category	**Taxonomic Name**
Domain	*Bacteria*
Phylum	*Proteobacteria*
Class	*Delta Proteobacteria*
Order	*Myxococcales* (*Myxobacteria*)
Family	*Myxococcaceae*
Genus	*Pyxidicoccus*
Species	*Pyxidicoccus caerfyrddinensis*
	Pyxidicoccus fallax
	Pyxidicoccus trucidator

Table 1.2: Taxonomy of *Pyxidicoccus*, currently 3 species have been identified according to NCBI Taxonomy [11]. NCBI = National Center for Biotechnology Information.

1.2 Computational phylogeny reconstruction

To identify and classify newly found species or explore the relationship among a group of existing species we need to do phylogeny reconstruction. A phylogeny is the evolutionary history of a group of certain species or entities [18]. Fully determining a phylogeny is usually a non-trial task [19]. For this purpose, the study of phylogeny reconstruction aims to describe the common ancestries of the target species as accurate as possible and to help researchers understand their relative

relationships to each other. The derived relationships are usually represented as a phylogenetic tree (see ***Preliminaries*** 2.4 for more details). For a certain period of time, phylogeny reconstruction relied on investigating the morphological data. More recently, aided by the rapid development of molecular sequencing technology and computational algorithms, computational phylogeny reconstruction is becoming more powerful. Computational phylogeny reconstruction uses algorithms and computational resources to calculate the relationships (usually distances) between multiple target organisms based on molecular sequence data [18]. For the purpose of comparing different sequences, it usually makes extensive use of sequence alignment. Here we define sequence alignment as arranging two or multiple molecular sequences together and using mathematical models to identify the similarities among the them (see ***Preliminaries*** 2.1 for more details).

1.2.1 Methodologies

There are various methods developed for phylogeny reconstruction and distance matrix method is one of the most widely-applied. The central idea behind distance matrix method is to use sequence comparison algorithms, usually pairwise sequence alignment or multiple sequence alignment (MSA) to generate a matrix which represents the pairwise distances between each pair of target organisms. Then, using well-established algorithms which represent different evolutionary models to construct a phylogenetic tree based on this distance matrix. A typical example are hierarchical clustering algorithms such as Unweighted Pair Group Method with Arithmetic mean (UPGMA). The central idea behind UPGMA is to recursively combine the two closest clusters according to the distance matrix table and eventually construct a rooted tree (dendrogram). UPGMA is based on the assumption of constant-rate evolution. In other words, the distances from root to each branch tip are equal [20].

Other methods include Maximal Parsimony, Bayesian Inference and Maximum Likelihood. Maximum Parsimony is a method of constructing and identifying the phylogenetic tree which requires the least number of evolutionary events to explain the given sequence data. The most naive way of identifying such a phylogenetic tree (also called parsimony tree) is simply by enumeration. In other words, this technique considers each possible tree for all the species, calculates a score evaluating the fitness of each tree, and finds the one with the smallest score. However, this

is known to be an NP-hard problem, which leads to the development of accessory heuristic algorithms. [19]. Maximum Likelihood method uses statistical methods of probability distributions and assigns these probabilities to candidate phylogenetic trees. Usually, it requires a substitution model to assess the probability of particular mutations. The tree which requires fewer mutations for the interior nodes to explain the given sequence data will be granted a higher probability [21]. The biggest drawback is the expensive computation process as it is believed to be an NP-hard problem. Another approach is Bayesian Inference that uses a prior probability distribution of all the possible phylogenetic trees based on evolutionary assumptions, such as speciation occurring as stochastic processes. Then the probability is calculated for each candidate phylogenetic tree being correct under the available input data [22].

1.2.2 Input data

The choice of using DNA or amino acids sequences as the source of input to perform computational phylogeny reconstruction is an ongoing topic of discussion. A consensus point of view is that DNA sequences hold the 'raw' information better than amino acids since the information within DNA and within amino acids are separated by transcription and translation events. [18]. For this reason, DNA sequences are often more informative as they may contain mutations which are not recognisable from the corresponding protein sequences. However, many of these mutations are random and non-coding, including them may cause extra difficulty to build the optimal alignment. On the other hand, protein sequence data holds more flexibility in each position (20 amino acid characters over 4 nucleotide characters) ans it is more stable from an evolutionary point of view.

In bacteria, the genes of different strains can be divided into two parts: the core genes which are present in the majority of the other strains of the same species and the auxiliary genes which correspond to the genes which are relatively unique to each strain [23]. All together they form a 'pan-genome' for a bacteria species ('pan' refers to 'whole' in Greek). The core genes are often related to house keeping functions and primary metabolism of the lineage [24]. They are usually well-conserved but still mutations may take place. We are mainly interested in investigating these mutations because they represent the differentiation events across all the strains within this species [1]. The same concepts also apply to the proteins derived from

the core genes and auxiliary genes.

To identify core genes and auxiliary genes from an organism's genome, a traditional approach is to plot the universal gene frequency distribution. An example can be seen in Figure 1.1, in which we can infer that majority of the genes are present in either one or very few strains, or in almost all strains. In other words, most genes are either rare (*auxiliary*, far-left part of the plot) or common (*core*, far-right part of the plot) [1]. This kind of plot is also know as U-shape plot for its asymmetrical U curve pattern.

Usually, we are interested in using the core genes or proteins derived from them to perform phylogeny reconstruction. This is because different species normally contain diverse ranges of auxiliary genes with divergent functionalities. Comparing them is less meaningful and usually generates unreliable results.

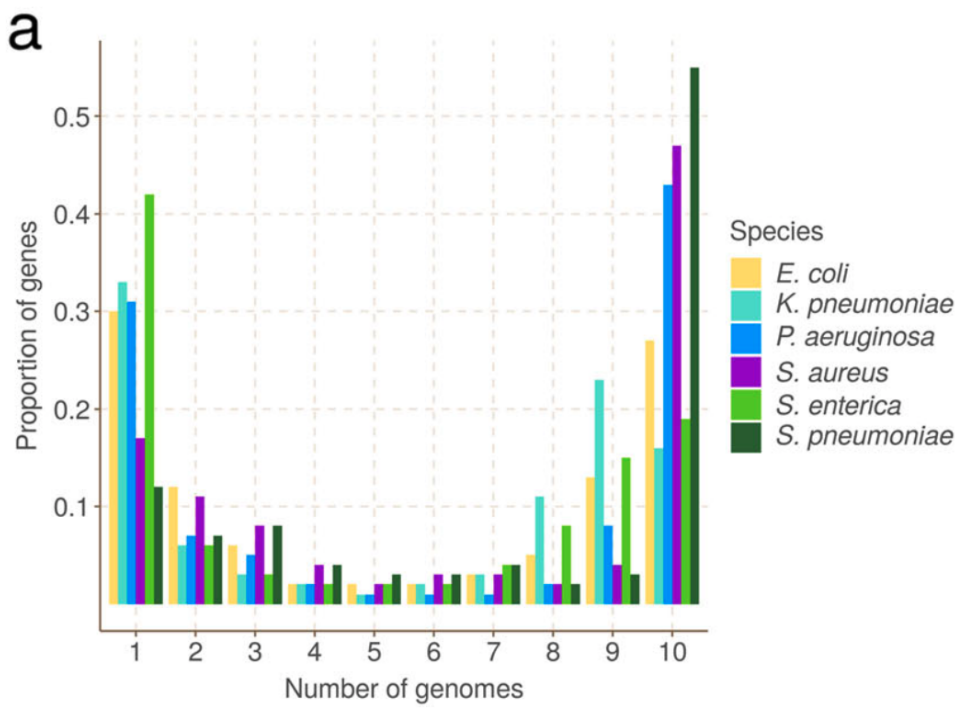

FIGURE 1.1: An example U-shape plot from Colquhoun *et al* [1]. It shows the frequency distribution of genes in 10 genomes of 6 bacterial species. The plot has the characteristic U-shaped curve, indicating most genes are either rare or common.

1.3 Alignment-free sequence comparison

1.3.1 A brief history

Since ***1960s***, thanks to the breakthrough of sequencing technologies, more and more sequence data became available and waiting for bioinformaticians to uncover their mysteries. To properly study sequence data, one of the most fundamental steps is to align the sequences and find the relationships between them. This critical step, sequence comparison, becomes one of the most essential process of dealing with almost all the important biology problems, including phylogeny reconstruction, protein family classification, identification of horizontally transferred genes, detection of recombined sequences and so on [25]. While at that time, sequence comparison was restricted to search the correspondence of individual characters (nucleotides or amino acids) that are in the same order in the candidate sequences – in other words, sequence alignment. Starting from ***1980s***, the scale of sequence data largely stimulated the need and development of novel sequence alignment algorithms. According to Blanchette *et al*, many successful alignment-based tools are still serving as the backbone in this field, such as BLAST [26], FASTA [27], ClustalW [28] and so on. Since ***2010s***, as the development of third and fourth generation of sequencing technology, the data we are able to collect is exponentially larger and easier accessible than only two decades ago. However, other than the abundant data providing massive convenience to solve the biology problem, it also comes with the pitfalls:

1. Current alignment-based methods do not scale with the exponential increasing data today, we need much faster tools to analyse them [6];

2. When comparing distantly related sequences, alignment-based techniques have been shown to be less reliable [29];

3. Within the compared sequences, current alignment-based algorithms assume the linear order of homologous sequence blocks is preserved, they do not apply to the scenarios of the rearrangements in sequences (e.g., recombination and protein domain swapping) or horizontal gene transfer [30, 31];

4. With the development of fourth generation sequencing, it is not practical to align two sequences with hundreds of thousands of nucleotides, let alone millions.

Therefore, as an alternative to sequence alignment, alignment-free sequence comparison approaches have been developed. There is no specific consensus definition of alignment-free sequence comparison so far, but in general it can be regarded as comparing the sequences without specifically producing an alignment [25]. In general, their development can be divided into three phases: 1). Early exploration; 2). Theoretical development; 3). Increased attention.

Early exploration: *1970s*. The earliest study on the alignment-free sequence comparison dates back to mid 1970s. Fox *et at.* classified 10 species of methanogenic bacteria using 16S ribosomal RNA data. In this study, the distributions of 5-mers to 24-mers (consecutive substring of length 5 to 24) were used to calculate association coefficients between species. This coefficient was used to infer the phylogenetic tree [32].

Theoretical development: *1980 - 2000*. Entering 1980s, although alignment-free methods were getting more attention, the overall number of publications in this field was still low. This phase can be regarded that more and more theoretical concepts were invented or borrowed from other fields, e.g., string processing from computer science methodologies [33]. Vinga *et al.* pointed out that in general the alignment-free methods developed during this time can be divided into two groups: methods based on k-mer (a substring of length k) frequency and methods that do not require splitting sequence into k-mers [29]. The same study further concluded that the first group includes procedures based on methods which are based on utilizing such k-mer count information, such as the Euclidean distance and relative entropy of frequency distributions. The second category, in contrast, corresponds to the techniques that are independent from the resolution of the sequence, i.e. they do not involve counting substrings of fixed length. Instead, they used complexity theory or other representation of sequences. These two categories descend from distinct theoretical lineages, and the former had much greater presence in the literature. [29].

Increased attention: *2000 - now*. During this period of time, the amount of attention garnered by alignment-free sequence comparison increased tremendously. More and more tools have been developed and the categories of alignment-free are being refined. In general, according to the previous studies [6, 25, 29], the mainly categories used today are: 1). k-mer counts; 2). Length of common substrings; 3).

Micro-alignments; 4). Graph-based; 5). Sequence representations based on chaos theory; 6). Others. The first three categories be regarded as descendants of k-mer frequency methods popularized in the ***1980s***. Some of these tools have shown good prospects and compatibility when compared to the traditional alignment-based tools, including the three used in this thesis: Prot-SpaM, Multi-SpaM and SANS-Serif.

1.3.2 Categories of alignment-free sequence comparison

According to the previous studies [6, 25, 29], the categories of alignment-free methods can be generally divided as follows:

1. k-mer counts. By far this is the most widely used group of methods. In general, it summarises the k-mer statistics for each input sequence and uses these statistics as the source for comparison. The conversion from sequence to k-mers and further manipulations on k-mers ensure good a scalability for these tools. However, these methods don't touch upon the deeper mechanism and characteristics of the target sequence. For example, when comparing homologous sequences there are certain highly similar and conserved sequence blocks, yet k-mer counts would possibly neglect to use this valuable information for comparison;

2. Length of common substrings. This category mainly searches for the longest common substrings between the input sequences. Sometimes some mismatches are tolerated when searching for such common substring. Then, the length of the common substrings are used to infer the distances;

3. Micro-alignments. This category of methods searches for a matched region of fixed length where some mismatches at certain positions are allowed. This region usually represents homologous sequence block. Then the quality of this matched region is evaluated and later used to infer the distance (see later in this section for more details);

4. Number of word matches. This method is similar to k-mer counts method which also relies on inspecting the input sequences through their representing sets of k-mers. However, this method compares these different sets of k-mers directly and converts it into a distance;

5. Others. Other methods include methods from information theory, e.g., estimating genome entropy or motif and region classification.

A comprehensive list of main alignment-free sequence comparison tools based on the summary from Zielezinski *et al* can be found in Appendix A.1 [6]. In this thesis, we want to focus on two recent alignment-free sequence comparison methods as they are developed with well-designed mechanisms and have shown good prospects: micro-alignment method and *de Bruijn Graph* based method.

Micro-alignment sequence comparison refers to the group methods which requires the compared sequences aligning at a small region, where at certain positions there should be exact matches while at other positions mismatches are allowed. Strictly speaking, this kind of method is not 'alignment-free' as there are still small numbers of alignments performed within the small region. However, in contrast to the traditional concept of sequence alignment which requires the full (or close to full) comparisons among two or more sequences, this group of methods only focus on this small region and omits the other majority of the comparisons. Hence we still regard it as alignment-free sequence comparison method. One representative subgroup of methods within micro-alignments is called *filtered spaced-word matches* [8]. The basic idea is to use gap-free pairwise alignments of fixed-length words with matching nucleotides or amino acid residues at certain pre-defined positions. Such *spaced-word matches* can be rapidly identified. Also, after discarding random background matches, the remaining 'homologous' *spaced-word matches* can be used to estimate the phylogenetic distance between two taxa. The details are discussed in the ***Preliminaries*** 2.2. In this thesis, we explored two micro-alignment tools Prot-SpaM and Multi-SpaM, the details will be introduced in the *Methods* section [8, 34].

Another category of the alignment-free sequence comparison method is the *de Bruijn Graph* based method. This name of category was not formally proposed by previous studies [6, 25, 29], yet for its uniqueness and novelty which elegantly utilizes the characteristics of *de Bruijn Graph* (*dBG*) (see ***Preliminaries*** 2.3 for more details), in this thesis we name it '*de Bruijn Graph* based method'. The *de Bruijn Graph* based method studies the phylogeny problem from a totally new point of view. The current sequencing procedure is becoming cheaper and fast, where large quantity of new data is generated in a unprecedented pace. However,

completely finishing an assembly still remains a laborious work, where many sequencing data still stay in an unfinished state (e.g., sets of multiple contigs[1]). This causes a big challenge for the traditional phylogeny methods as they normally rely on comparing the marker sequences. However, finding these marker sequences requires multiple sequence alignment, which is regarded as a computationally expensive task. The characteristics of *colored de Bruijn Graph* or *de Bruijn Graph* actually make it very suitable for such scenario: the nodes of *colored de Bruijn Graph* represent a k-mer, which can be generated by assembly data at any level (reads, contigs, scaffolds). On the other hand, each node is associated with one or more color tags, which represent the genomes it come from. This characteristic serves as a perfect hint to infer the bipartition of input organisms. Constructing a *colored de Bruijn Graph* usually takes less time than multiple sequence alignment [34]. To this end, *de Bruijn Graph* method was developed. So far there is only one representative tool: SANS-Serif [34]. The k-mers of all genomic sequences (assemblies or reads) are stored in a *colored de Bruijn graph*, which is then traversed to extract phylogenetic signals [9]. The details will be introduced in the ***Methods*** section.

[1]a series of overlapping DNA sequences

1.4 Focus of this thesis

There are three main focuses of this thesis:

1. Properly comparing three alignment-free sequence comparison tools[2] by utilizing the reference data and tree from a public alignment-free project;
2. Using three selected alignment-free sequence comparison tools in parallel to perform phylogeny reconstruction based on a collected set of 71 mixed *Myxococcus* and *Pyxidicoccus* strains;
3. Examining the impact of using representative protein sequences from core genes on the constructed phylogenetic trees.

1.5 Thesis structure

The structure of this thesis is described as such:

1. The following chapter (***Preliminaries***) gives a short introduction of the basics in the scope of this thesis, which serves as the foundation of the methodologies;
2. The next chapter (***Motivation***) describes in details on why we want to use the alignment-free methods to perform Phylogeny Reconstruction and what is the idea to prepare different data sets to properly compare the results from different alignment-free methods;
3. After that a chapter (***Methods***) will introduce the pipeline designed for this research;
4. Then there is the chapter (***Results***) giving the results;
5. Finally, a chapter (***Conclusions***) will give the discussion and conclusions.

[2]Prot-SpaM and Multi-SpaM are based on micro-alignment and SANS-Serif is based on *de Bruijn Graph*

Chapter 2

Preliminaries

This chapter provides the basic knowledge that serves as the mathematical background for this thesis:

1. Basics of sequence alignment;
2. Idea of *spaced-word match*;
3. Basics of *de Bruijn Graph*;
4. Basics of phylogenetic tree;

2.1 Sequence alignment

2.1.1 Definition

We define a *string* as a sequence of characters of a non-empty set, *alphabet* Σ. For *string* of DNA, $\Sigma = \{A, T, C, G\}$. A gap sign is denoted as $\{-\}$ and $\{-\} \cap \Sigma = \emptyset$.

For *string* s, $t \in \Sigma \bigcup \{-\}$, we define their *alignment* A as a *string* over $(\Sigma \bigcup \{-\})^2 \setminus \{(-,-)\}$ with $\pi_1(A) = s$ and $\pi_2(A) = t$, where:

1. $\pi_1(A_i) = \pi_1((a,b)_i) := a$ for $a \in \Sigma$ and
2. $\pi_1((-,b)_i) := \epsilon$

and π_2 is defined analogously [4].

For an alignment A, the *column* of A is denoted as C where: 1) a column of form $\binom{a}{a}$ for $a \in \Sigma$ is called a *match* column; 2) a column of form $\binom{a}{b}$ for $a \in \Sigma$, $b \in \Sigma$ and $a \neq b$ is called a *mismatch* column; 3) if one of the characters in C is a gap sign, C is an indel (insertion or deletion) column.

A *global alignment* is an alignment that both sequences must be aligned from start to end. A *local alignment* is an alignment of fragments of both sequences can start and end anywhere. It is useful for comparing two *strings* a and b, that are not globally similar to each other but only show similarity in some positions. In such cases, it makes more sense to search for the most similar substrings with a and b. A hybrid form of alignment from *global alignment* and *local alignment* is called *semi-global alignment* or *glocal alignment*. Given two *strings* a and b, a is short and we are interested in the best match of a within b. Here, the alignment form that the whole of a is required to be aligned to any part of b is called a *semi-global alignment*.

So far, we are discussing on the scenario of comparing two *strings*, it is called *pairwise alignment*. As an extension of this, *Multiple Sequence Alignment (MSA)* deals with the problem of aligning more than two *strings* at the same time. MSA is usually represented in a matrix with multiple rows and each row corresponds to a *string*. Formally, for *strings* $S_1, ..., S_k$ over *alphabet* Σ, the *MSA* A is defined as a *string* over $(\Sigma \bigcup \{-\})^k \setminus \{(-)^k\}$. In other words, elements of A are **columns** of k symbols of $\sigma \bigcup \{-\}$ and at least one of them is from Σ.

2.2 Spaced-word match

2.2.1 Definition

For an *alphabet* Σ, a sequence S of length L and $0 \leq i \leq L$, $s[i]$ denotes the i-th character of S. We define a *binary pattern* P a word over *alphabet* $\{0, 1\}$. A position k in P is called a *match-position* if $P[k] = 1$ and a *don't-care-position* if $P[k] = 0$. The total number of *match-positions* in P is called its *weight*. We denote $*$ as a wildcard character that $* \notin \Sigma$. A *spaced word* of S with respect to

P is a word s over *alphabet* $\Sigma \bigcup *$ of the same length of P that $s[i] = S[i]$ for i is a *match-position* and $s[i] = *$ if i is a *don't-care-position*. We say that a *spaced word* s occurs in a sequence S at position i if $s[k] = S[i + k - 1]$ for all *match-positions* k of *binary pattern* P.

For sequence S_1 and S_2 of length L_1 and L_2 respectively, a *binary pattern* P of length L_p and positions i, j where $1 \leq i \leq L_1 - L_p + 1$ and $1 \leq L_2 - L_p + 1$, we say that there is a *spaced-word match* between S_1 and S_2 at (i, j) with respect to P if the same *spaced word* occurs at position i for S_1 and position j for S_2. In other words, $S_1[i + k - 1] = s_2[j + k - 1]$.

For a *spaced-word match*, we define its score as the sum of the substitution scores of the nucleotide pairs aligned at the *don't-care positions* using a substitution matrix (DNA sequence: *HOXD70* [35], amino acid sequence: *BLOSUM62* [36]). Given a predefined threshold t, if the score of a *spaced-word match* is over t, we call it a *filtered spaced-word match (FSWM)* or *homologous spaced-word match*. If the score is under t, we call it a *background spaced-word match* which represents a random match.

S_1 :		T	T	A	T	G	A	C	C	A	C	T	C
S_2 :	A	C	T	A	C	G	A	T	C	G	A		
P :			1	1	0	0	1	0	1				

FIGURE 2.1: An example of a *spaced-word match* between DNA sequences S_1 and S_2 with respect to a binary pattern $P = \{1100101\}$ of length 7 from Lau *et al* [2]. This *spaced-word match* represents the two sequences match at the positions of 1s in P (match positions) and mismatches are allowed at the positions of 0 in P (don't-care positions).

An example of a *spaced-word match* between sequence S_1 and S_2 with respect to *binary pattern* $P = \{1100101\}$ can be found in Figure 2.1 [2]. The *weight* of this *spaced-word match* is 4. Here we are aligning two sequences S_1 and S_2. A predefined *binary pattern* P of length l is used to indicate *match positions* (1) and *don't-care positions* (0). This pattern P usually represents the source of homologous while some certain numbers of mismatches are allowed. We use this pattern P to find a 'micro-alignment' region between S_1 and S_2 such that at all match

positions (1) there are always exactly matched nucleotides/amino acids while mismatches can be tolerated at the *don't-care positions* (0).

2.2.2 Biological inference

FSWM is particularly useful when comparing homologous sequences as each *FSWM* can be seen as a representation of a homologous block. On the other hand, matching of *FSWM*s between two sequences is flexible and not restricted to sequential searching when compared to the traditional alignment-based algorithms. Further, applying a predefined threshold is very useful to filter out the *background spaced-word matches* from the *FSWM*s of true homologous. We can plot the number of *spaced-word matches* against the corresponding scores for two sequences and a binary pattern P, i.e. for each possible score value, one plots the number of *spaced-word matches* with this score. We call such a plot a *spaced-words histogram*, an example is shown as Figure 2.2 [3]. We can see that there are two peaks, the left one with the lower scores represents the *background spaced-word matches* and the right peak with higher scores represent the *homologous spaced-word matches*. These two peaks can be clearly separated by a threshold, e.g., 0. Studies have shown that the background *spaced-word matches* are approximately normally distributed, while the distribution of homologous *spaced-word matches* is more complex [3]. Under the assumption of no insertions and deletions, a simulation shows that the distribution of homologous *spaced-word matches* is also approximately normal [8].

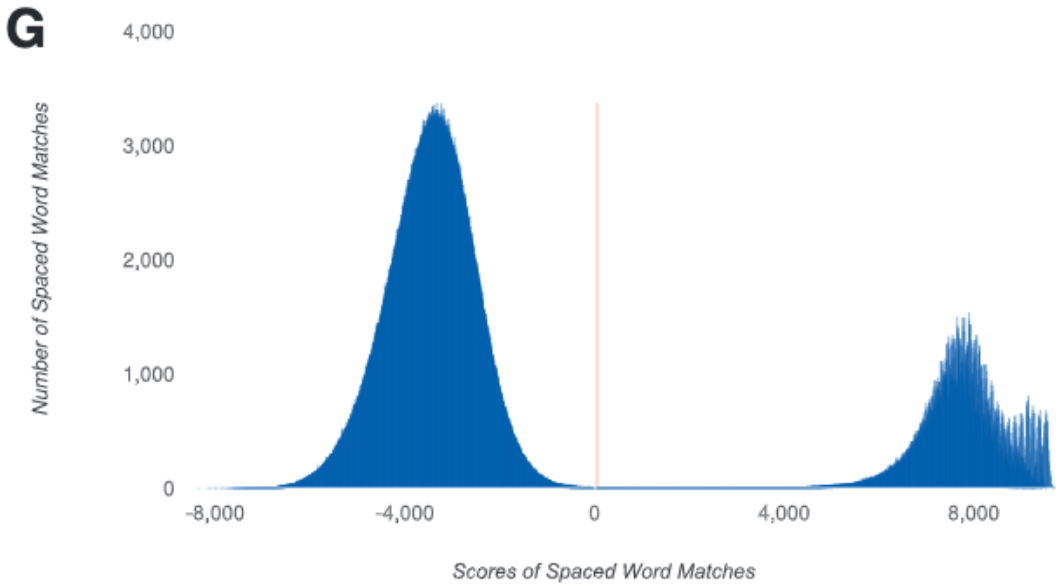

FIGURE 2.2: Spaced-word histograms for comparing *Saccharomyces mikatae* with *Saccharomyces cerevisiae* sequences from Leimeister *et al* [3]. There are two peaks separated by the red line. The left peak represents the random spaced-word matches and the right one represents the homologous spaced-word matches.

2.3 *de Bruijn Graph*

2.3.1 Definition

Same as in the previous sequence alignment section, we define a *string* s as a sequence of characters of a non-empty set, *alphabet*. The i-th character of s is denoted as *s[i]* and the length of s is denoted as $|s|$. A contiguous *substring* of length k is called k-mer. We define a *genome* as a set of *strings* over *alphabet* $\{A, T, C, G\}$.

We define a *de Bruijn Graph* of *genome* G as a directed graph $G = \{V, E\}$ with node set V and edge set E where each node $V_i \in V$ represents a unique k-mer and each edge $E_{ij} \in E$ connects two nodes V_i and V_j such that they overlap by *k-1* characters. In other words, for two nodes $V_i, V_j \in V$, there exists an edge E_{ij} from V_i to V_j *iff* $V_i[2...k] == V_j[1...k-1]$. For each node which has exactly one incoming and outgoing edge, it can be *collapsed* with the adjacent node. A maximal non-branching path is a *unitig*.

An example of *de Bruijn Graph* can be described as Figure 2.3. In this example, each node of the *de Bruijn Graph* is a k-mer of size 4. In this example, the lower subfigure illustrate the collapsed *de Bruijn Graph* from the upper subfigure.

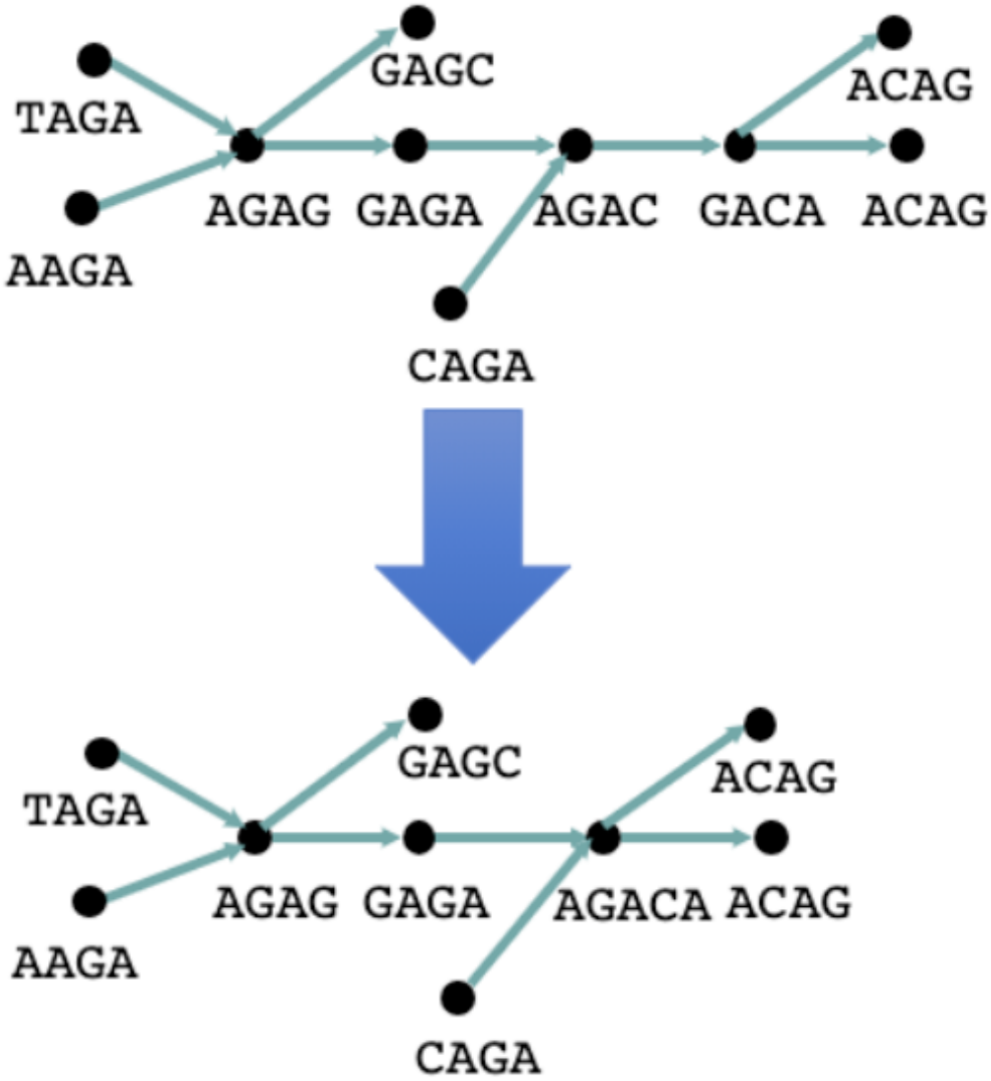

FIGURE 2.3: An example *de Bruijn Graph* with each node as a 4-mer [4]. Nodes with k-mer $\{AGAC\}$ and $\{GACA\}$ (upper figure) can be collapsed into node with k-mer $\{AGACA\}$ (downside figure).

2.3.2 Variants of *de Bruijn Graphs*

There are also variants of *de Bruijn Graph*, including *colored de Bruijn Graph*. *colored de Bruijn Graph* is a node-labeled *de Bruijn Graph* (V, E, col) where each

node represents a k-mer associated with an addidtional set of colors $col(v)$ representing the set of genomes the k-mer occurs in. An example of *colored de Bruijn Graph* can be seen in Figure 2.4.

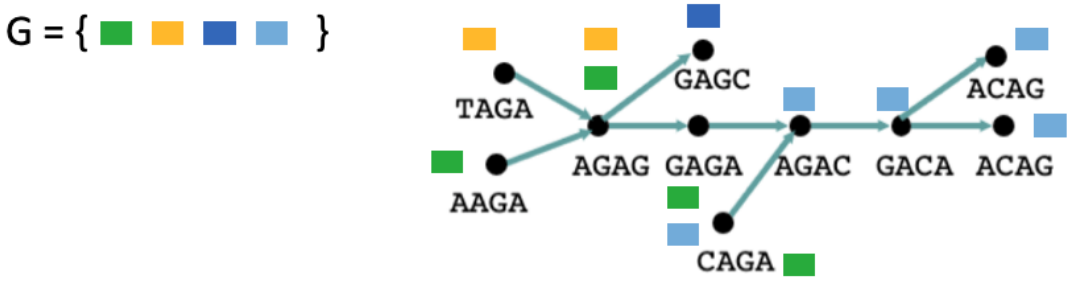

FIGURE 2.4: An example *colored de Bruijn Graph* using the same topology as Figure 2.3 [4]. In total it contains four color tags, each represents a source genome to which the corresponding node belongs.

2.4 Phylogenetic tree

A phylogenetic tree (also known as phylogeny or evolutionary tree) is a tree or branching diagram displaying the evolutionary relationships among various biological species or other entities. It is usually constructed upon similarities in their morphological or genetic characteristics. [19]. The structure of phylogenetic tree can be described as such:

1. Leaves correspond to each species;
2. Each node with descendants represents the inferred most recent common ancestor of those descendants [18]. Internal nodes represent the most common recent ancestor of the covered leaves. Internal nodes are also regarded as hypothetical taxonomic units because they usually cannot be directly observed [37];
3. The length of the edge represents the evolutionary distance calculated by specific algorithms;
4. Phylogenetic tree can be rooted or unrooted. In a rooted phylogenetic tree, there exists a root node inferring the most recent common ancestor of all the covered leaves.

An example phylogenetic tree is shown as Figure 2.5, this is a rooted phylogenetic tree constructed for 14 plant species using MULTI-SPAM [5].

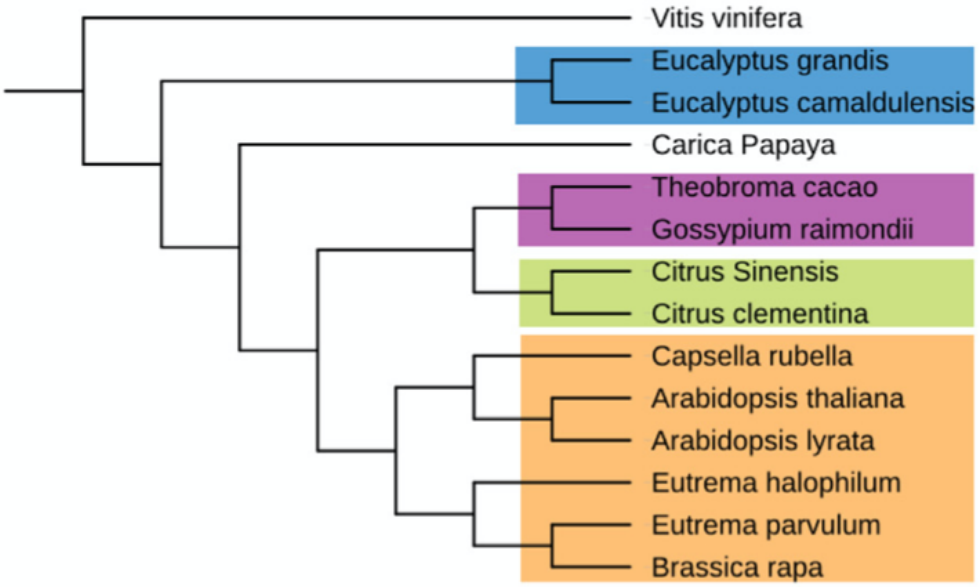

FIGURE 2.5: An example rooted phylogenetic tree from Dencker *et al* [5]. Here it concludes 14 plant species, with closer species been annotated with the same colors.

Chapter 3

Motivation

Built on the previous two chapters on the introductions and preliminaries, here we would like to introduce the main motivation of this thesis. First of all, so far there is no study that focuses on performing a comprehensive phylogenetic study on *Myxococcus* and *Pyxidicoccus* altogether. Given that they are the target organisms in drug discovery, we are interested in gathering as much strain data of good quality as possible and performing a comprehensive phylogeny reconstruction. Secondly, some studies indicate that *Pyxidicoccus* and *Myxococcus* are not far from each other from the evolutionary perspective and may actually represent a single genus, and thus should be taxonomically reclassified [17]. We are interested in investigating the positions of *Pyxidicoccus* and *Myxococcus* strains relative to each other in the phylogenetic trees to see if there is any further support for this hypothesis. Last but not least, we are interested in investigating and applying the novel alignment-free sequence comparison methods. We explored three methods that have different mechanisms and targets where we could like to access their utility for reconstructing phylogeny relationships for the collected strains. We would also like to investigate the impact of using selected proteins from core genes as the input data for phylogeny reconstruction and compare it with the results using the full proteome.

Chapter 4

Methods

4.1 Pipeline design

4.1.1 Workflow

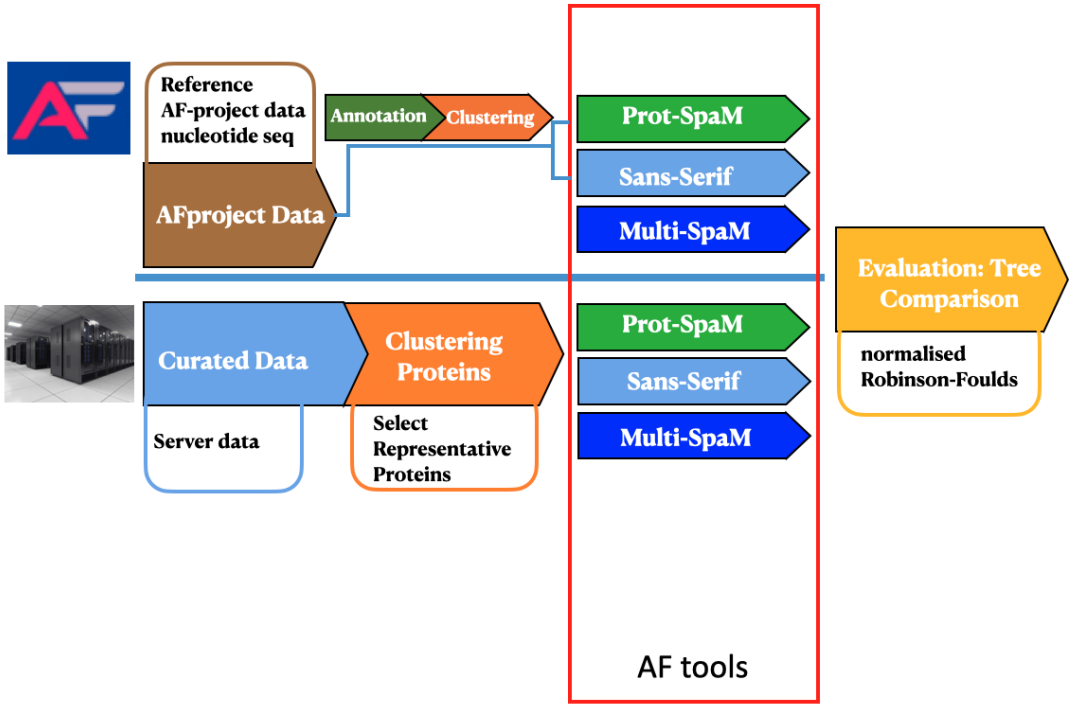

Figure 4.1: Workflow of this thesis. It involves two branches. The bottom branch focuses on performing phylogenetic reconstruction on the collected *Myxococcus* and *Pyxidicoccus* strains. The top branch represents a comprehensive benchmarking test using data from a public project [6].

The overall workflow of this thesis can be described as Figure 4.1. We designed two separating branches, which represent the analysis of **Curated Data** (bottom

branch) and **AFproject Data** (top branch). The main objective underpinning this design is that the major focus is put on the analysis of the **Curated Data** of 71 *Pyxidicoccus* and *Myxococcus* strains and construct a phylogenetic tree using different alignment-free sequence comparison tools (bottom branch). In order to properly compare the power of these tools, we also need a reference tree. Hence we introduced a second branch which utilizes the data from a public project **AFproject** [6]. **AFproject** is a free service designed for objectively evaluating the performance of alignment-free sequence comparison tools on different datasets. This web service aims at simplifying and standardizing the process of benchmarking alignment-free tools.

For each of the two branches, the data is collected and filtered. These data are fed into each chosen alignment-free method: Prot-SpaM, Multi-SpaM and Sans-Serif. For reason that Prot-SpaM uses only amino acid sequence data as input, the protein sequences of **Curated Data** are subsequently extracted. Due to the lack of available protein sequences of **AFproject Data**, Prokka is used to annotate the nucleotide sequences [38]. We are also interested in choosing the proteins of core genes as the input data for the phylogeny reconstruction, hence we performed protein clustering using MMseqs2 and the proteins representing core genes are then used as the input data for Prot-SpaM [7].

Then we construct the phylogeny trees based on the output of each alignment-free sequence comparison method. The results were compared to each other using normalized Robinson-Foulds metric [39].

The rest of this chapter will introduce the details of this workflow.

4.1.2 Data collection and filtering

4.1.2.1 Curated Data

The idea of **Curated Data** is that we want to collect as many high-quality *Pyxidicoccus* and *Myxococcus* strains as possible. After collecting all available sequences (see below) and analyzing the genome size distribution, here we defined high-quality strains as those with genome sizes over 8 million bp.

In this thesis, **Curated Data** has two sources: the in-house strain data from the Helmholtz Institute for Pharmaceutical Research Saarland (HIPS) and the public data collected from National Center for Biotechnology Information (NCBI). The HIPS in-house data consist of GenBank-formatted files of 14 strains in total (11 *Myxococcus* strains and 3 *Pyxidicoccus* strains). For querying and downloading the public data more efficiently, the E-UTILITIES (Entrez Programming Utilities) tool is used to search and collect complete or close-complete sequence records (over 8 million bp) in the assembly and nuccore database from NCBI [40]. The E-UTILITIES are a set of programs on the server that provide a stable interface to the Entrez query and database system at the NCBI. It includes a terminal program EDIRECT (Entrez Direct), which provides access to the NCBI's suite of interconnected databases (publication, sequence, structure, gene, etc.) from a Unix terminal window [41]. In this thesis, we used EDIRECT to download all the available sequence data in Genbank format.

The example query command for searching high-quality strains is `esearch -db nuccore -query "txid32[ORGN] AND ("500000"[SLEN] : "40000000"[SLEN])" | efetch -format uid`, where `txid32` represents the taxonomy id of *Myxococcus* from NCBI (alternatively, `txid224458` for *Pyxidicoccus*). This command will generate a list of ids where each one represents a high-quality strain. Then we used the following example command to download these strains in GenBank format based on these ids: `efetch -db nuccore -id uid -format gb > each-strain.gb`.

4.1.2.2 AFproject Data

We directly downloaded the reference data from **AFproject**, a free web service for evaluating alignment-free sequence comparison tools. This dataset consists of 29 fully assembled *Escherichia coli* (*E. coli*) genomes in FASTA format with an average sequence length of 4,895,247 bp. This dataset was originally curated by Yin and Jin [42] and has been widely used in the past by other researchers to evaluate their alignment-free tools [5, 43, 44].

4.1.3 Data pre-processing

4.1.3.1 Curated Data: Protein sequence extraction

For **Curated Data**, we extracted proteins from the GenBank file of each collected strain from the previous step. To finish this job, we used the module `Bio.SeqIO` from `Biopython`. `Biopython` is a collection of tools written in `Python` for biological computation problems [45] [46] [47]. `Bio.SeqIO` has clean interface and well-defined functions to efficiently parse GenBank files.

4.1.3.2 AFproject Data: Protein annotation

Due to the lack of amino acid data, protein annotation is performed on the **AFproject** data as it only contains the fully-assembled genomes. Here we used the tool Prokka to do the protein annotation. Prokka is a standard command-line tool to fully annotate a draft bacterial genome [38].

Prokka does the annotation in two steps. First, it uses a list of dependency tools to predict and locate the coordinates of genomic features of the input sequences [48–52]. Then, comparing the coding sequence region with multiple datasets holding known sequences. In order to finish this task faster, Prokka starts searching from a small range of trustworthy databses, then moves to domain-specific ones. In the end switching to the curated database of protein families [38].

We ran Prokka using the **AFproject Data** of fully assembled 29 *E. coli* genomes directly as input. Among the output files, we only need the FASTA file (`.fna`) of

translated coding genes as the main output to subsequently feed the alignment-free tools PROT-SPAM and SANS-SERIF.

4.1.3.3 Identification of core genes and corresponding proteins

In order to select the proteins that correspond to the core genes out of the proteome, we firstly performed protein clustering using MMSEQS2, a widely applied tool for this purpose [7].

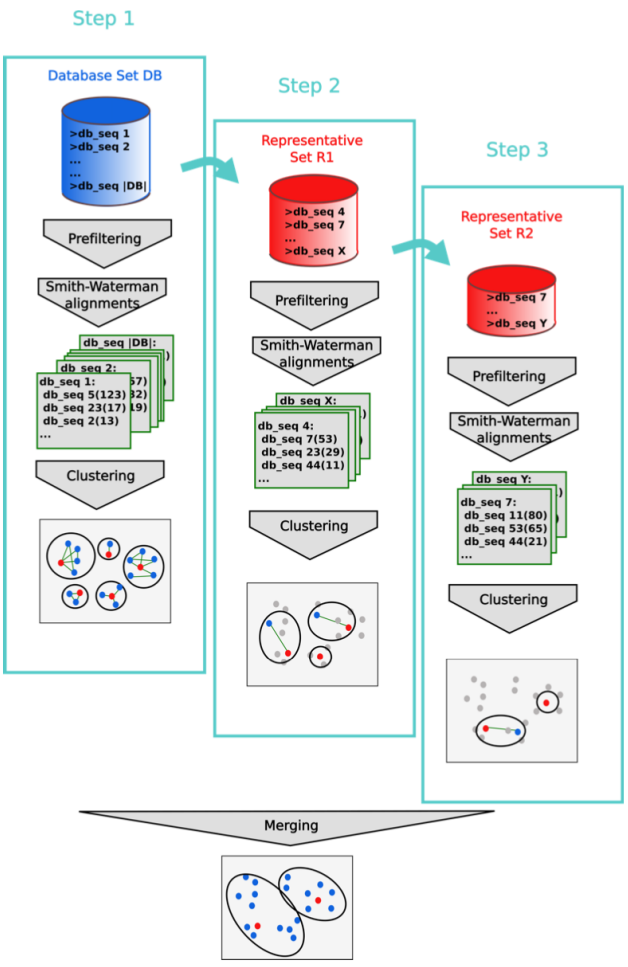

FIGURE 4.2: Workflow of MMSEQS2 clustering function from Steinegger *et al* [7]. It involves a three-step cascade clustering procedure, which repeatedly performs sequence alignment using the `Smith-Waterman` algorithm and clustering from high to low sequence identity in order to accelerate the process. In the last step, all clustering results are merged.

The workflow of MMseqs2 can be seen in Figure 4.2. MMseqs2 uses a cascade clustering procedure (steps 1 - 3). Each of these steps entails sequence alignment via `Smith-Waterman` algorithm followed by clustering from high sequence identity to low. Clustering terminates at a user-specified sequence identity. The resulted representative sequences are fed into the next step. In the last step, the clustering results are merged.

Firstly, we collected all the proteins for each strain from the previous step and put them all together in a single FASTA file. Next, we performed the protein clustering with MMseqs2 using the `easy-cluster` flag option. Usually, we don't know how similar the core proteins are related to each other before the analysis. Hence here we extracted some typical ubiquitous proteins *rpoB*, *rpoC* and *rpoD* from 10 randomly selected strains and calculate their sequence identities. These sequence identities serve as the upper bound of the overall sequence identity. Then we gradually lowered the sequence identity threshold and performed the protein clustering based on it until we could get a clear U-shape plot (see below). The clustering results are typically three files: a FASTA file containing all proteins, a TSV file containing the clustering results, and a FASTA file containing the representative proteins from each cluster.

Finally, to identify the proteins from core genes, we used the TSV file to generate a U-shape plot, it is a histogram plot where the x-axis represents the number of unique strains and the y-axis represents the counts of protein clusters containing certain number of unique strains. The plot is usually in a U shape and the proteins from core genes are located on the right side of the plot (see ***Introduction*** 1.2.2 for more details). We chose a threshold according to the shape and extracted these proteins as the input for the subsequent alignment-free tools.

4.1.4 Use of alignment-free tools

After the collected data (**Curated** and **AFproject**) were filtered and pre-processed, they were served as the input data to the three selected alignment-free tools: Prot-SpaM, Multi-SpaM, and Sans-Serif.

4.1.4.1 Prot-SpaM

Workflow

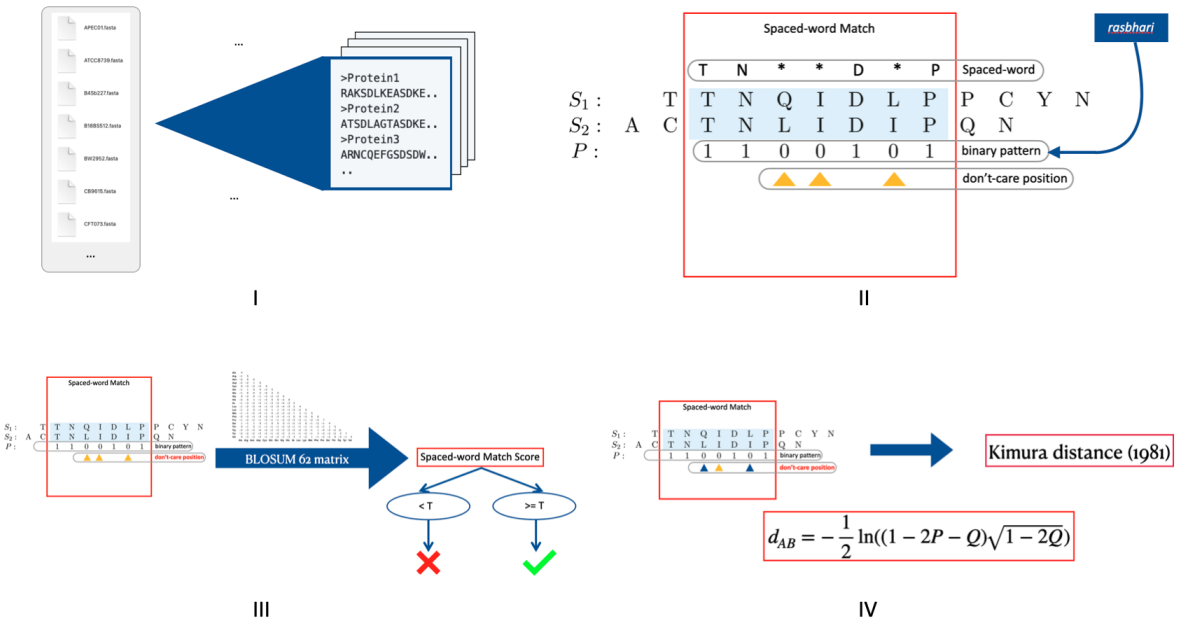

Figure 4.3: Workflow of Prot-SpaM [8]. Firstly the amino acid sequences in FASTA format are inputted. Then the program searched for all the *spaced-word match* for each pair of sequences. In the next step. these *spaced-word match* are filtered in order to select the homologous ones. In the last step, the scores of the *filtered spaced-word match* is converted into distance scores.

Figure 4.3 summarises the workflow of Prot-SpaM.

1. Input: First, the program takes the proteome of each strain in the FASTA format as input. Ideally, the whole proteome is suggested;

2. Searching for *spaced-word matches*: Then a set of binary patterns (by default: 4) are generated using a dependency tool RASBHARI [53]. RASBHARI uses a hill-climbing algorithm to optimize pattern sets according to a user-defined criterion. In this case, the binary patterns of the same set have the minimal overlap-complexity. In other words, these binary patterns are maximally different from each other. These binary patterns are used to calculate the *spaced-word matches* for each pair of proteins from different strains. When there are multiple *spaced-word matches* occurred with respect to the same *spaced-word* W, PROT-SPAM calculates the scores of all *spaced-word matches* which include W and ranks them in descending order. Then it selects them one by one and ensures that each W occurs at least one of the selected *spaced-word matches*;

3. Filtering: However, some of these *spaced-Word matches* are random matches and do not represent true homologies. In the next step, the program calculates the quality score for each *spaced-word match* by adding up the substitution scores of the aligned amino acids at the *don't-care* positions using `BLOSUM62` substitution matrix. If it passes the threshold, then it can be regarded as a *filtered spaced-word match*. By default, threshold 0 is used;

4. Estimating distance: Finally, in order to estimate pairwise distances between two input sequences, each pair of amino acids aligned to each other at the don't-care positions of the selected *spaced-word matches* is used to insert into the Kimura model that approximates the PAM distance between sequences based on the number of mismatches per position [54]. The average distance score is outputted as the distance between the two target organism.

Installing and Running

PROT-SPAM is implemented in `C++`, the source code is available at `https://github.com/jschellh/ProtSpaM`. We downloaded and compiled the program locally. PROT-SPAM only takes amino acid sequences as input, hence for **Curated Data**, we used the selected core proteins as input from the mixed 71 *Pyxidicoccus* and *Myxococcus* and for **AFproject Data** we used the annotated proteins of the 29 *E. coli* strains. Note that PROT-SPAM suggests using the whole proteome as input. However, as part of our interest in this thesis, we are also interested in the result of the phylogenetic tree built only on the proteins from core genes data. To this end, we also ran PROT-SPAM based on all proteins for the **Curated Data**

and made a comparison.

4.1.4.2 Multi-SpaM

Workflow

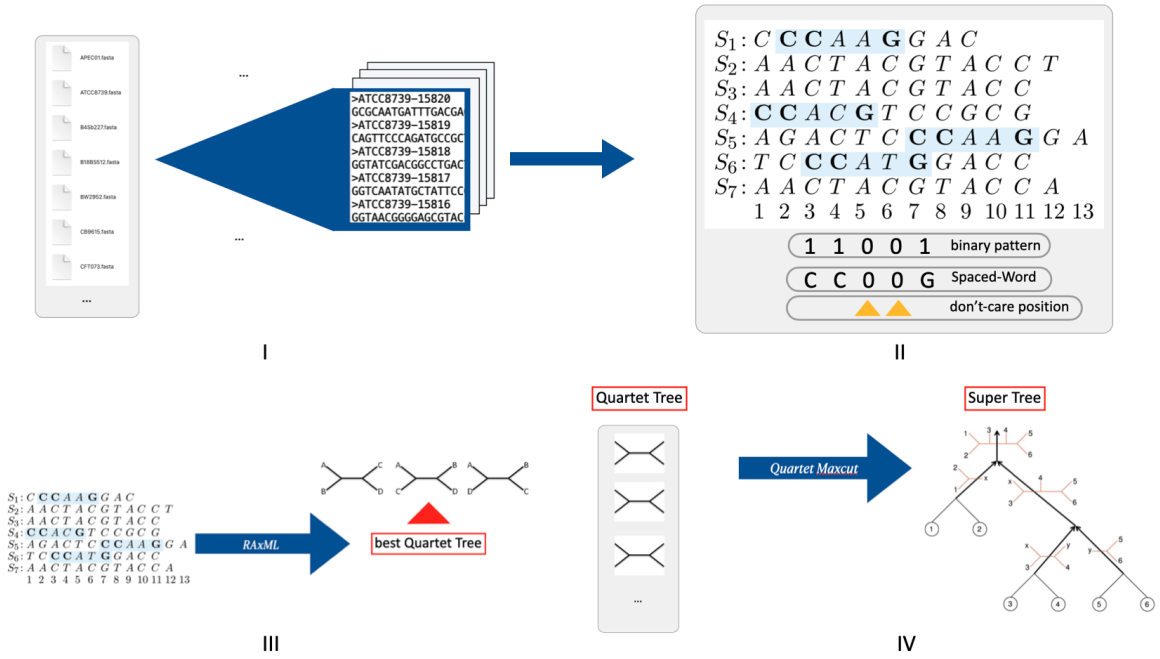

FIGURE 4.4: Workflow of MULTI-SPAM [5]. It involves four steps: data input, searching for *P-blocks*, constructing *quartet tree* and constructing *super tree*.

Similary to PROT-SPAM, MULTI-SPAM also uses the *spaced-word matches* to infer distances. However, the biggest difference is that it compares 4 sequences at the same time to seek the shared *spaced-word matches*. The workflow of MULTI-SPAM can be described as Figure 4.4:

1. Input: Unlike PROT-SPAM, MULTI-SPAM only accepts nucleotide sequences as input, where each FASTA file contains the sequences (assembled or unassembled) for each strain;

2. Searching for *P-blocks*: Next, the program uses RASBHARI (like PROT-SPAM) to generate a set of binary patterns with minimal overlap-complexity. Then the program tries to find a four-way 'micro alignments' corresponding

to the same *spaced-word match* in four different sequences. It is also called a *quartet P-block* or a *P-block*. In order to find the *P-blocks* efficiently, the program sorts all *spaced-words* with respect to the binary pattern in lexicographical order. In such case, the same *spaced-word* will appear contiguously. Then, the program extracts four *spaced-word* together as a *P-block*. The program proceeds until all *P-blocks* are exhausted.

3. *Quartet tree* construction: Once the *P-block* has been found, we can infer a small binary unrooted phylogenetic tree where on the same side infers the two closest target organisms. Such a phylogenetic tree is also called a *quartet tree* and it can possibly have 3 topologies. RAxML is a dependency tool which does this job, it uses Maximum Likelihood methods to infer the best of three topologies given a *P-block* of four sequences [55]. All the *P-blocks* are processed with RAxML and there is one binary unrooted tree generated for each;

4. *Super tree* construction: After the binary tree is generated for each P-block, all of the binary trees are amalgamated into a single *super tree* which spans all the target organisms. This is an NP-hard problem, a tool Quartet Maxcut with heuristic approach is applied here to find the optimal *super tree* which violates minimal number of the topologies from binary trees. Such *super tree* is the final phylogenetic tree [56, 57].

Installing and Running

Multi-SpaM is implemented in `C`, `C++`, and `Python`. The source code is available at `https://github.com/tdencker/multi-SpaM`. We downloaded and compiled it locally. Multi-SpaM uses only nucleotide sequences as input. For **AFproject Data**, the result of Multi-SpaM is already uploaded by the developers and is available in the bench-marking results. We downloaded its phylogeny tree directly there. For **Curated Data**, although the genome data of the 71 strains can be extracted as FASTA files, however not all of these 71 genome assemblies are complete. Some assemblies are in the scaffolding level. To achieve this, we tried to use only the genomes from the RefSeq database, but there the corresponding reference genomes are missing. Hence a complete genome assembly cannot be performed for each of these strains. Due to the constraint that Multi-SpaM only accepts connected sequences as input, we couldn't run Multi-SpaM on the **Curated Data**.

4.1.4.3 Sans-Serif

Workflow

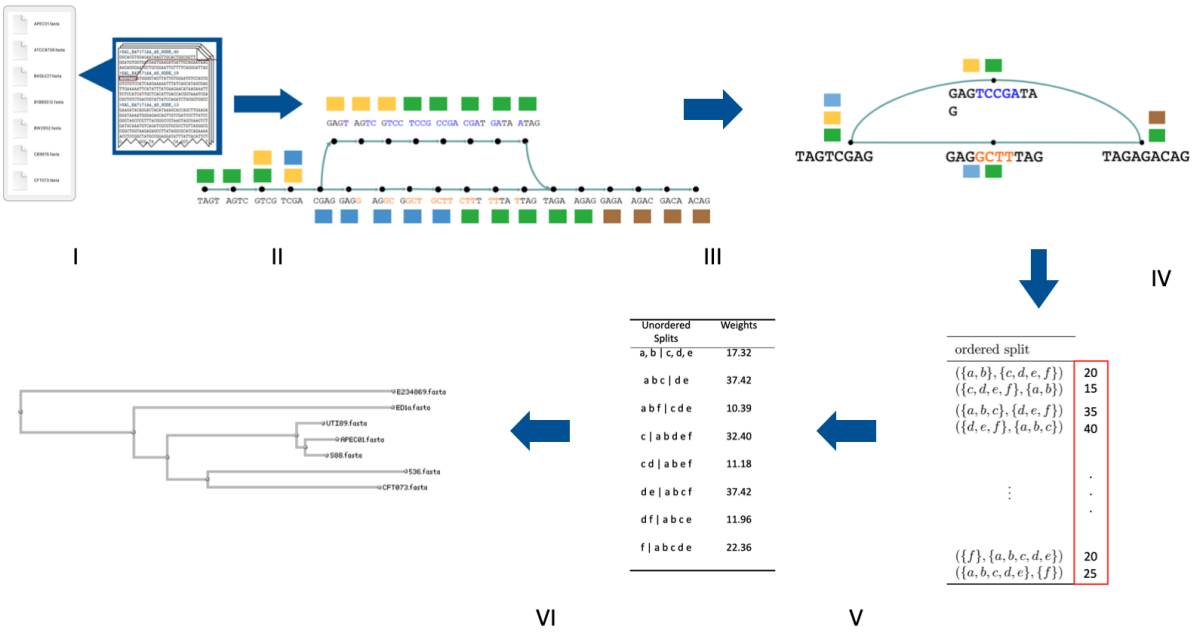

FIGURE 4.5: Workflow of SAN-SERIF [9]. It involves six steps: data input, constructing *colored de Bruijn Graph*, collapsing *colored de Bruijn Graph*, calculating weights of ordered splits, generating unordered splits and phylogenetic tree inference.

The workflow of SANS-SERIF can be described as Figure 4.5:

1. Input: SANS-SERIF takes a versatile range of inputs: assembled genomes, unassembled reads *etc.* It can also work with both nucleotide sequences and amino acid sequences;

2. *colored de Bruijn Graph* construction: the *colored de Bruijn Graph* is constructed using a dependency tool BIFROST [58]. It is a fast tool and easy to use. It can also deal with full genome sequences, assemblies, read data and their combinations;

3. Collapsing: Then the *colored de Bruijn Graph* is collapsed such than each node represents a *unitig* (a maximal non-branching path in the *de Bruijn Graph*, see details in ***Preliminaries*** 2.3);

4. Calculating weight of *ordered splits*: For each *unitig*, its separation signal is defined as the genomes to which this *unitig* belongs against the genomes which does not. For example, a *colored de Bruijn Graph* G is constructed based on 4 genomes $\{A, B, C, D\}$. A *unitig* u has 2 associated genomes $\{A, B\}$. Here we say the separation signal of u is $\{A, B \| C, D\}$. This separation signal is also called *ordered split*. We call it 'ordered' because $\{A, B \| C, D\}$ is different from $\{C, D \| A, B\}$. In fact, we call $\{A, B \| C, D\}$ as the *inverse split* of $\{C, D \| A, B\}$. We define the weight of such *ordered split* as the length of the *unitig*. For every *unitig*, there is one *ordered split* collected. For duplicated *ordered splits*, they are merged by adding up their corresponding weights together;

5. Generating *unordered splits*: After all the splits are collected and the corresponding weights are calculated, the *ordered split* with weight w_1 and its *inverse split* with weight w_2 will be merged as an *unordered split* with function $\sqrt{w_1 * w_2}$. A list of *unordered splits* is thus generated. Such a list forms a *split network* because it infers the splitting scenarios among the target genomes;

6. Tree inference: A dependency tool SplitsTree is used to infer the phylogenetic tree using the split network from the previous step. SplitsTree is a program which is able to incorporate both implicit and explicit network into standard phylogenetic trees [59].

Installing and Running

SANS-Serif is implemented in `C++`. The source code is available at `https://gitlab.ub.uni-bielefeld.de/gi/sans`. We downloaded and compiled the program locally. Sans-Serif can use a versatile range of data as input: unassembled reads, partially assembled genome, fully assembled genome, amino acid sequences. This is due to the characteristic that Sans-Serif constructs a `de Bruijn Graph` as its first step based on all input strain data. For this thesis, we want to compare the result of Sans-Serif and Prot-SpaM, hence we also used the same input for Prot-SpaM, which is the selected protein sequences of core genes of the 71 strains from **Curated Data** to feed Sans-Serif. We use the default k-mer size 10 for amino acid sequences as the configuration.

For **AFproject Data**, we use the same benchmarking data as Prot-SpaM and

MULTI-SPAM, which are 29 fully assembled *E.coli* genomes. We use the default k-mer size 31 for nucleotide sequences as the configuration.

4.1.5 Phylogenetic tree construction and visualisation

The outputs of PROT-SPAM and SANS-SERIF are already tree files in `Newick` `(.new)` format, while for PROT-SPAM, the output is a distance matrix in `PHYLIP` format. We used a web-based tool TREX to convert the distance matrix to `newick` format. TREX is a web service designed for the inference, validation, and visualization of phylogenetic trees and phylogenetic networks [60]. It has the service which can directly convert an input distance matrix in `PHYLIP` format into a tree file in `newick` format.

For phylogenetic tree visualisation, we used ICYTREE, an online service application to visualize a wide range of phylogenetic trees and networks. It is very light and suitable to handle large phylogenetic trees [61].

4.1.6 Tree comparisons

To compare and evaluate different phylogeny trees on the same target organisms, Robinson–Foulds (RF) distance is widely applied [39]. It measures the dissimilarity between two tree topologies with the same number of leaves and same labels (species) associating with the leaves and ignores branch lengths. Formally, Robinson *et al* defined the RF distance between two trees as the number of edit operations that are necessary to convert the first topology into the second one, or vice versa [6, 39]. In other words, the RF distance is the total number of bipartitions that can be obtained exclusively from one tree but not from the other. Figure 4.6 gives an example of how to calculate the RF distance score [10]. Tree 1 and Tree 2 both have 4 same nodes.

1. The bipartitions of Tree 1 are: $\{A, BCD\}$, $\{B, ACD\}$, $\{C, ABD\}$, $\{D, ABC\}$, $\{AB, CD\}$;

2. The bipartitions of Tree 2 are: $\{A, BCD\}$, $\{B, ACD\}$, $\{C, ABD\}$, $\{D, ABC\}$, $\{AC, BD\}$.

Hence the Robinson-Foulds distance between Tree 1 and Tree 2 are the sum of each unique bipartitions: $1 + 1 = 2$.

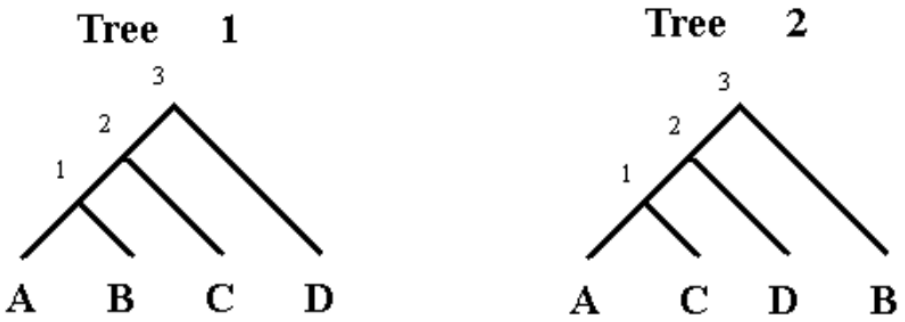

FIGURE 4.6: An example of Robinson-Foulds distance calculation created by Yuancheng [10]. Tree 1 and Tree 2 have in total 2 different bipartition cases ($\{AB, CD\}$ for Tree 1 and $\{AC, BD\}$ for Tree 2), hence their Robinson-Foulds distance is 2.

As the value of RF distance may depend on the number of input organisms, sometimes it is easier to access it using a normalized value. The normalized Robinson-Foulds metric (nRF) measures the normalized RF distance score. It is defined as the number of edit operations that are necessary to convert the first topology into the second one (RF distance) divided by the total number of possible bipartitions in the two trees. Thus, the nRF distance has values between 0 and 1 with 0 representing two trees sharing identical topology and 1 for there is no bipartition shared between the two trees (the trees are maximally different to each other). Using the same example in Figure 4.6, the total number of bipartitions is 6 and 2 bipartitions are unique to each tree. Hence the normalized Robinson-Foulds between Tree 1 and Tree 2 is: $2/6 = 0.33$.

Here in this thesis, we used the function `Tree.compare` from `Python` package ETE3 to calculate the nRF distances. ETE3 is a computational framework that focuses on the reconstruction, analysis, and visualization of phylogenetic trees [62].

4.2 Implementation

The implementation of this pipeline consists of the following five parts:

1. For downloading the sequence records from **NCBI**, implementation can be found in this link.
2. For preprocessing the data, i.e. extracting proteins from GenBank file, plotting the sequence length distribution, implementation can be found in this link.
3. For analysing the results from MMseqs2, i.e. plotting the U-shaped plot, selecting the representative protein clusters and assigning the proteins back to each corresponding strains, implementations can be found in this link.
4. For analysing the results of Prot-SpaM, namely processing the distance matrix, implementation can be found in this link.
5. For comparing different phylogenetic trees, implementation can be found in this link.

Chapter 5

Results

5.1 Data summary

5.1.1 Curated Data

In total we have collected 71 strains, which are summarized in table 5.1. A full list can be found in the ***Appendix*** A.2.

Genus	**Source**	**Number of Strains**
Pyxidicoccus	NCBI assembly	5
	NCBI nuccore	0
	HIPS in-house	3
Myxococcus	NCBI assembly	51
	NCBI nuccore	1
	HIPS in-house	11

Table 5.1: Summary of Curated Data.

In general, we collected these data from three categories: the in-house HIPS data, the records in the NCBI assembly database (assembly), and the records in the NCBI nucleotide database (nuccore). First we collected all sequences and then analyzed the length distribution. The idea is to gather as many high-quality genomes as possible. Here we defined the high-quality data as complete or near-complete assembled data. According to the genome records in NCBI Datasets, the common genome size of *Pyxidicoccus* is 12 - 13 million bp and the common genome size of *Myxococcus* is 9 - 12 million bp [63]. In this study, we categorised

the strains with genome size over 8 million bp as high-quality strains.

In summary, we collected 8 strains for *Pyxidicoccus* and 63 strains for *Myxococcus*. For the 8 strains in *Pyxicoccus*, we collect 3 strains from our own incubated HIPS database, 5 records collected from NCBI assembly database and 5 records in the NCBI nuccore database. However, all of these 5 records in the NCBI nucleotide database are duplicated records as in the assembly database hence they were discarded. For 63 strains in *Myxococcus*, we collected 11 in-house HIPS strains, 51 strains from NCBI assembly database, and 1 record in NCBI nuccore database which is not present in the results in NCBI assembly database. The length distribution histogram of all the 71 curated stains can be seen in Figure 5.1.

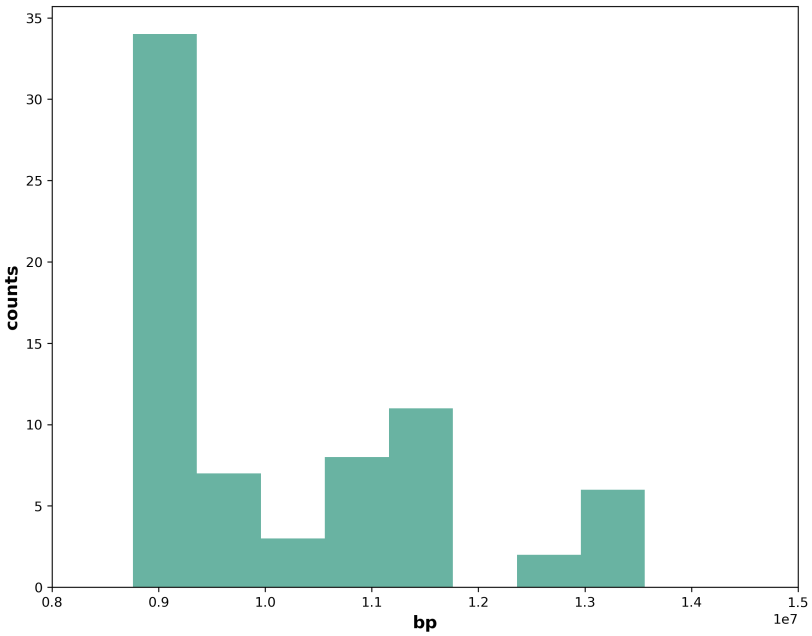

FIGURE 5.1: Length distribution of collected 71 *Pyxidicoccus* and *Myxococcus* strains as **Curated Data**.

5.1.2 AFproject Data

We downloaded the fully assembled genome of 29 strains of *E. coli* from **AFproject** server (`http://afproject.org/app/benchmark/genome/std/assembled/ecoli/dataset/`). The length distribution can be seen in Figure 5.2.

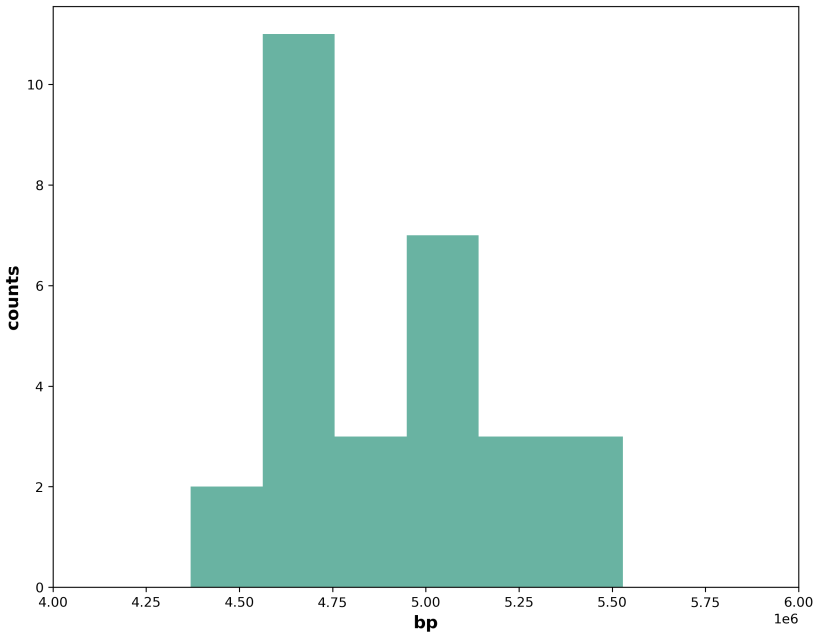

FIGURE 5.2: Length distribution of 29 fully assembled *E. coli* genomes from **AFproject Data**.

5.2 Protein clustering

5.2.1 Establishing protein sequence identity threshold

For **Curated Data**, we randomly selected 10 strains and extracted their RpoB, RpoC, and RpoD proteins. We used these protein sequences to perform pairwise sequence alignments using IDENT AND SIM, a web interface tool to calculate the identity and similarity of each pair of input sequences. IDENT AND SIM is part of SEQUENCE MANIPULATION SUITE, a collection of JavaScript programs to analyze short DNA and protein sequences [64]. The average sequence identity of these typically reserved protein sequences among these 10 strains is 95%. This served as the upper bound of sequence identity for the subsequent protein clustering.

5.2.2 Determining core and auxiliary genes

To select the proteins from core genes out of the whole proteome for each strain, we performed the protein clustering using MMSEQS2 using different sequence identities. As the previous step indicated, we started the clustering at sequence identity

95%, then gradually lowered this number until we could get a clear U-shape plot. We found that for both **Curated Data** and **AFproject Data**, using sequence identities of 50% to 70%, we could get a clear separation between proteins from core genes and auxiliary genes (**Curated Data:** Figure 5.3 - 5.5, **AFproject Data:** Figure 5.6 - 5.8). After inspecting the plots, we chose 50 and 15 as the threshold respectively for **Curated Data** and **AFproject Data**. The protein clusters over this threshold were extracted and later fed into alignment-free sequence comparison tools.

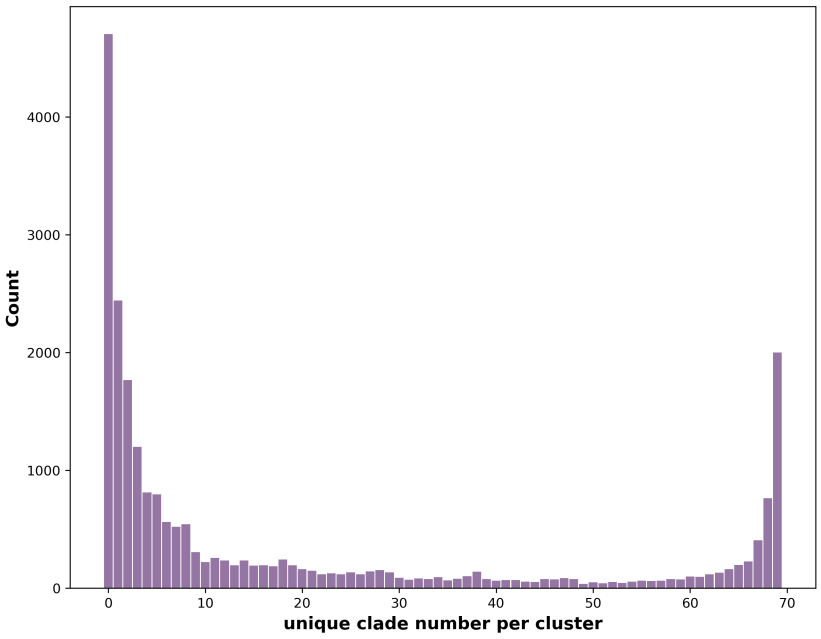

FIGURE 5.3: Histogram of the number of unique strains from the **Curated Data** protein clusters by MMseqs2 using clustering sequence identity threshold 50%. The histogram shows a clear U-shape curve. The far-left part indicates proteins from auxiliary genes and the far-right part indicates proteins from core genes.

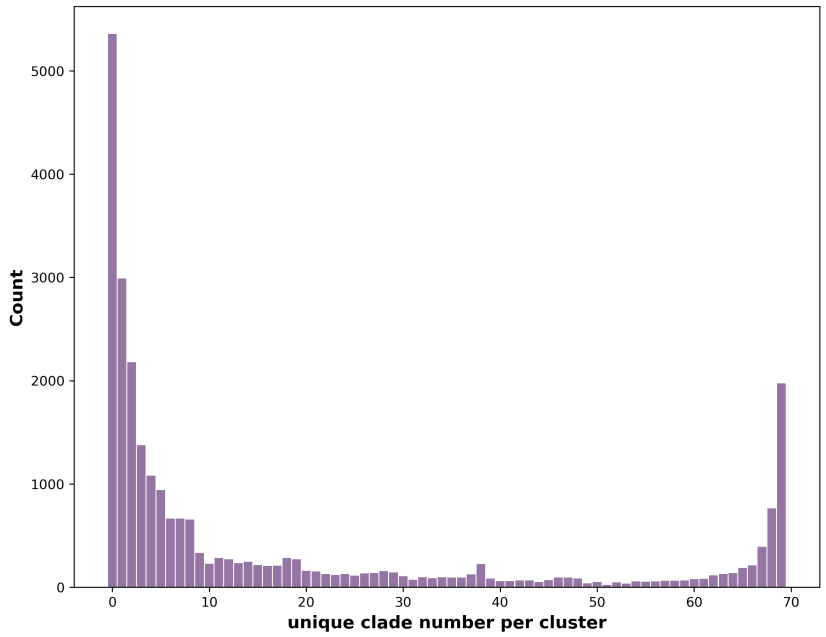

FIGURE 5.4: Histogram of the number of unique strains from the **Curated Data** protein clusters by MMSEQS2 using clustering sequence identity threshold 60%. The histogram shows a clear U-shape curve. The far-left part indicates proteins from auxiliary genes and the far-right part indicates proteins from core genes.

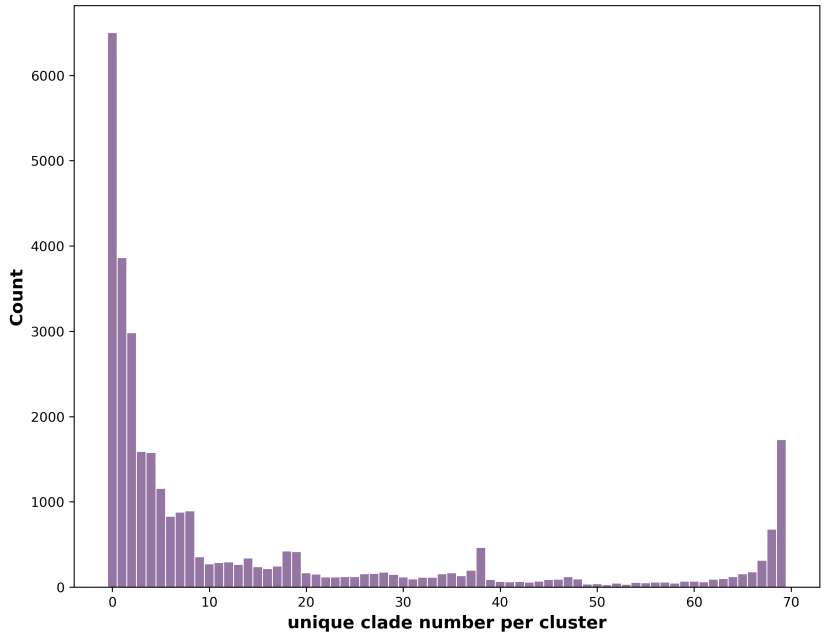

FIGURE 5.5: Histogram of the number of unique strains from the **Curated Data** protein clusters by MMSEQS2 using clustering sequence identity threshold 70%. The histogram shows a clear U-shape curve. The far-left part indicates proteins from auxiliary genes and the far-right part indicates proteins from core genes.

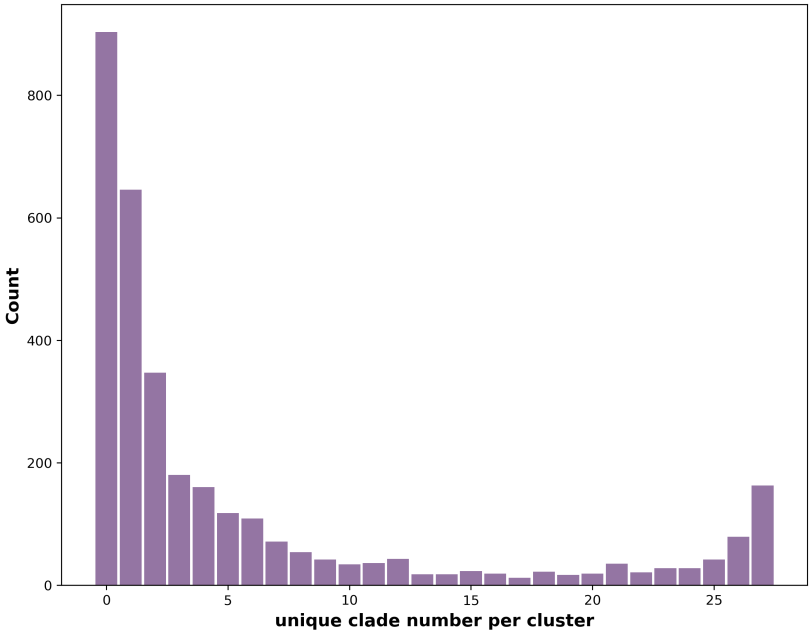

FIGURE 5.6: Histogram of the number of unique strains from the **AFproject Data** protein clusters by MMSEQS2 using clustering sequence identity threshold 50%. The histogram shows a clear U-shape curve. The far-left part indicates proteins from auxiliary genes and the far-right part indicates proteins from core genes.

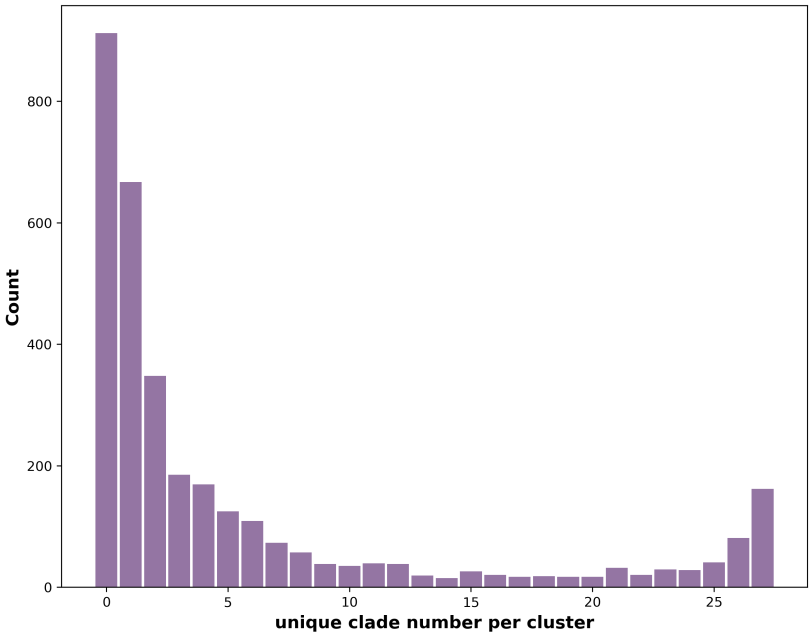

FIGURE 5.7: Histogram of the number of unique strains from the **AFproject Data** protein clusters by MMSEQS2 using clustering sequence identity threshold 60%. The histogram shows a clear U-shape curve. The far-left part indicates proteins from auxiliary genes and the far-right part indicates proteins from core genes.

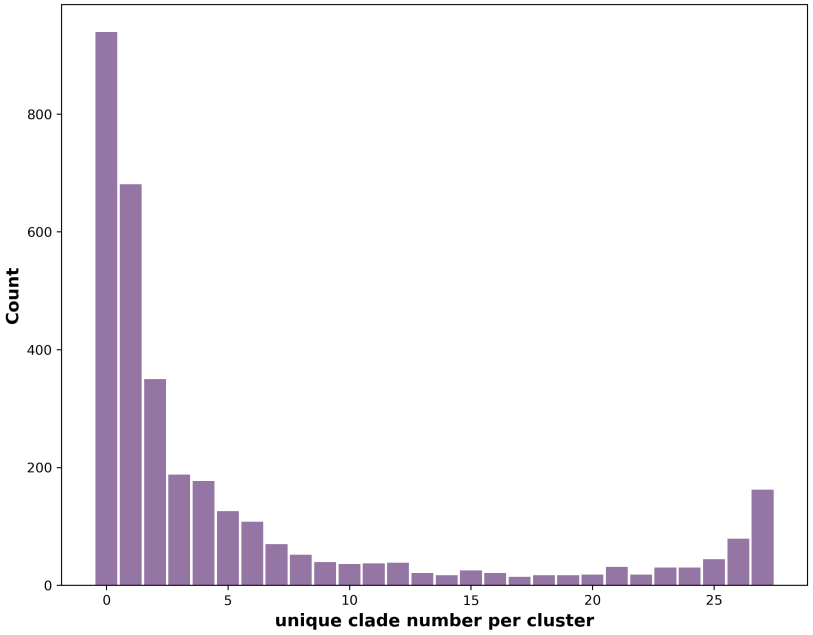

FIGURE 5.8: Histogram of the number of unique strains from the **AFproject Data** protein clusters by MMSEQS2 using clustering sequence identity threshold 70%. The histogram shows a clear U-shape curve. The far-left part indicates proteins from auxiliary genes and the far-right part indicates proteins from core genes.

For **Curated Data**, we selected 51.7%, 47.7%, and 39.5% of the total proteins under clustering sequence identity 50%, 60%, and 70%. While for **AFproject Data**, the proportion of proteins we chose does not have a significant difference in terms of using different sequence identities (Table. 5.2).

Sequence Identity	**Curated Data**	**AFproject Data**
0.5	51.7%	53.5%
0.6	47.7%	52.0%
0.7	39.5%	51.5%

TABLE 5.2: Proportions of chosen proteins for **Curated Data** and **AFproject Data** under different sequence identities 50% to 70% for protein clustering by MMSEQS2.

5.3 Phylogenetic trees

The phylogenetic trees of 71 *Pyxidicoccus/Myxococcus* strains using the full proteome can be found in Figure 5.9 (Prot-SpaM) and Figure 5.10 (SANS-Serif). Both two trees demonstrate a similar structure:

1. The total 71 strains with mixed *Pyxidicoccus* and *Myxococcus* are divided into two major groups;
2. The 8 *Pyxidicoccus* strains are grouped together to each other (in red block) yet integrated, separating from other *Myxococcus* strains;
3. One unclassified *Myxococcus* strain, *Myxococcus sp. RHSTA-1-4* is grouped along with other 8 *Pyxidicoccus* strains.

Phylogenetic trees using representative proteins from core genes are shown in Figure A.1 - A.6. They show similar conclusions as above. Moreover, the topologies of phylogenetic trees from Prot-SpaM using proteins of core genes with clustering thresholds of 0.5 to 0.7 are almost identical to each other. For the phylogenetic trees from SANS-Serif using proteins of core genes with clustering threshold 0.5 to 0.7, the distances between some strains are close to zero. Also, *Pyxidicoccus* strains are placed not all-together in the these trees, yet they are still very close to each other.

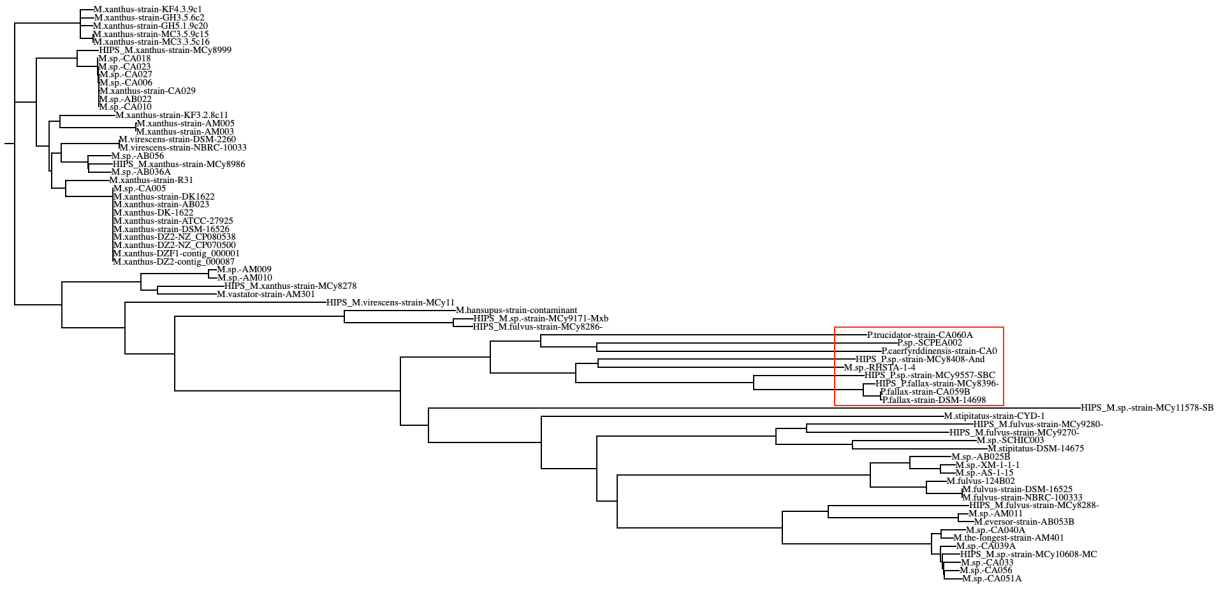

FIGURE 5.9: Phylogenetic tree of mixed 71 *Pyxidicoccus* and *Myxococcus* strains by PROT-SPAM using the whole proteome. The red block indicates the *Pyxidicoccus* strains. Description of each strain can be found in Appendix A.2.

FIGURE 5.10: Phylogenetic tree of mixed 71 *Pyxidicoccus* and *Myxococcus* strains by SANS-SERIF using the whole proteome. The red block indicates the *Pyxidicoccus* strains. Description of each strain can be found in Appendix A.2.

5.4 Tree comparisons

In this section, we will discuss the comparisons of different phylogenetic trees under different scenarios for **Curated Data** and **AFproject Data**.

5.4.1 Curated Data

Levels of protein clustering	PROT-SPAM vs. SANS-SERIF
no clustering	0.29
0.5 sequence identity	0.64
0.6 sequence identity	0.59
0.7 sequence identity	0.61

TABLE 5.3: normalized Robinson Foulds comparison between PROT-SPAM and SANS-SERIF on **Curated Data** using different clustering levels of proteins from core genes.

The comparison results between PROT-SPAM and SANS-SERIF on **Curated Data** are shown in Figure 5.3. We could see that using the full proteome (no protein clustering) as input, the difference between the trees constructed from these two methods is minimal (0.29) when compared to using representative proteins from core genes. This is possibly due to that PROT-SPAM is sensitive to using partial proteome as input. For the reason that MULTI-SPAM was not able to be applied to **Curated Data**, the comparisons involving it are neglected.

5.4.2 AFproject Data

5.4.2.1 Comparison against the reference tree

Table 5.4 and 5.5 show the results of comparing the trees constructed by PROT-SPAM and SANS-SERIF against the reference tree for the **AFproject Data**. We could see that for PROT-SPAM, the effect of using proteins from core genes does not have a major impact on the accuracy of constructed phylogenetic trees. Given the fact that for PROT-SPAM it is suggested to take the whole proteome as input, we may infer that the proteins we have selected are truly representative (Figure 5.4). It is also interesting to see that for SANS-SERIF, using the whole proteome hinders it from constructing the most accurate phylogenetic tree, which fulfills the purpose of this thesis to compare the impact of using the representative proteins

from core genes (Figure 5.5).

According to the benchmarking results from **AFproject**, the normalized Robinson Foulds score between the phylogeny tree constructed by Multi-SpaM (using whole proteome) and the reference tree is 0.27 and it ranks at the ninth position (9/61) [6]. Combining the results from Table. 5.4 and 5.5, we can conclude that Prot-SpaM performs slightly behind Multi-SpaM (0.31 vs. 0.27), and SANS-Serif has the best performance when using selected proteins as input.

Levels of protein clustering	Prot-SpaM vs. Reference tree
no clustering	0.31
0.5 sequence identity	0.31
0.6 sequence identity	0.30
0.7 sequence identity	0.30

Table 5.4: normalized Robinson Foulds score between Prot-SpaM and reference tree on **AFproject Data** using different clustering levels of proteins from core genes.

Levels of protein clustering	Sans-Serif vs. Reference tree
no clustering	0.31
0.5 sequence identity	0.15
0.6 sequence identity	0.19
0.7 sequence identity	0.19

Table 5.5: Normalized Robinson Foulds score between SANS-Serif and reference tree on **AFproject Data** using different clustering levels of proteins from core genes.

5.4.2.2 Comparison against each other

We also performed a comprehensive comparison between each pair of the three selected alignment-free sequence comparison tools, the results are shown in Table 5.6 - 5.8. We can see that with the more rapid protein clustering (higher sequence identity threshold), trees constructed by Prot-SpaM and SANS-Serif tend to be more similar (Table 5.6). Also, using different levels of protein clustering does not affect the distance between the trees generated by Prot-SpaM or SANS-Serif and Multi-SpaM (Table 5.7 - 5.8).

Levels of protein clustering	Prot-SpaM vs. Sans-Serif
no clustering	0.31
0.5 sequence identity	0.23
0.6 sequence identity	0.15
0.7 sequence identity	0.15

Table 5.6: Normalized Robinson Foulds score between Prot-SpaM and SANS-Serif on **AFproject Data** using different clustering levels of proteins from core genes.

Levels of protein clustering	Prot-SpaM vs. Multi-SpaM
no clustering	0.38
0.5 sequence identity	0.31
0.6 sequence identity	0.35
0.7 sequence identity	0.35

Table 5.7: Normalized Robinson Foulds score between Prot-SpaM and Multi-SpaM on **AFproject Data** using different clustering levels of proteins from core genes.

Levels of protein clustering	Sans-Serif vs. Multi-SpaM
no clustering	0.35
0.5 sequence identity	0.35
0.6 sequence identity	0.35
0.7 sequence identity	0.35

Table 5.8: Normalized Robinson Foulds score between SANS-Serif and Multi-SpaM on **AFproject Data** using different clustering levels of proteins from core genes.

Chapter 6

Conclusion

Phylogeny reconstruction, alongside other fundamental biology questions, needs to adapt to the current age of booming data. Sequencing data can be produced these days rapidly and easily, which brings great convenience. On the other hand, sequencing data can be incomplete, massive, redundant, and raw. In the context of this thesis, efficiently distilling and utilizing the essence of input data is a key to constructing an accurate phylogenetic tree.

Following two decades of rapid development, alignment-free sequence comparison has shown good prospects. Some of the tools provide a peek at finding a balance between comparison accuracy and efficiency. However, the purpose of alignment-free sequence comparison is never to replace alignment-based algorithms completely, but it is to be believed that comparing sequences can and should be diverse, according to different scenarios.

In this thesis, we reconstructed the phylogeny relationship among selected 71 mixed *Pyxidicoccus* and *Myxococcus* strains using three pioneering alignment-free tools. A comprehensive assessment of their performance was performed subsequently using data from a reference project. Results have shown that the *Pyxidicoccus* strains are grouped all together in the phylogenetic trees, which separates them from *Myxococcus* strains. Inside the group, we also found that there is one *Myxococcus* strain which might need to be reclassified as *Myxococcus*. The results may indicate that *Myxococcus* and *Pyxidicoccus* are regarded as separate genera. Further, the length of the branch that separates *Pyxidicoccus* from the rest of *Myxococcus* is comparable to other branches in *Myxococcus* tree. For future analysis, we need an outgroup species to establish the truth whether it is a separate branch

or *Pyxidicoccus* is embedded into the *Myxococcus* subtree. Also, benchmarking results using public project data indicated that SANS-Serif slightly outperforms Prot-SpaM and Multi-SpaM, while the latter two methods were still considered to be able to deliver good results.

For future work, we want to:

1. Build an end-to-end pipeline for collecting strain data, pre-processing data, installing and running alignment-free tools under different configurations;
2. Identify a more accurate way of selecting proteins from core genes. The results shown in this thesis by MMseqs2 are not ideal, the U-shape plots were slightly distorted. We are interested on investigating or benchmarking other major clustering tools;
3. Use more genomes that are currently being sequenced in HIPS to confirm the findings.

We believe that future alignment-free sequence comparison tools will touch upon deeper mechanisms underpinning the sequence codes and show their reliability.

Bibliography

[1] Rachel M Colquhoun, Michael B Hall, Leandro Lima, Leah W Roberts, Kerri M Malone, Martin Hunt, Brice Letcher, Jane Hawkey, Sophie George, Louise Pankhurst, and Zamin Iqbal. Pandora: nucleotide-resolution bacterial pan-genomics with reference graphs. *Genome Biol*, 22(1):267, sep 2021. ISSN 1474-760X.

[2] Anna-Katharina Lau, Svenja Dörrer, Chris-André Leimeister, Christoph Bleidorn, and Burkhard Morgenstern. Read-SpaM: assembly-free and alignment-free comparison of bacterial genomes with low sequencing coverage. *BMC Bioinformatics*, 20(Suppl 20):638, dec 2019.

[3] Chris-André Leimeister, Salma Sohrabi-Jahromi, and Burkhard Morgenstern. Fast and accurate phylogeny reconstruction using filtered spaced-word matches. *Bioinformatics*, 33(7):971–979, apr 2017.

[4] Olga Kalinina. Lecture notes in algorithms for sequence analysis, July 2020.

[5] Thomas Dencker, Chris-André Leimeister, Michael Gerth, Christoph Bleidorn, Sagi Snir, and Burkhard Morgenstern. 'multi-SpaM': a maximum-likelihood approach to phylogeny reconstruction using multiple spaced-word matches and quartet trees. *NAR Genom Bioinform*, 2(1):lqz013, mar 2020.

[6] Andrzej Zielezinski, Hani Z Girgis, Guillaume Bernard, Chris-Andre Leimeister, Kujin Tang, Thomas Dencker, Anna Katharina Lau, Sophie Röhling, Jae Jin Choi, Michael S Waterman, Matteo Comin, Sung-Hou Kim, Susana Vinga, Jonas S Almeida, Cheong Xin Chan, Benjamin T James, Fengzhu Sun, Burkhard Morgenstern, and Wojciech M Karlowski. Benchmarking of alignment-free sequence comparison methods. *Genome Biol*, 20(1):144, jul 2019.

[7] Martin Steinegger and Johannes Söding. MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nat Biotechnol*, 35 (11):1026–1028, nov 2017.

[8] Chris-Andre Leimeister, Jendrik Schellhorn, Svenja Dörrer, Michael Gerth, Christoph Bleidorn, and Burkhard Morgenstern. Prot-SpaM: fast alignment-free phylogeny reconstruction based on whole-proteome sequences. *Gigascience*, 8(3), mar 2019.

[9] Roland Wittler. Alignment- and reference-free phylogenomics with colored de bruijn graphs. *Algorithms Mol Biol*, 15:4, apr 2020. ISSN 1748-7188.

[10] Yuancheng Wang. Performance of supertree methods for estimating species trees. Master's thesis, University of Canterbury, New Zealand, 2010.

[11] Conrad L Schoch, Stacy Ciufo, Mikhail Domrachev, Carol L Hotton, Sivakumar Kannan, Rogneda Khovanskaya, Detlef Leipe, Richard Mcveigh, Kathleen O'Neill, Barbara Robbertse, Shobha Sharma, Vladimir Soussov, John P Sullivan, Lu Sun, Seán Turner, and Ilene Karsch-Mizrachi. NCBI taxonomy: a comprehensive update on curation, resources and tools. *Database (Oxford)*, 2020, jan 2020.

[12] Kathrin I Mohr, Marc Stechling, Joachim Wink, Elke Wilharm, and Marc Stadler. Comparison of myxobacterial diversity and evaluation of isolation success in two niches: Kiritimati island and german compost. *Microbiology-open*, 5(2):268–278, apr 2016.

[13] Paul G Livingstone, Russell M Morphew, and David E Whitworth. Myxobacteria are able to prey broadly upon clinically-relevant pathogens, exhibiting a prey range which cannot be explained by phylogeny. *Front Microbiol*, 8:1593, aug 2017.

[14] Juliane Korp, María S Vela Gurovic, and Markus Nett. Antibiotics from predatory bacteria. *Beilstein J Org Chem*, 12:594–607, mar 2016.

[15] Paul G Livingstone, Russell M Morphew, and David E Whitworth. Genome sequencing and pan-genome analysis of 23 corallococcus spp. strains reveal unexpected diversity, with particular plasticity of predatory gene sets. *Front Microbiol*, 9:3187, dec 2018. ISSN 1664-302X.

[16] Lucky S Mulwa and Marc Stadler. Antiviral compounds from myxobacteria. *Microorganisms*, 6(3), jul 2018.

[17] Ronald Garcia, Klaus Gerth, Marc Stadler, Irineo J Dogma, and Rolf Müller. Expanded phylogeny of myxobacteria and evidence for cultivation of the 'unculturables'. *Mol Phylogenet Evol*, 57(2):878–887, nov 2010.

[18] C Jill Harrison and Jane A Langdale. A step by step guide to phylogeny reconstruction. *Plant J*, 45(4):561–572, feb 2006.

[19] David Penny. Inferring phylogenies.felsenstein. 2003. sinauer associates, sunderland, massachusetts. *Syst Biol*, 53(4):669–670, aug 2004. ISSN 1076-836X.

[20] Charles D. Michener and Robert R. Sokal. A quantitative approach to a problem in classification. *Evolution*, 11(2):130, jun 1957. ISSN 00143820.

[21] Benny Chor and Tamir Tuller. Maximum likelihood of evolutionary trees: hardness and approximation. *Bioinformatics*, 21 Suppl 1:i97–106, jun 2005.

[22] Joseph Felsenstein. *Inferring Phylogenies*. Sinauer Associates is an imprint of Oxford University Press, Sunderland, Mass, 2 edition, sep 2003. ISBN 978-0-87893-177-4.

[23] Hervé Tettelin, Vega Masignani, Michael J Cieslewicz, Claudio Donati, Duccio Medini, Naomi L Ward, Samuel V Angiuoli, Jonathan Crabtree, Amanda L Jones, A Scott Durkin, Robert T Deboy, Tanja M Davidsen, Marirosa Mora, Maria Scarselli, Immaculada Margarit y Ros, Jeremy D Peterson, Christopher R Hauser, Jaideep P Sundaram, William C Nelson, Ramana Madupu, Lauren M Brinkac, Robert J Dodson, Mary J Rosovitz, Steven A Sullivan, Sean C Daugherty, Daniel H Haft, Jeremy Selengut, Michelle L Gwinn, Liwei Zhou, Nikhat Zafar, Hoda Khouri, Diana Radune, George Dimitrov, Kisha Watkins, Kevin J B O'Connor, Shannon Smith, Teresa R Utterback, Owen White, Craig E Rubens, Guido Grandi, Lawrence C Madoff, Dennis L Kasper, John L Telford, Michael R Wessels, Rino Rappuoli, and Claire M Fraser. Genome analysis of multiple pathogenic isolates of streptococcus agalactiae: implications for the microbial "pan-genome". *Proc Natl Acad Sci USA*, 102 (39):13950–13955, sep 2005. ISSN 0027-8424.

[24] Jeanneth Mosquera-Rendón, Ana M Rada-Bravo, Sonia Cárdenas-Brito, Mauricio Corredor, Eliana Restrepo-Pineda, and Alfonso Benítez-Páez. Pangenome-wide and molecular evolution analyses of the pseudomonas aeruginosa species. *BMC Genomics*, 17:45, jan 2016.

[25] Andrzej Zielezinski, Susana Vinga, Jonas Almeida, and Wojciech M Karlowski. Alignment-free sequence comparison: benefits, applications, and tools. *Genome Biol*, 18(1):186, oct 2017.

[26] S F Altschul, T L Madden, A A Schäffer, J Zhang, Z Zhang, W Miller, and D J Lipman. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res*, 25(17):3389–3402, sep 1997.

[27] W R Pearson and D J Lipman. Improved tools for biological sequence comparison. *Proc Natl Acad Sci USA*, 85(8):2444–2448, apr 1988.

[28] J D Thompson, D G Higgins, and T J Gibson. CLUSTAL w: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Res*, 22(22):4673–4680, nov 1994.

[29] Susana Vinga and Jonas Almeida. Alignment-free sequence comparison-a review. *Bioinformatics*, 19(4):513–523, mar 2003.

[30] Nicolas Terrapon, January Weiner, Sonja Grath, Andrew D Moore, and Erich Bornberg-Bauer. Rapid similarity search of proteins using alignments of domain arrangements. *Bioinformatics*, 30(2):274–281, jan 2014.

[31] Yingnan Cong, Yao-Ban Chan, and Mark A Ragan. A novel alignment-free method for detection of lateral genetic transfer based on TF-IDF. *Sci Rep*, 6:30308, jul 2016.

[32] G E Fox, L J Magrum, W E Balch, R S Wolfe, and C R Woese. Classification of methanogenic bacteria by 16S ribosomal RNA characterization. *Proc Natl Acad Sci USA*, 74(10):4537–4541, oct 1977.

[33] Bernice Sacks Lipkin. *String Processing And Text Manipulation In C: Selected Data Structures And Techniques/book And Disk (prentice Hall Series On Programming Tools And M)*. Prentice Hall, Englewood Cliffs, N.J, 1995. ISBN 0131214438.

[34] Roland Wittler. Alignment- and reference-free phylogenomics with colored de bruijn graphs. *Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik GmbH, Wadern/Saarbruecken, Germany*, 2019.

[35] F Chiaromonte, V B Yap, and W Miller. Scoring pairwise genomic sequence alignments. *Pac Symp Biocomput*, pages 115–126, 2002.

[36] S Henikoff and J G Henikoff. Amino acid substitution matrices from protein blocks. *Proc Natl Acad Sci USA*, 89(22):10915–10919, nov 1992.

[37] Ying Wu, An Chun Cheng, Ming Shu Wang, De Kang Zhu, and Xiao Yue Chen. Molecular cloning and phylogenetic analysis of the duck enteritis virus UL55 gene. *Applied Multivariate Research*, 204-210:663–671, feb 2011. ISSN 1662-8985.

[38] Torsten Seemann. Prokka: rapid prokaryotic genome annotation. *Bioinformatics*, 30(14):2068–2069, jul 2014. ISSN 1367-4803.

[39] D.F. Robinson and L.R. Foulds. Comparison of phylogenetic trees. *Mathematical Biosciences*, 53(1-2):131–147, feb 1981. ISSN 00255564.

[40] Sayers E. A general introduction to the e-utilities. in: Entrez programming utilities help [internet]. *Bethesda (MD): National Center for Biotechnology Information (US)*, 2010-. URL `https://www.ncbi.nlm.nih.gov/books/NBK25497/`.

[41] Kans J. Entrez direct: E-utilities on the unix command line. *Bethesda (MD): National Center for Biotechnology Information (US)*, apr 2013 Apr 23. URL `https://www.ncbi.nlm.nih.gov/books/NBK179288/`.

[42] Huiguang Yi and Li Jin. Co-phylog: an assembly-free phylogenomic approach for closely related organisms. *Nucleic Acids Res*, 41(7):e75, apr 2013.

[43] Bernhard Haubold, Fabian Klötzl, and Peter Pfaffelhuber. andi: fast and accurate estimation of evolutionary distances between closely related genomes. *Bioinformatics*, 31(8):1169–1175, apr 2015.

[44] Ngoc Hieu Tran and Xin Chen. Comparison of next-generation sequencing samples using compression-based distances and its application to phylogenetic reconstruction. *BMC Research Notes*, 7:320, may 2014.

[45] Guido Van Rossum and Fred L Drake Jr. *Python reference manual*. Centrum voor Wiskunde en Informatica Amsterdam, 1995.

[46] Guido Van Rossum and Fred L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009. ISBN 1441412697.

[47] Peter J A Cock, Tiago Antao, Jeffrey T Chang, Brad A Chapman, Cymon J Cox, Andrew Dalke, Iddo Friedberg, Thomas Hamelryck, Frank Kauff, Bartek

Wilczynski, and Michiel J L de Hoon. Biopython: freely available python tools for computational molecular biology and bioinformatics. *Bioinformatics*, 25 (11):1422–1423, jun 2009.

[48] Doug Hyatt, Gwo-Liang Chen, Philip F Locascio, Miriam L Land, Frank W Larimer, and Loren J Hauser. Prodigal: prokaryotic gene recognition and translation initiation site identification. *BMC Bioinformatics*, 11:119, mar 2010.

[49] Karin Lagesen, Peter Hallin, Einar Andreas R, Hans-Henrik Staerfeldt, TorbjRognes, and David W Ussery. RNAmmer: consistent and rapid annotation of ribosomal RNA genes. *Nucleic Acids Res*, 35(9):3100–3108, jan 2007.

[50] Dean Laslett and Bjorn Canback. ARAGORN, a program to detect tRNA genes and tmRNA genes in nucleotide sequences. *Nucleic Acids Res*, 32(1): 11–16, jan 2004.

[51] Thomas Nordahl Petersen, SBrunak, Gunnar von Heijne, and Henrik Nielsen. SignalP 4.0: discriminating signal peptides from transmembrane regions. *Nat Methods*, 8(10):785–786, sep 2011.

[52] Diana L Kolbe and Sean R Eddy. Fast filtering for RNA homology search. *Bioinformatics*, 27(22):3102–3109, nov 2011.

[53] Lars Hahn, Chris-André Leimeister, Rachid Ounit, Stefano Lonardi, and Burkhard Morgenstern. rasbhari: Optimizing spaced seeds for database searching, read mapping and alignment-free sequence comparison. *PLoS Comput Biol*, 12(10):e1005107, oct 2016.

[54] M Kimura. The neutral theory of molecular evolution. *Sci Am*, 241(5):98–100, 102, 108 passim, nov 1979.

[55] Alexandros Stamatakis. RAxML version 8: a tool for phylogenetic analysis and post-analysis of large phylogenies. *Bioinformatics*, 30(9):1312–1313, may 2014.

[56] Sagi Snir and Satish Rao. Quartets MaxCut: a divide and conquer quartets algorithm. *IEEE/ACM Trans Comput Biol Bioinform*, 7(4):704–718, dec 2010.

[57] Sagi Snir and Satish Rao. Quartet MaxCut: a fast algorithm for amalgamating quartet trees. *Mol Phylogenet Evol*, 62(1):1–8, jan 2012.

[58] Guillaume Holley and Páll Melsted. Bifrost: highly parallel construction and indexing of colored and compacted de bruijn graphs. *Genome Biol*, 21(1): 249, sep 2020.

[59] Tobias H Kloepper and Daniel H Huson. Drawing explicit phylogenetic networks and their integration into SplitsTree. *BMC Evol Biol*, 8:22, jan 2008.

[60] Alix Boc, Alpha Boubacar Diallo, and Vladimir Makarenkov. T-REX: a web server for inferring, validating and visualizing phylogenetic trees and networks. *Nucleic Acids Res*, 40(Web Server issue):W573–9, jul 2012.

[61] Timothy G Vaughan. IcyTree: rapid browser-based visualization for phylogenetic trees and networks. *Bioinformatics*, 33(15):2392–2394, aug 2017.

[62] Jaime Huerta-Cepas, François Serra, and Peer Bork. ETE 3: reconstruction, analysis, and visualization of phylogenomic data. *Mol Biol Evol*, 33(6):1635–1638, jun 2016.

[63] Eric W Sayers, Evan E Bolton, J Rodney Brister, Kathi Canese, Jessica Chan, Donald C Comeau, Ryan Connor, Kathryn Funk, Chris Kelly, Sunghwan Kim, Tom Madej, Aron Marchler-Bauer, Christopher Lanczycki, Stacy Lathrop, Zhiyong Lu, Francoise Thibaud-Nissen, Terence Murphy, Lon Phan, Yuri Skripchenko, Tony Tse, Jiyao Wang, Rebecca Williams, Barton W Trawick, Kim D Pruitt, and Stephen T Sherry. Database resources of the national center for biotechnology information. *Nucleic Acids Research*, 50(D1):D20–D26, jan 2022.

[64] P Stothard. The sequence manipulation suite: JavaScript programs for analyzing and formatting protein and DNA sequences. *BioTechniques*, 28(6): 1102, 1104, jun 2000.

[65] Huan Fan, Anthony R Ives, Yann Surget-Groba, and Charles H Cannon. An assembly and alignment-free method of phylogeny reconstruction from next-generation sequencing data. *BMC Genomics*, 16:522, jul 2015.

[66] Brian B Luczak, Benjamin T James, and Hani Z Girgis. A survey and evaluations of histogram-based statistics in alignment-free sequence comparison. *Briefings in Bioinformatics*, 20(4):1222–1237, jul 2019.

[67] Yang Young Lu, Kujin Tang, Jie Ren, Jed A Fuhrman, Michael S Waterman, and Fengzhu Sun. CAFE: aCcelerated alignment-FrEe sequence analysis. *Nucleic Acids Research*, 45(W1):W554–W559, jul 2017.

[68] Gregory E Sims, Se-Ran Jun, Guohong A Wu, and Sung-Hou Kim. Alignment-free genome comparison with feature frequency profiles (FFP) and optimal resolutions. *Proc Natl Acad Sci USA*, 106(8):2677–2682, feb 2009.

[69] Cheong Xin Chan, Guillaume Bernard, Olivier Poirion, James M Hogan, and Mark A Ragan. Inferring phylogenies of evolving sequences without multiple sequence alignment. *Scientific Reports*, 4:6504, sep 2014.

[70] Kevin D Murray, Christfried Webers, Cheng Soon Ong, Justin Borevitz, and Norman Warthmann. kWIP: The k-mer weighted inner product, a de novo estimator of genetic similarity. *PLoS Computational Biology*, 13(9):e1005727, sep 2017.

[71] Fabian Klötzl and Bernhard Haubold. Phylonium: fast estimation of evolutionary distances from large samples of similar genomes. *Bioinformatics*, 36 (7):2040–2046, apr 2020.

[72] Sharma V Thankachan, Sriram P Chockalingam, Yongchao Liu, Alberto Apostolico, and Srinivas Aluru. ALFRED: A practical method for alignment-free distance computation. *Journal of Computational Biology*, 23(6):452–460, jun 2016.

[73] Sharma V Thankachan, Sriram P Chockalingam, Yongchao Liu, Ambujam Krishnan, and Srinivas Aluru. A greedy alignment-free distance estimator for phylogenetic inference. *BMC Bioinformatics*, 18(Suppl 8):238, jun 2017.

[74] Bernhard Haubold, Peter Pfaffelhuber, Mirjana Domazet-Loso, and Thomas Wiehe. Estimating mutation distances from unaligned genomes. *Journal of Computational Biology*, 16(10):1487–1500, oct 2009.

[75] Brian D Ondov, Todd J Treangen, Páll Melsted, Adam B Mallonee, Nicholas H Bergman, Sergey Koren, and Adam M Phillippy. Mash: fast genome and metagenome distance estimation using MinHash. *Genome Biol*, 17(1):132, jun 2016.

[76] Sophie Röhling and Burkhard Morgenstern. The number of spaced-word matches between two DNA sequences as a function of the underlying pattern weight. *BioRxiv*, jan 2019.

[77] Shahab Sarmashghi, Kristine Bohmann, M Thomas P Gilbert, Vineet Bafna, and Siavash Mirarab. Skmer: assembly-free and alignment-free sample identification using genome skims. *Genome Biol*, 20(1):34, feb 2019.

Appendix A

Appendix

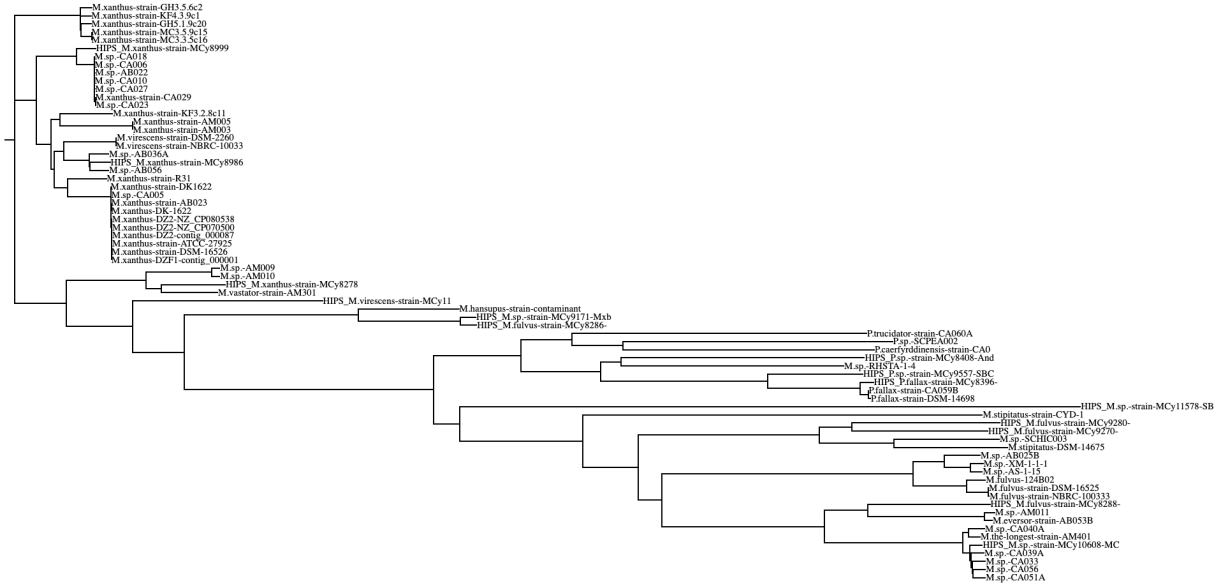

Figure A.1: Phylogenetic tree of mixed 71 *Pyxidicoccus* and *Myxococcus* strains by Prot-SpaM using the proteins from core genes under protein clustering sequence identity 0.5. Description of each strain can be found in Appendix A.2.

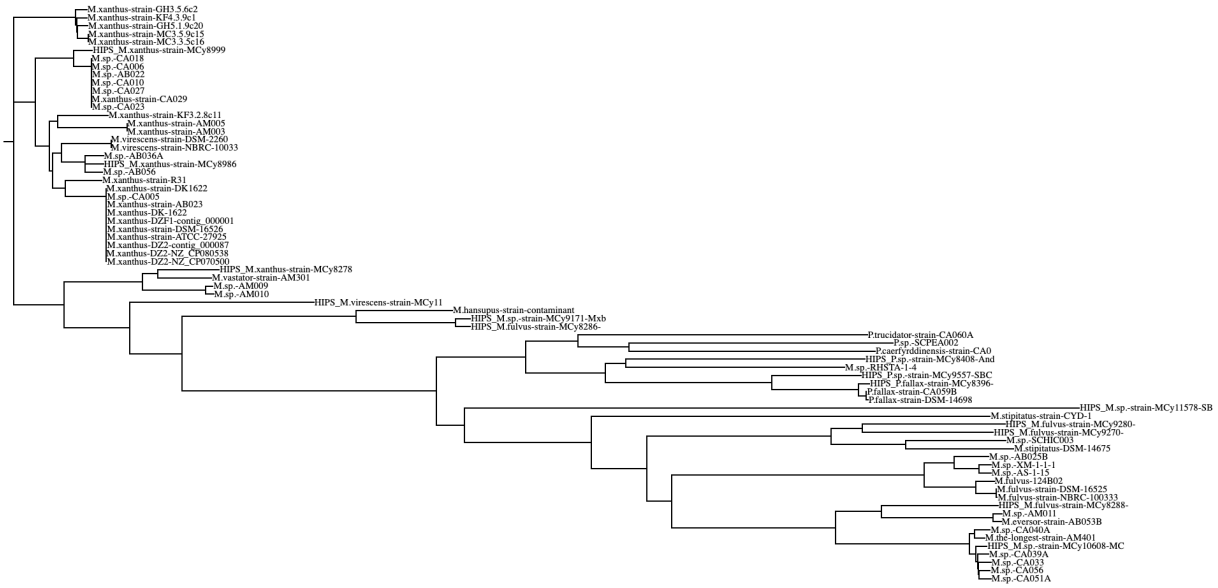

FIGURE A.2: Phylogenetic tree of mixed 71 *Pyxidicoccus* and *Myxococcus* strains by PROT-SPAM using the proteins from core genes under protein clustering sequence identity 0.6. Description of each strain can be found in Appendix A.2.

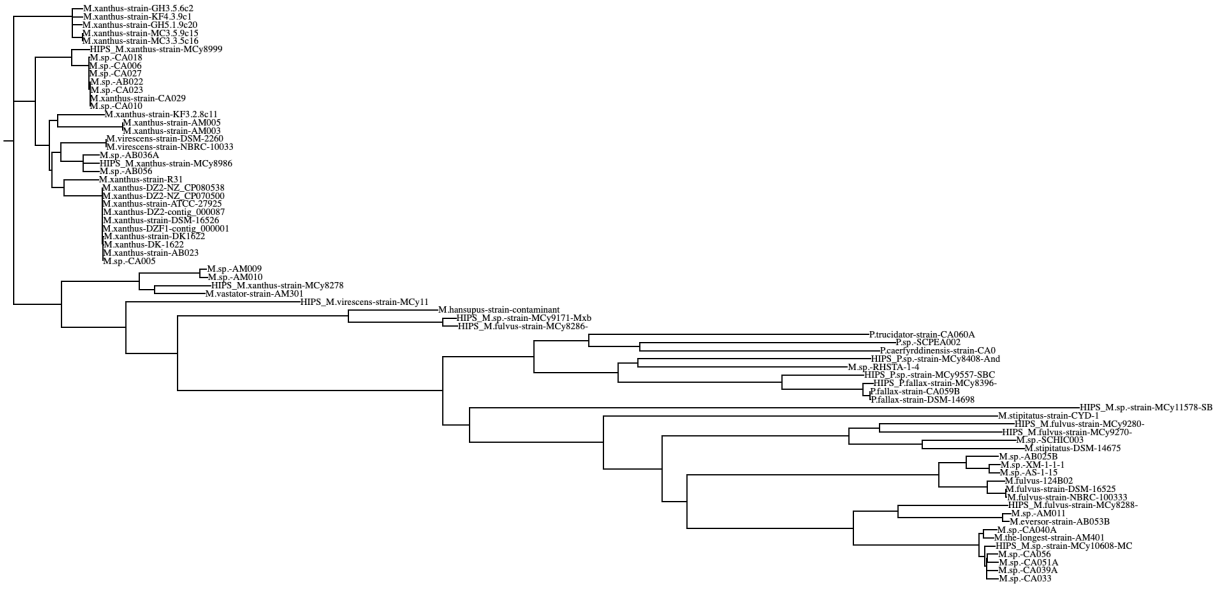

FIGURE A.3: Phylogenetic tree of mixed 71 *Pyxidicoccus* and *Myxococcus* strains by PROT-SPAM using the proteins from core genes under protein clustering sequence identity 0.7. Description of each strain can be found in Appendix A.2.

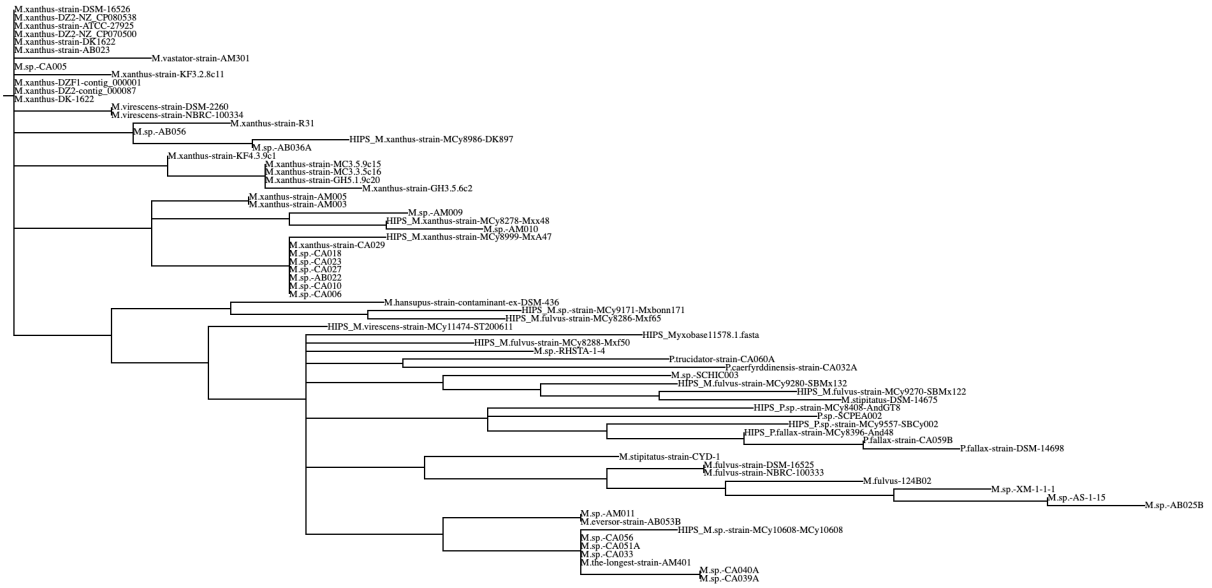

FIGURE A.4: Phylogenetic tree of mixed 71 *Pyxidicoccus* and *Myxococcus* strains by SANS-SERIF using the proteins from core genes under protein clustering sequence identity 0.5. Description of each strain can be found in Appendix A.2.

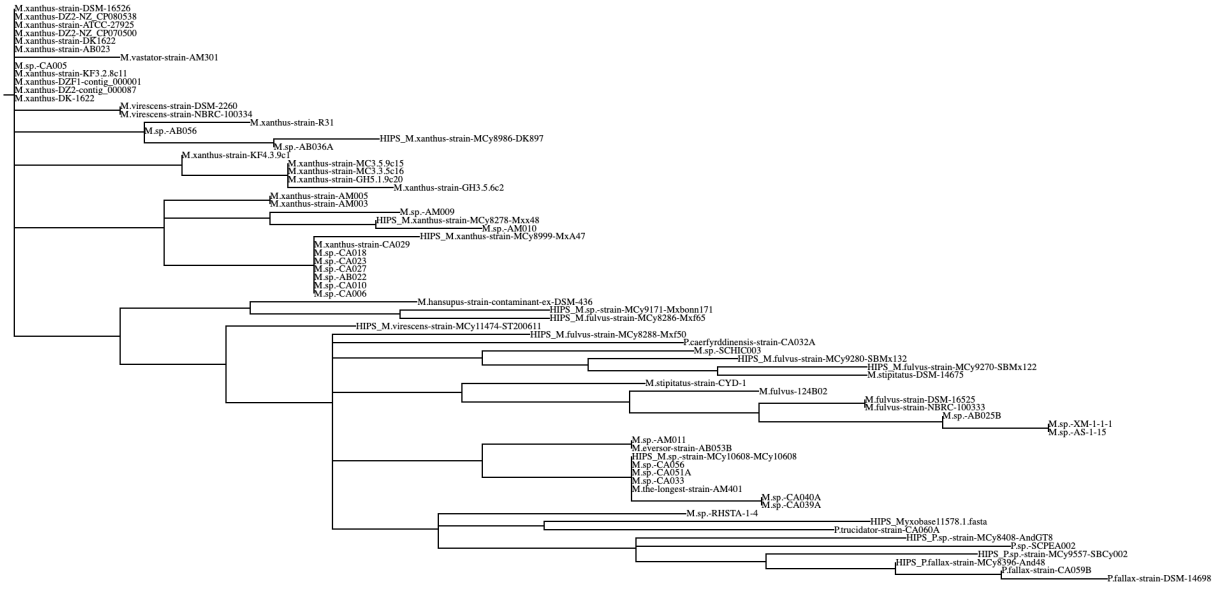

FIGURE A.5: Phylogenetic Tree of mixed 71 *Pyxidicoccus* and *Myxococcus* strains by SANS-SERIF using the proteins from core genes under protein clustering sequence identity 0.6. Description of each strain can be found in Appendix A.2.

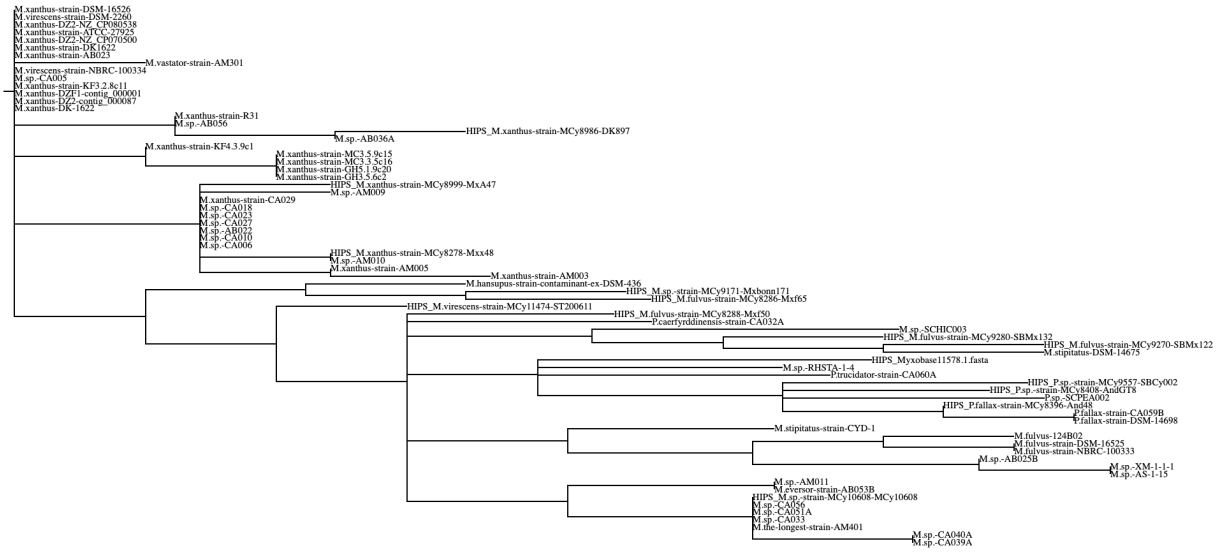

FIGURE A.6: Phylogenetic tree of mixed 71 *Pyxidicoccus* and *Myxococcus* strains by SANS-SERIF using the proteins from core genes under protein clustering sequence identity 0.7. Description of each strain can be found in Appendix A.2.

Software	Category	Version	Year	Availability
AAF [65]	k-mer count	NA	2015	https://github.com/fanhuan/AAF
AFKS [66]		1.0	2017	https://github.com/TulsaBioinformaticsToolsmith/Alignment-Free-Kmer-Statistics
ALFPY [25]		1.0.6	2017	https://github.com/aziele/alfpy
CAFÉ [67]		1.0.0	2018	https://github.com/younglululu/CAFE
FFP [68]		2v.4.0	2017	https://github.com/jaejinchoi/FFP
JD2STAT [69]		1.0	2014	http://bioinformatics.org.au/tools/jD2Stat/
KWIP [70]		0.2.0	2015	https://github.com/kdmurray91/kWIP
ANDI [43]	Micro-alignment	v0.14	2017	https://github.com/EvolBioInf/andi
CO-PHYLOG [42]		NA	2010	https://github.com/yhg926/co-phylog
⋄PROT-SPAM [8]		NA	2019	https://github.com/jschellh/ProtSpaM
⋄MULTI-SPAM [5]		1.0	2019	https://github.com/tdencker/multi-SpaM
PHYLONIUM [71]		v1.6	2018	https://github.com/kloetzl/phylonium
⋄SANS-SERIF [9]	de Bruijn Graph based	NA	2019	https://gitlab.ub.uni-bielefeld.de/gi/sans
KMACS	Length of common substrings	1.0	2014	http://kmacs.gobics.de/
ALFRED-G [72, 73]		1.0	2015	https://alurulab.cc.gatech.edu/phylo
KR [74]		2.0.2	2009	http://guanine.evolbio.mpg.de/cgi-bin/kr2/kr.cgi.pl
MASH [75]	Number of word matches	v2.3	2016	https://github.com/marbl/Mash
SLOPE-SPAM [76]		0.1	2020	https://github.com/burkhard-morgenstern/Slope-SpaM
SKMER [77]		3.0.0	2019	https://github.com/shahab-sarmashghi/Skmer

TABLE A.1: A list of main alignment-free sequence comparison tools based on the summary from Zielezinski *et al* [6]. It lists the tools with their category, current version, released year and source. The tools marked with ⋄ are used in this thesis, NA = Not Available.

Description	Name used	GenBank ID	Source	Availability
Myxococcus fulvus 124B02, complete genome	M.fulvus-124B02	CP006003.1	nuccore	https://www.ncbi.nlm.nih.gov/nuccore/CP006003.1/
Myxococcus xanthus DK 1622, complete sequence	M.xanthus-DK-1622	13488	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_000012685.1/
Myxococcus xanthus DZ2 contig_000087, whole genome shotgun sequence	M.xanthus-DZ2-contig_000087	541118	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_000012685.1/
Myxococcus hansupus strain contaminant ex DSM 436 chromosome, complete genome	M.hansupus-strain-contaminant-ex-DSM-436	2031178	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_000280925.3/
Myxococcus stipitatus DSM 14675, complete sequence	M.stipitatus-DSM-14675	524338	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_000331735.1/
Myxococcus xanthus DZF1 contig_000001, whole genome shotgun sequence	M.xanthus-DZF1-contig_000001	540178	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_000340515.1/
Myxococcus xanthus strain GH3.5.6c2 chromosome, complete genome	M.xanthus-strain-GH3.5.6c2	11407168	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_006400955.1/
Myxococcus xanthus strain GH5.1.9c20 chromosome, complete genome	M.xanthus-strain-GH5.1.9c20	11407588	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_006401215.1/
Myxococcus xanthus strain KF3.2.8c11 chromosome, complete genome	M.xanthus-strain-KF3.2.8c11	11407968	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_006401635.1/
Myxococcus xanthus strain KF4.3.9c1 chromosome, complete genome	M.xanthus-strain-KF4.3.9c1	11408388	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_006402015.1/
Myxococcus xanthus strain MC3.3.5c16 chromosome, complete genome	M.xanthus-strain-MC3.3.5c16	11408758	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_006402415.1/
Myxococcus xanthus strain MC3.5.9c15 chromosome, complete genome	M.xanthus-strain-MC3.5.9c15	11408928	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_006402735.1/
Myxococcus sp. CA006, whole genome shotgun sequence	M.sp.-CA006	11571798	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_006518195.1/
Myxococcus sp. CA005, whole genome shotgun sequence	M.sp.-CA005	11571818	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_006518205.1/
Myxococcus sp. AB025B, whole genome shotgun sequence	M.sp.-AB025B	11571838	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_006518215.1/
Myxococcus sp. CA010, whole genome shotgun sequence	M.sp.-CA010	11603008	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_006547325.1/
Myxococcus sp. AB022, whole genome shotgun sequence	M.sp.-AB022	11603068	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_006547345.1/
Myxococcus sp. AB036A, whole genome shotgun sequence	M.sp.-AB036A	11603058	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_006547355.1/
Myxococcus sp. AB056, whole genome shotgun sequence	M.sp.-AB056	11603028	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_006547365.1/
Myxococcus llanfairpwllgwyngyllgogerychwyrndrobwllllantysiliogogogochensis strain AM401, whole genome shotgun sequence	M.the-longest-strain-AM401	11696838	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_006636215.1/
Myxococcus virescens strain NBRC 100334, whole genome shotgun sequence	M.virescens-strain-NBRC-100334	13181528	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_007989405.1/
Myxococcus fulvus strain NBRC 100333, whole genome shotgun sequence	M.fulvus-strain-NBRC-100333	13183968	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_007991095.1/
Myxococcus sp. CA027, whole genome shotgun sequence	M.sp.-CA027	16403738	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_010279825.1/
Pyxidicoccus caerfyrddinensis strain CA032A, whole genome shotgun sequence	P.caerfyrddinensis-strain-CA032A	17043638	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_010894405.1/
Pyxidicoccus trucidator strain CA060A, whole genome shotgun sequence	P.trucidator-strain-CA060A	17043658	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_010894435.1/
Myxococcus eversor strain AB053B, whole genome shotgun sequence	M.eversor-strain-AB053B	17043678	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_010894455.1/
Myxococcus vastator strain AM301, whole genome shotgun sequence	M.vastator-strain-AM301	17043698	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_010894475.1/
Myxococcus sp. CA023, whole genome shotgun sequence, whole genome shotgun sequence	M.sp.-CA023	17244108	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_010998615.1/
Myxococcus sp. CA018, whole genome shotgun sequence, whole genome shotgun sequence	M.sp.-CA018	17244168	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_010998655.1/
Pyxidicoccus fallax strain DSM 14698, whole genome shotgun sequence	P.fallax-strain-DSM-14698	19357928	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_012933655.1/
Myxococcus xanthus strain AB023, whole genome shotgun sequence	M.xanthus-strain-AB023	19357928	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_013116805.1/
Myxococcus xanthus strain AM003, whole genome shotgun sequence	M.xanthus-strain-AM003	19641398	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_013116825.1/
Myxococcus xanthus strain CA029, whole genome shotgun sequence	M.xanthus-strain-CA029	19641408	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_013116835.1/
Myxococcus xanthus strain AM005, whole genome shotgun sequence	M.xanthus-strain-AM005	19641428	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_013116865.1/
Pyxidicoccus fallax strain CA059B, whole genome shotgun sequence	P.fallax-strain-CA059B	19690678	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_013155555.1/
Myxococcus sp. CA033, whole genome shotgun sequence	M.sp.-CA033	20049168	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_013336625.1/
Myxococcus sp. CA039A, whole genome shotgun sequence	M.sp.-CA039A	20049178	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_013336645.1/
Myxococcus sp. CA051A, whole genome shotgun sequence	M.sp.-CA051A	20049198	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_013336705.1/
Myxococcus sp. CA056, whole genome shotgun sequence	M.sp.-CA056	20049238	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_013336715.1/
Myxococcus sp. CA040A, whole genome shotgun sequence	M.sp.-CA040A	20049228	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_013336725.1/
Myxococcus sp. AM010, whole genome shotgun sequence	M.sp.-AM010	20140668	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_013372585.1/
Myxococcus sp. AM011, whole genome shotgun sequence	M.sp.-AM011	20140658	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_013372595.1/
Myxococcus sp. AM009, whole genome shotgun sequence	M.sp.-AM009	20140688	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_013372605.1/

Description	Name used	GenBank ID	Source	Availability
Myxococcus xanthus strain DK1622::pDPO-Mxn116-Pvan-Tpase chromosome, complete genome	M.xanthus-strain-DK1622	23587308	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_015775755.1/
Myxococcus xanthus strain R31 chromosome, complete genome	M.xanthus-strain-R31	24648108	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_016698685.1/
Myxococcus sp. SCHIC003 chromosome	M.sp.-SCHIC003	25481178	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_017301635.1/
Pyxidicoccus sp. SCPEA002 chromosome	P.sp.-SCPEA002	25481278	assembly	https://www.ncbi.nlm.nih.gov/labs/gquery/all/?term=GCF_017301735.1_ASM1730173v1
Myxococcus xanthus DZ2 chromosome, complete genome	M.xanthus-DZ2-NZ_CP070500	26931718	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_018517205.1/
Myxococcus xanthus strain ATCC 27925 chromosome, complete genome	M.xanthus-strain-ATCC-27925	28743098	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_019895115.1/
Myxococcus sp. AS-1-15, whole genome shotgun sequence	M.sp.-AS-1-15	28915118	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_020037055.1/
Myxococcus sp. XM-1-1-1, whole genome shotgun sequence	M.sp.-XM-1-1-1	28915158	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_020037095.1/
Myxococcus sp. RHSTA-1-4, whole genome shotgun sequence	M.sp.-RHSTA-1-4	28915198	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_020037125.1/
Myxococcus xanthus DZ2 chromosome, complete genome	M.xanthus-DZ2-NZ_CP080538	30074838	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_020827275.1/
Myxococcus stipitatus strain CYD-1 Scaffold10_1, whole genome shotgun sequence	M.stipitatus-strain-CYD-1	30871788	assembly	https://www.ncbi.nlm.nih.gov/assembly/11729421
Myxococcus virescens strain DSM 2260, whole genome shotgun sequence	M.virescens-strain-DSM-2260	3637888	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_900101905.1/
Myxococcus xanthus strain DSM 16526, whole genome shotgun sequence	M.xanthus-strain-DSM-16526	3641658	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_900106535.1/
Myxococcus fulvus strain DSM 16525, whole genome shotgun sequence	M.fulvus-strain-DSM-16525	3696408	assembly	https://www.ncbi.nlm.nih.gov/assembly/GCF_900111765.1/
Myxococcus xanthus strain MCy8278 (Mxx48), draft genome	HIPS_M.xanthus-strain-MCy8278-Mxx48	NA	HIPS	NA
Myxococcus fulvus strain MCy8286 (Mxf65), draft genome	HIPS_M.fulvus-strain-MCy8286-Mxf65	NA	HIPS	NA
Myxococcus fulvus strain MCy8288 (Mxf50), complete genome	HIPS_M.fulvus-strain-MCy8288-Mxf50	NA	HIPS	NA
Pyxidicoccus fallax strain MCy8396 (And48), draft genome	HIPS_P.fallax-strain-MCy8396-And48	NA	HIPS	NA
Pyxidicoccus sp. strain MCy8408 (AndGT8), draft genome	HIPS_P.sp.-strain-MCy8408-AndGT8	NA	HIPS	NA
Myxococcus xanthus strain MCy8986 (DK897), draft genome	HIPS_M.xanthus-strain-MCy8986-DK897	NA	HIPS	NA
Myxococcus xanthus strain MCy8999 (MxA47), draft genome	HIPS_M.xanthus-strain-MCy8999-MxA47	NA	HIPS	NA
Myxococcus sp. strain MCy9171 (Mxbonn171), draft genome	HIPS_M.sp.-strain-MCy9171-Mxbonn171	NA	HIPS	NA
Myxococcus fulvus strain MCy9270 (SBMx122), draft genome	HIPS_M.fulvus-strain-MCy9270-SBMx122	NA	HIPS	NA
Myxococcus fulvus strain MCy9280 (SBMx132), draft genome	HIPS_M.fulvus-strain-MCy9280-SBMx132	NA	HIPS	NA
Pyxidicoccus sp. strain MCy9557 (SBCy002), complete genome	HIPS_P.sp.-strain-MCy9557-SBCy002	NA	HIPS	NA
Myxococcus sp. strain MCy10608 (MCy10608), complete genome	HIPS_M.sp.-strain-MCy10608-MCy10608	NA	HIPS	NA
Myxococcus virescens strain MCy11474 (ST200611), draft genome	HIPS_M.virescens-strain-MCy11474-ST200611	NA	HIPS	NA
Myxococcus sp. strain MCy11578 (SBMx268), draft genome	HIPS_M.sp.-strain-MCy11578-SBMx268	NA	HIPS	NA

TABLE A.2: A summary of the 71 strains within the **Curate Data** of this thesis. For each strain, the description, name used in the phylogenetic tree, Genbank ID, source and link are provided. nuccore = NCBI nucleotide database, assembly = NCBI assembly database, NA = Not Available, HIPS = Helmholtz Institute for Pharmaceutical Research Saarland.