Novel taxonomic profiling and benchmarking methods for high-resolution metagenomics

Joachim Fritscher
100317396/1

A thesis presented for the degree of Doctor of Philosophy

University of East Anglia
School of Biological Sciences
&
Quadram Institute Bioscience

June 2024

Abstract

The ever-increasing scale of available metagenomic data demands both fast and accurate tools. This comes at a time when assembly-based metagenomics substantially increased represented bacterial diversity in taxonomic databases and holds great potential for accurate and fast taxonomic profilers. Yet, current metagenomic profilers and strain-resolved tools do rarely utilise the vast known taxonomic diversity, nor do they leverage state-of-the-art approaches to provide both efficient and accurate metagenomic taxa profiles. With the importance of both species and strain-level analysis for gaining insights into the workings of microbial communities, there is a need to improve our understanding of how current tools work, and to improve the integration of available genetic resources using standardized and community supported databases.

This thesis presents approaches towards understanding the limitations of species- and strain-resolved metagenomic analysis and introduces two newly developed metagenomic profilers. Firstly, **benchpro** is a tool for in-depth benchmarking of taxonomic profilers using synthetic metagenomes. Benchpro disentangles the signal of false predictions by introducing a shared phylogenetic context between gold-standard profile and prediction. Second, **varkit** is a fast k-mer based taxonomic profiler that detects *de novo* SNPs based on k-mer match patterns relative to a taxonomic database. Lastly, **protal** is an alignment-based approach to taxonomic profiling and strain-resolved analysis and demonstrates how the integration of k-mer and alignment-based concepts can elevate accuracy and efficiency. Protal provides sensitive and accurate analysis at species-level while being 6.5 times faster than similar profilers. At strain-level, protal builds on top of a custom alignment algorithm and leverages this in a reference-guided multiple sequence alignment algorithm, achieving speed-ups of up to 55-fold.

Contents

Abstract i

Contents ii

Acknowledgements v

List of Acronyms and Abbreviations vii

List of Figures xxiv

List of Tables xxvi

1 Introduction 1

1.1 Problem statement 1

1.1.1 The complexity of metagenomics and taxonomic profiling . . 1

1.2 Thesis overview 1

2 Background 3

2.1 Microbial communities 3

2.1.1 Microbes, their Structure, and DNA 10

2.2 Genetic variation, Evolution, and Phylogeny 12

2.3 Describing Bacterial Taxonomy 15

2.3.1 Modern DNA-based Taxonomy 20

2.3.2 Of species and strains 22

2.4 How to study microbial communities? 23

2.4.1 Sequencing technologies 23

2.4.2 Read Errors 25

2.4.3 Approaches for sequencing complex microbial communities . . 26

2.5 Computational microbiome analysis 28

2.5.1 Taxonomic profiling 30

2.5.2 Expanding known diversity through Metagenomic Assembly . 34

2.5.3 Strain-level analysis 36

2.5.4 Long-reads and hybrid assembly improve MAGs 38

2.6 Computational concepts and data structures 39

3 Benchmarking of metagenomic profilers with benchpro 43
3.1 Introduction . . . 43
3.1.1 Contribution of this Chapter . . . 44
3.2 Methods . . . 45
3.2.1 Workflow . . . 45
3.2.2 Metrics . . . 46
3.2.3 Input Format for Benchpro . . . 51
3.2.4 Output Format . . . 52
3.2.5 Datasets . . . 53
3.2.6 Metagenomics . . . 54
3.2.7 Runtime and memory benchmark . . . 55
3.2.8 Implementation Details . . . 56
3.3 Results . . . 56
3.3.1 Overview . . . 56
3.3.2 Taxonomic Profiling . . . 58
3.3.3 Strain-level Benchmarks . . . 68
3.4 Discussion . . . 74
3.5 Author contributions . . . 77

4 Alignment-free taxonomic profiling and SNP detection with varkit 78
4.1 Introduction . . . 78
4.1.1 Contribution of this thesis . . . 79
4.2 Methods . . . 80
4.2.1 SNP calling with k-mer look up patterns . . . 80
4.2.2 Building the varkit reference database of hit-patterns . . . 80
4.2.3 Finding k-mer shapes for SNP calling . . . 81
4.2.4 Assessing varkit's SNP calling sensitivity . . . 84
4.2.5 K-mer data structure . . . 84
4.2.6 Building the database . . . 85
4.2.7 Taxonomic classification and SNP detection . . . 87
4.2.8 Implementation details . . . 88
4.3 Results . . . 89
4.3.1 SNP calling sensitivity . . . 89
4.3.2 Taxonomic profiling with varkit . . . 94
4.3.3 Runtime . . . 98
4.4 Discussion . . . 98
4.5 Author contributions . . . 100

5 Alignment-based taxonomic profiling and strain-resolved metagenomics using protal 101
5.1 Introduction . . . 101

5.2 Methods . . . 102
5.2.1 Workflow Approach . . . 102
5.2.2 Alignment . . . 103
5.2.3 Taxonomic Profiling using Random Forests . . . 108
5.2.4 Strain-level . . . 110
5.2.5 Additional Datasets and Benchmarks . . . 111
5.3 Results . . . 112
5.3.1 Overview . . . 112
5.3.2 Alignment evaluation . . . 114
5.3.3 Unique-k-mers and Random forest . . . 115
5.3.4 Species-level profiling . . . 118
5.3.5 Strain-level evaluation . . . 122
5.3.6 Runtime . . . 131
5.4 Discussion . . . 131
5.5 Author contributions . . . 134

6 Critical Assessment of Work **135**
6.1 Overview and chapter summaries . . . 135
6.1.1 Benchmarking of metagenomic profilers with benchpro . . . 135
6.1.2 Alignment-free taxonomic profiling and SNP detection with varkit . . . 136
6.1.3 Alignment-based taxonomic profiling and strain-resolved metagenomics with protal . . . 137
6.2 Limitations and assessment of results . . . 137
6.3 Further work . . . 138
6.3.1 Benchpro . . . 139
6.3.2 Varkit . . . 140
6.3.3 Protal . . . 140
6.3.4 Cross-mapping between closely related species . . . 141
6.4 Outlook . . . 142

Bibliography **145**

A Appendix **176**
A.1 Chapter 5 . . . 176
A.1.1 Benchpro . . . 176
A.1.2 Varkit . . . 212
A.1.3 Protal . . . 212

Acknowledgements

First, I want to thank my supervisor Falk Hildebrand for his support and countless productive discussions. His enthusiasm about science and trust in my abilities has been instrumental to my research. I want to thank Lindsay Hall and Rob Kingsley, who consistently provided perspective and support whenever I encountered obstacles during my PhD journey. I would also like to convey my gratitude to the Biotechnology and Biological Sciences Research Council and the UKRI-BBSRC Norwich Research Park Biosciences Doctoral Training Partnership. Their financial support has been essential to my research.

I would like to thank my colleagues, who over time have become great friends, for making my life as a PhD-student a joy and giving me fond memories to look back on. Ezgi Özkurt, thank you for your unwavering support; I cannot imagine my PhD journey without you. Thank you, Klara Cerk, for always being there, getting me back on track whenever I struggled, and lending an ear. Thank you, Ece Silan, for your fierce support, shared breaks, and the wonderful time we spent together at the conference in Edinburgh. Thank you, Rebecca Ansorge, for the fond memories of our shared time at ISME and for being a wonderful person and (former) desk neighbour. Thank you, Anthony Duncan, for always letting me interrupt you to show you cool plots. Thank you, Katarzyna Sidorczuk, for your calm presence and support. Thank you, Ángela del Castillo Izquierdo, for being a ball of joy and for being my colleague and friend during COVID. Thank you, David Schneider, for our badminton sessions and for bolstering the German contingent in our group. Thank you Wing Suet Koon for sharing your brain cell with me. Thank you, Raven Reynolds, for your support when I interviewed for the PhD position. Thank you, Falk Nagies, for taking us to play Magic the Gathering and for the great time in Belgium. Thank you, Oliver Charity, for saying the right things when I needed it, you are missed.

Thanks also go to everyone from my band at the Research Park, 'The Recs'. Thank you, Reuben, for being a great guitarist, friend, Alex Turner impersonator, and person. Thank you, Ryan, for your good energy, support, and energetic bass play. Thank you, Gary, it was a joy to perform with you. Thank you, Victor, for

your bass play, good taste in music, and for founding the band with me and Reuben. Thank you, Jay, for your drumming and your chill attitude. Thanks Charlotte, for you eclectic performance on 'Weird Bird'.

Thanks also go to my other friends, ones I made in the UK and from back home. Thank you, Moritz, for being my loyal friend and for supporting me whenever I needed it. Thank you, Esther, for being my one-man-army support system on the phone and for always being there. Thank you, Niels, for being there for me and for our countless discussions. Thank you, Sven, for supporting me and for sharing your home with me. Thank you, Tobi, for being you. Thank you, Dani, for being my friend. Thank you, Janos and Daisy, for making me feel at home when I first came to England. Thanks to everyone who has impacted my life here and made it such a joy. There are so many things I would like to express my gratitude for that there is no possible way to fit them all into this acknowledgement. I will miss you all!

Thank you Bella for being by my side from the start of my PhD to the very end. I could not have done this without you. No words can express how grateful I am to have you. Thank you, Evelyn, for being the best sister I could wish for. I'm grateful to you and Hannes for always having an open door for me and I can't wait to see the three riots grow into their own. Thank you, Dad, for believing in me and supporting me, always. I am coming home. To my mum: I think you would be proud of me. I miss you.

List of Acronyms and Abbreviations

AF	Alignment Fraction
ARG	Antibiotic Resistance Gene
BC	Bray-Curtis similarity
bp	base pairs
BWT	Burrows-Wheeler Transformation
CCS	Circular Consensus Sequencing
CLR	Continuous Long Sequencing
DDBJ	DNA Data Bank of Japan
DNA	Desoxyribonucleic Acid
EBI	European Bioinformatics Institute
FM-index	Full-text index in Minute space
FN	False Negative
FP	False Positive
GTDB	Genome Taxonomy DataBase
GWAS	Genome-wide Association Study
HGT	Horizontal Gene Transfer
HMP	Human Microbiome Project
IBD	Inflammatory Bowel Disease
ICBN	International Code of Botanical Nomenclature
ICBN	International Code of Nomenclature of Bacteria

ICBN	International Code of Nomenclature of Prokaryotes
indel	Insertion or Deletion
INSDC	International Nucleotide Sequence Database Collection
L2-TP	L2 similarity for True Positive abundances
LCA	Lowest Common Ancestor
LPSN	List of Prokaryotic names with Standing in Nomenclature
MAG	Metagenome Assembled Genome
MCE	Max Cluster Error
MetaHIT	METAgenomics of the Human Intestinal Tract
MGE	Mobile Genetic Element
mRNA	messenger RNA
MSA	Multiple Sequence Alignment
NCBI	National Center for Biotechnology Information
NGS	Next Generation Sequencing
ONT	Oxford Nanopore Technologies
OTU	Operational Taxonomic Unit
PacBio	Pacific Biosciences
PC	Pearson Correlation
PC-TP	Pearson Correlation for True Positive abundances
RED	Relative Evolutionary Divergence
RNA	Ribonucleic Acid
rRNA	ribosomal RNA
SAG	Single Amplified Genome
SCFAs	Short-chain fatty acids
SeqCode	Code of Nomenclature of Prokaryotes Described from Sequence Data
SIMD	Single Instruction, Multiple Data

SMRT	Single Molecule, Real-Time
SNP	Single Nucleotide Polymorphism
SV	Structural Variant
TP	True Positive
tRNA	transfer RNA
WFA	Wavefront Alignment

List of Figures

2.1 Schematic of different human microbiomes with common representatives. Image taken from Choudhry et al. under the license CC BY 4.0 [45] 6

2.2 Schematic drawing of a bacterial cell. All bacteria are surrounded by a cell membrane, delimiting the space into within the cell and the outside. Most bacteria have an additional cell wall. The inside of the cell is filled with the cytoplasm, containing different organelles such as the ribosomes. The chromosome, a single circular piece of DNA, encodes the genetic blueprint. Image is taken from Hiremath et al. from 2012 [233]. 11

2.3 This figure shows different structural variants and how to detect them with paired-end reads. RC refers to read count, RP to read-pair, SR to split-read and AS to assembly. **A** describes a deletion event, **B** an insertion, **C** an inversion and **D** a tandem repeat such as microsatellites. The figure is taken from Tattini et al. [273]. 13

2.4 Average nucleotide identity (ANI) between prokaryotic genomes of different taxonomic ranks. Image is taken from Hildebrand [95] . . . 23

2.5 Overview over different sequencers and sequencing technologies. Image taken from https://www.pacb.com/blog/the-evolution-of-dna-sequencing-tools/ 23

2.6 A, Schematic depiction of culture-dependent and culture-independent methods. Metagenomics emerged from the necessity to study microbial communities *in vivo* as opposed to *in vitro*, to study the microbe-microbe interactions as well as microbe-host interactions. B, Schematic of the two major metagenomic approaches. The assembly-based approach aims to reconstruct draft genomes of known and unknown microbes from sequencing data to capture both known and unknown microbial diversity. Reference-based methods rely on databases with reference genomes to profile metagenomes and analyse the known portion of microbial diversity. Assembly-based approaches are orders of magnitude more complex in terms of computational requirements as well as quality control. Image is taken from Yang et al. [313]. 29

2.7 Visualization how the number of computations for algorithms with different runtime complexities scales with input size. Image generated using R. 40

3.1 Benchpro's workflow listing required input, optional input, and output. Taxonomic profiling requires gold-standard and prediction profiles for each tool, dataset, and taxonomy. Optional files are phylogenetic tree for GTDB, and a list of all species in the database for each tool. File paths and optional data is configured in the META filer. For strain-level benchmarks, benchpro requires phylogenetic trees and MSAs for each tool and species across all datasets. Further, gold-standard phylogenetic trees and strain-resolved information for each dataset is required. 45

3.2 Tree visualization of TP, FP and FN of predicted and gold-standard profile. Each tip is a true positive, false positive, or false negative prediction. A, A genuine false negative, often because the abundance is below the detection threshold for the tool, but the taxon is covered by the database. B, A genuine false positive caused by false signals of nearby false positive predictions. C, A wrong false positive and false negative. The false negative taxon is not covered in the tool database, so its merged with its closest neighbouring false positive if the distance is under a threshold (distance of 0.04). 47

3.3 Here, the monophyly score describes the purity of a phyletic clade. It is computed by dividing the number of samples carrying strain A by the number of leaves in the smallest clade containing all samples carrying strain A. The monophyly score is constrained to be between 0 and 1, with one being perfect monophyly score. 49

3.4 MaxClusterError (MCE) compares pairwise phylogenetic distances within a strain to pairwise phylogenetic distances to other strains. High values of MCE indicate that two strains are not separated in the tree. Further, MCE quantifies large topological incongruities in the phylogenetic tree by reporting strains with high pairwise distances within its members. MaxClusterError (MCE) is a tree-dependent, subjective measure that reflects the relative placement of strains in a phylogenetic tree. While its absolute values may vary with tree topology, MCE is a useful comparative measure for assessing the clustering performance of different tools. When computing MCE for strain A, all other strains are considered for the pairwise between distances. . 50

3.5 Overview over datasets, tools, and analyses regarding taxonomic profiling in this chapter. 57

3.6 Overview over datasets, tools, and analyses regarding strain-level benchmarks in this chapter. 58

3.7 Species level profiling performance across samples, environments, and tools measured with F1-score, precision, and sensitivity. Each box plots represents a tool and each data point is a sample. Top to bottom, the panels display F1-Score, precision, and sensitivity and the column panels stratify between different environments with the left row being the summary across all environments. 59

3.8 Change of F1-score, precision, and sensitivity on species-level on the y-axis across tools when applying different abundance thresholds for the predicted profiles on the x-axis. Each panel shows the performance of one tool. The y-axis shows the mean of F1-score, precision, and sensitivity and is calculated for each tool over 107 samples across all datasets (CAMI Human, Mouse, Marine). 60

3.9 Genus level profiling performance across samples, environments, and tools measured with F1-score, precision, and sensitivity. Each boxplot represents a taxonomic profiler and each data point is a sample. From top to bottom, the row panels are F1-Score, precision, and sensitivity and the column panels stratify between different environments with the left column being the summary across all environments. 61

3.10 Benchmark to test the ability to correctly reconstruct microbial abundances. The column panels show different datasets, with the leftmost column being the summary across all; the row panels stratify different metrics. Each plot shows the performance on the y-axis stratified for all tools on the x-axis. Each data point in the boxplots represents a sample. The metrics displayed are Bray-Curtis similarity (BC), 1 - L2-error (L2) and Pearson Correlation (PC). The suffix '-TP' indicates that the metric has only been computed on TP abundances (see Section 3.2.2 for details). 62

3.11 Each point is a FP prediction, plotted with respect to the phylogenetically closest TP, FN- or FN+ in the same sample. FN+ are false negatives that are contained in the taxonomic database of the tool. FN- are absent from the tool database. The x-axis shows the tree distance to the closest TP and the y-axis shows the true abundance of the TP, FN+, and FN-. TP, FP, and FN values are after re-evaluation. 64

3.12 Profiling performance across samples, environments, and tools measured with F1-score, precision, and sensitivity after re-evaluating false predictions. The boxplots represent different tools and each data point is a sample. From top to bottom, the row panels are F1-Score, precision, and sensitivity and the column panels stratify between different environments with the left column being the summary across all environments. 66

3.13 Detection threshold of species based on abundances. The y-axis displays the gold standard abundance of each species (point) across all datasets (horizontal panels, UR=Urogenital, GI=Gastrointestinal, MA=Marine, MO=Mouse, OR=Oral, SK=Skin, AI=Airways) and the x-axis stratifies tools (color). The left panel shows FNs and TPs after adjustment and the right panel shows the unadjusted values. The top panel shows the FNs and the bottom panel shows the TP. High abundant FNs indicate failure to detect important species while low abundant TPs show that a tool has a high sensitivity for low-abundant species. FNs are filtered for species that are detectable by the respective tool. 67

3.14 Richness with respect to F1-score, precision, and sensitivity. Each dot represents one sample in the Mouse dataset and all statistics are after adjustment on species level. Richness on the x-axis is calculated as TP+FN and the y-axis shows the value for the respective statistic in each panel. 68

3.15 Strain and sample sensitivity of StrainPhlAn 4 on different species. **A** shows sample sensitivity plotted against strain sensitivity for each species. **B** also shows sample and strain sensitivity per species and includes the mean coverage across samples for this species. **C** resolves the gold-standard vertical coverage per sample in boxplots for each species, stratified by whether it is included in StrainPhlAn 4 phylogenies. Refer to Table A.2 for species abbreviations. 69

3.16 The monophyly score of StrainPhlAn 4 stratified across bacteria in panels. The monophyly score measures how pure clusters of samples carrying the same strain in a tree are (see Section 3.2.2). The rightmost column is a boxplot summarizing monophyly score across all species. Refer to Table A.2 for species abbreviations. 71

3.17 Correlation between sensitivity and monophyly scores per species. Each point is a species and the monophyly score measures how pure samples of the same strain cluster together. Sensitivity on the x-axis is how many strains and samples from the original dataset are still contained in the resulting phylogenetic trees. The correlation is computed with pearson and the regression line was computed according to a linear model. 72

3.18 Max Cluster Error (MCE) on the y-axis per strain and stratified by species. MCE is computed as explained in section 3.2.2 and quantifies how well all samples with a certain strain cluster together in the phylogenetic tree with respect to all other samples. The color encodes the number of strains with a positive MCE. 73

3.19 **A**, Error rate and alignment length of MSAs per species. Error rate is determined as number of multi-allelic positions per group of sequences representing the same strain considering positions with at minimum 2 informative bases (A,C,G,T) and alignment length is the number of bases in the alignment with at least 2 informative bases (see Section 3.2.2 for details). Each data point is a single strain. **B**, summary over alignment length, error rate and error count. **C**, error rate in context of monophyly per strain. There is no significant correlation. 74

3.20 Distance between the phylogenetic trees of StrainPhlAn 4 and randomly generated trees to the gold standard tree constructed with Roary for each species (n=42, four species not detected by MetaPhlAn 4+StrainPhlAn 4 due to lack of coverage in their database, see Section 3.2.5). Significance (***: $p \leq 0.001$) was calculated with a paired t-test, with species between protal and StrainPhlAn 4 as pairs. The utilized metrics are normalized Robinson-Fould distance (RFnorm), normalized weighted Robinson-Fould (wRFnorm), Steele and Penny distance (SP), weighted Steele and Penny distance (wSP), and Kuhnert-Felstenstein distance (KF) (see Chapter 3.2.2 for details). 75

4.1 Given a query and a k-mer shape (here: '`X_XX_X`') a hit-pattern is defined as a sequence of successful (1) or unsuccessful (0) k-mer lookups with respect to a reference. Position 5 (starting position being 0) in the query is considered mutated with respect to the reference (C→G) so all k-mers overlapping and with an 'X' at this position are unsuccessful lookups. A k-mer overlaps all positions where the k-mer shape has an 'X'. Gap positions denoted as '_' are ignored. 80

4.2 The workflow for building a pattern database with a given k-mer shape and pattern length (here: `X_XX_X` and 8). 82

4.3 The memory layout of varkit's hash map requires 64-bits per cell. The offset key (highlighted in orange in k-mer example) is a part of the key that is implicitly stored in the position such that all k-mers starting with the same prefix with predefined length lay in one contiguous block. Here, the keys in Cells 1-3 all start with the prefix "Offset key 1" and the keys in Cells 4-6 start with "Offset key 2". For each k-mer (offset key + internal key) varkit stores the taxonomic id (taxid), gene id (geneid) and gene position (genepos). 85

4.4 Space savings of varkit's hash map with 64 bits per cell + offset in extra data structure compared to a hash table with 96 bits without offset as baseline. With offset 32, the effective cell size is the same between both data structures. The labels denote the GB size at the break-even point for each offset, i.e. what the size of the map is when it is equally space efficient as the implementation without offset. . . 86

4.5 Basic workflow of varkit's classification and SNP detection algorithm. A, extract all k-mers from read based on the k-mer shape. Look up each k-mer in pre-built database, which contains k-mers from reference sequences. If the database has an exact matching hit, report taxid, geneid, and genepos, otherwise report a miss. B, hits are counted in the taxonomic (sub)tree and increment the count for that node by one. C, to classify the read, all hits are counted in the tree. The tree is then traversed from top to bottom (bottom is species-level) and the path is determined by the subtree with the higher total counts. D, hits and misses from the k-mer database lookup forms hit-pattern. If the classification result is on species-level, sub-patterns are looked up in a separate database to receive SNP positions between read and reference. Varkit reports the taxonomic classification as well as SNP positions for each read. 88

4.6 In 9623 iterations, each iteration tests the fitness of a new k-mer shape and then moves to the next (see Section 4.2.3). Fitness is defined as mean SNP sensitivity for ANIs between 95% and 99% (y-axis). The red dots are the three best performing shapes based on their fitness. From left to right these are 'XXXX_XX_X_XXX_X__XXXXX__X_XXX_X_XX_XXXX', 'XXXX_XX_XXXX__XX_X_X_X_XX__XXXX_XX_XXXX', and 'XXXX_XX_X_XXX_XX_X_X_X_XX_XXX_X_XX_XXXX'. The blue dots mark the lowest scoring k-mer shapes and from left to right these are 'XXXXXXXX_X_X_X_X_X_X_X_X_X_X_XXXXXXXX', 'XXXXXXXXX_____XXXXXXXXX_____XXXXXXXXX', and 'XXXX_XXXXX_X_X_X_X_X_X_X_X_X_XXXXX_XXXX'. 90

4.7 In 138 iterations, each iteration tests the fitness of a new k-mer shape and then moves to the next (see Section 4.2.3). Fitness is defined as mean SNP sensitivity for ANIs between 95% and 99% (y-axis). The red dots are the three best performing shapes based on their fitness. From left to right these are 'XXXXX__XX_XXXX_X_XXXX_XX__XXXXX' (mean ANI of 0.4875677), 'X_XXXXX_XXX__XXXXX__XXX_XXXXX_X' (mean ANI of 0.4904544), and 'XXXX_XXX_XX_XX_X_XX_XX_XXX_XXXX' (mean ANI of 0.4894113). The blue dots mark the lowest scoring k-mer shapes and from left to right these are 'XXX____XXXXXXXXXXXXXXXX____XXX' (mean ANI of 0.4298529),'XXXXX_XXXXX___XXX___XXXXX_XXXXX' (mean ANI of 0.4278185), and 'XXXX_X_X_X_XXXXXXXXX_X_X_X_XXXX' (mean ANI of 0.4213629). 91

4.8 Depiction of how three properties of the k-mer shape affect SNP calling sensitivity at ANI 95%. Panels show on the x-axis: number of gaps with distance three (gap_dists_of_3), number of unique pairwise gap distances in a shape (unique_gap_distances), and the sum of both values (sum). The y-axis shows the mean SNP sensitivity. Data is from shape finding with k=27 and s=12 (Fig. 4.6). 92

4.9 Comparison for training the pattern database with two different shapes 'XXXX_XX_XXX_X__X_XXXXX_X__X_XXX_XX_XXXX' (k=27, s=12) and 'X_XXXXX_XXX__XXXXX__XXX_XXXXX_X' (k=23, s=8) for two different pattern sizes 16 and 32. The left plot shows SNP-calling sensitivity for ANIs 90%, 95% and 99% with respect to the training depth 'limit' shown on the x-axis. The right plot shows how many patterns (y-axis) were added to the database with increasing training depth (x-axis). 93

4.10 SNP calling sensitivity at different ANI rates to reference, stratified by genomes. Random species-representative marker genes were selected and SNPs were introduced to match certain ANI values (94, 95, 96, 97, 98, 99% and vertical coverages 1,2,5,10). Reads were then simulated on the new references. Sensitivity is the number of SNPs detected divided by all inserted SNPs. GUT_GENOME227824 belongs to the species *s__Collinsella sp900761165*. The x-axis denotes the percentage of species-level k-mers for this species. see Section 4.2.4 for more details. 94

4.11 Profiling performance across samples, environments, and tools measured as F1-score, precision, and sensitivity. The boxplots represent different tools and each data point is a sample. From left to right, the column panels are F1-Score, precision, and sensitivity and the row panels stratify between different environments with the top row being the summary across all environments. 95

4.12 A, relative abundance for TPs for different tools coloured by dataset. Points on the lower end show the sensitivity of the tool to detect low abundant taxa. B, relative abundance for FNs for different tools, only showing taxa that are covered by the tool database. Points are coloured by dataset and higher end points show that a tool failed to detect higher abundant taxa. 96

4.13 A, false positive species in context of their closest TP or FN neighbour in the phylogenetic tree. FNs are split into detectable (FN+) and undetectable (FN-) based on whether the tool has this taxon in their database. B, Abundance prediction benchmark on species-level for all datasets. Metrics are Bray-Curtis Similarity (BC), 1-L2 error (L2), Pearson Correlation (PC), 1-L2 error only on TP taxa (L2-TP), and Pearson Correlation only on TP taxa (PC-TP). 97

4.14 Runtime and memory analysis of varkit with respect to other taxonomic profilers and strain-resolved tools. The benchmark was done on a single node with no interfering input and output using 16 cores. The tools were run on 10 samples from CAMI Airways with 2x 5GB uncompressed paired-end reads (see Section 3.2.7 for more details). . 99

5.1 Protal takes a set of paired-end short reads from shotgun metagenomic sequencing and outputs per sample taxonomic profiles and strain-resolved MSAs per species present in multiple samples. Internally, protal has three distinct steps - alignment, profiling, and strain-level. In the first step, all reads are aligned against all species-representative marker genomes within GTDB r214.0. During profiling, these alignments are processed and counted for each marker gene of each species. A random forest evaluates evidence for each species to predict presence or absence. To achieve strain-level resolution, alignments from the same species across multiple samples are used to yield a reference-guided alignment. 103

5.2 Concept of the flex-map. The flex-map serves as data structure for key-value pairs, with keys being core-mers and values being all positions in the reference for a core-mer and additional flex-mer information. For each core-mer stored in KEYS, there is a bucket in VALUES storing the location of all its occurrences in the reference. Those locations are stored in the 'Body' section of the bucket. Additionally, buckets of size > 1 have a 'Header' section storing the flex-mer for each core-mer. The flex-mer is the 2x8 flanking region of each core-mer. Given a k-mer with core-mer and flex-mer from the query, exact matching of the core-mer is used to retrieve the bucket with all its locations in the reference and the flex-mer is used to further filter the location based on flanking region similarity. 104

5.3 Detailed description of protal's alignment workflow. See Section 5.2.2 for more details. Information about unique k-mers is not mentioned here, as they do not influence the alignment process. 107

5.4 Overview over protal's results section and the presented analyses. The results section broadly divides into tool internal benchmarks, such as evaluating the correctness of alignments, taxonomic profiling benchmarks, and strain-level analysis benchmarks. 113

5.5 A, benchmark on 1,889,866,936 reads simulated from all bacterial marker genomes in GTDB r214 (n=80,789), comparing different aligners. TP and FP are evaluated based on whether a read was mapped to its correct species cluster representative. All reads that are neither TPs nor FPs are FNs. TP-rate and FP-rate are calculated relative to the total number of reads at all MAPQ filtering thresholds. B, runtime in minutes and memory in GB for all aligners on a dataset of 2x7.9GB uncompressed paired-end reads. Output is uncompressed for all tools. C, protal's internal time benchmarks on the dataset in B. Alignment takes by far the longest out of all stages of the alignment process. Stages refer to the following sections in Fig. 5.3: Seeding (1,2,3,4), Sorting Seeds (5), Pairing (6), Sorting Anchors (8), Anchor recovery (9), Alignment handler (10), Joining alignment pairs and sorting (11), Output handler (End). D, seed- and anchor-size frequencies in protal. Smaller seed sizes and anchor sizes with higher frequency result in a better runtime. 116

5.6 Variable importance reported from constructing the random forest. MeanDecreaseGini quantifies the importance of a variable within the random forest. Variable explanations can be found in Table 5.1. . . . 117

5.7 The number of unique k-mers (short uniques and long uniques) within the protal database stratified by species. A, unique k-mers across all species. B, unique k-mers only for species of the genera *'g__Bacteroides'*, *'g__Collinsella'*, *'g__Neisseria'*. The distribution of unique k-mers is not uniform across taxa, and the majority of *Collinsella* species have only few (mean±sd = 1812±4419) unique k-mers. Out of 502 *Collinsella* species in GTDB r214, 6 have less than 200 unique k-mers in protal and 86 have less than 200. This indicates a high similarity to other species or high diversity within the species. 117

5.8 Profiling performance across samples, environments, and tools. Performance is measured with the metrics F1-score, precision, and sensitivity (row panels). The boxplots represent different tools and each data point is a sample. The column panels stratify datasets. '200R' is the dataset MSSS200R. A, benchmarks on species-level adjusted. B, benchmarks on genus-level. 118

5.9 A, relative abundance for TP and FN+ (Taxa covered by the respective tool databases) for different tools coloured by tool. Horizontal panels stratify datasets (UR=Urogenital, GI=Gastrointestinal, MA=Marine, MO=Mouse, OR=Oral, SK=Skin, AI=Airways, RA=MSSS200R). Points on the lower end show the sensitivity of the tool to detect low abundant taxa. B, Richness with respect to F1-score, precision, and sensitivity. Each dot represents one sample in the Mouse dataset and all statistics are after adjustment on species level. Richness on the x-axis is calculated as TP+FN and the y-axis shows the value for the respective statistic in each panel. 120

5.10 A, false positive species in context of their closest TP or FN neighbor in the phylogenetic tree. FNs are split into detectable (FN+) and undetectable (FN-) based on whether the tool has this taxon in its database. B, Abundance prediction benchmark on species-level for all datasets. Metrics are Bray-Curtis Similarity (BC), 1-L2 error (L2), Pearson Correlation (PC), 1-L2 error only on TP taxa (L2-TP), and Pearson Correlation only on TP taxa (PC-TP). 121

5.11 A, Per species sensitivity, measured by how many samples and strains are present in the tree. B, protal per sample true vertical coverage stratified by species and colored by presence (blue) or absence (red). C, per species percentage of detected samples (y-axis) and mean true vertical coverage (x-axis). 123

5.12 Per strain monophyly score of protal and StrainPhlAn 4 stratified by species. Monophyly score measures how pure clusters of samples carrying the same strain in a tree are (see Section 3.2.2 for details). A, considering all tips. B, trees are subset to tips shared between protal and StrainPhlAn 4 per species. Species are ordered by ascending mean monophyly score. 124

5.13 Protal's read alignments for all marker genomes under *s__Clostridium_Q fessum* (A) and *s__Bacteroides ovatus* (B) against the database containing all representative marker genomes. Alignments are classified correct or incorrect based on whether they align to their respective species cluster representative. Alignments with a MAPQ value lower than 4 are filtered (protal default). C, Proportions of read alignments of all marker genomes under *s__Bacteroides ovatus* with respect to the species they aligned best to. This figure both quantifies how many reads are retained after filtering, and how many reads align to other species, potentially leading to FPs species detection and chimeric signal in MSAs (in this case for example for *s__Bacteroides xylanisolvens*). 125

5.14 Monophyly analysis with respect to closest neighboring strain. Each data-point is the mean monophyly of a sliding window of 10 samples per tool (y-axis) with samples sorted by closest pairwise similarity to neighboring strain (x-axis). Values on the x-axis are rank-transformed and label-placement is according to the closest matching point in the data. Mind that samples from the same strain are simulated with sequencing errors, but otherwise genetically identical. A, for each tree, all samples are included. B, for all trees only samples shared by both protal and StrainPhlAn 4 are considered. The fitted line is a loess regression. 126

5.15 Max Cluster Error (MCE) on the y-axis per strain and stratified by species. MCE is computed as explained in Section 3.2.2 and quantifies how well all samples with a certain strain cluster together in the phylogenetic tree with respect to all other samples. The color encodes the number of strains with a positive MCE. Negative MCEs (good clusters) are discarded. 128

5.16 A, Error rate and alignment length for each sample for protal's MSAs per species. B, alignment length, error rate, and total errors within MSAs of both protal and StrainPhlAn 4 of strains across all species. C, monophyly of protal's trees per strain in context of MSA error rate. Correlation computed with pearson correlation. See Table A.2 for species abbreviations. 129

5.17 A, distance between the trees of protal, StrainPhlAn 4, and randomly generated trees to the gold standard tree. All trees are subset to samples shared between protal and StrainPhlAn 4 and only trees for species that are predicted by both tools are considered (n=42). B, same as A, but protal and StrainPhlAn 4 trees are not subsets to shared samples. Significance was calculated with a paired t-test (*: $p \leq 0.05$, ***: $p \leq 0.001$, n=46 for protal, n=42 for StrainPhlAn 4), with species between protal and StrainPhlAn 4 as pairs. For B, species that are not shared were removed from the test. The utilized metrics are normalized Robinson-Fould distance (RFnorm), normalized weighted Robinson-Fould (wRFnorm), Steele and Penny distance (SP), weighted Steele and Penny distance (wSP), and Kuhnert-Felstenstein distance (KF) (see Chapter 3.2.2 for details). 130

5.18 Runtime and memory analysis of protal with respect to varkit, other taxonomic profilers, and strain-resolved tools. The benchmark was done on a single node with no interfering input and output using 16 cores. The tools were run on 10 samples from CAMI Airways with 2x 5GB uncompressed paired-end reads (see Section 3.2.7 for more details). 131

A.1 Each point is a FP prediction in Kraken2+bracken, plotted with respect to the phylogenetically closest TP, FN- or FN+ in the same sample (horizontal panels). FN+ are false negatives that are contained in the taxonomic database of the tool. FN- are absent from the tool database. The x-axis shows the tree distance to the closest TP and the y-axis shows the true abundance of the TP, FN+, and FN-. TP, FP, and FN values are after re-evaluation. Vertical panels stratify datasets. 232

A.2 Excerpt of phylogenetic tree as provided by GTDB r207, subset to TP, FP, and FN species as predicted by mOTUs3 GTDB on the dataset 3 of the CAMI Marine dataset. Red color indicates FPs, yellow indicates FN, and green indicates TPs. 234

A.3 Density plots showing the phylogenetic distances between undetectable false negative (FN) species—only those absent from the tool's database—and their closest false positive (FP) neighbours within the same CAMI sample (n = 107). Each panel represents a different profiling tool. The red vertical line marks the 0.04 cophenetic distance threshold used throughout this thesis to adjust binary classification metrics (F1 score, precision, sensitivity), accounting for missing taxa in tool databases. This threshold was selected to balance tolerance for taxonomic mismatches with avoiding overestimation of performance. 234

A.4 Species in the MSSS200C Dataset on the x-axis and their mean vertical coverage with standard deviation across all 200 samples on the y-axis. Each species is represented in each sample with exactly one strain. . 235

A.5 Each sample from the CAMI datasets (Human, Mouse, Marine) on the x-axis with all relative abundances in % displayed in a boxplot on the left y-axis. The red dots show species richness on the right y-axis. 235

A.6 In 138 iterations, each iteration tests the fitness of a new k-mer shape and then moves to the next (See 4.2.3). Fitness is defined as mean SNP sensitivity for ANIs between 95% and 99% (y-axis). The red dots are the three best performing shapes based on their fitness. From left to right these are 'X_XXXXX_XXX__XXXXX__XXX_XXXXX_X' (mean ANI of 0.7652134), 'XXXX_XX_XXXX__XXX__XXXX_XX_XXXX' (mean ANI of 0.7619734), and 'XXXX_XXX_XX_XX_X_XX_XX_XXX_XXXX' (mean ANI of 0.7685434). The blue dots mark the lowest scoring k-mer shapes and from left to right these are 'XXX____XXXXXXXXXXXXXXXX____XXX' (mean ANI of 0.6922234),'XXXXX__XX__XXXXXXXXX__XX__XXXXX' (mean ANI of 0.6972566), and 'XXXX_X_X_X_XXXXXXXXX_X_X_X_XXXX' (mean ANI of 0.6849034). 237

A.7 Varkit species prediction profile for sample Oral 13 in the context of the phylogenetic tree of GTDB r207. Red tips are FPs, green tips are TPs, and yellow tips are FNs. The bottom right cluster of species under the genus *g__Streptococcus* are FPs likely wrong hits from the TP *s__Streptococcus suis* 238

A.8 Profiling performance with respect to different abundance threshold filtering. 238

A.9 Subtree of GTDB r214 species tree containing all *s__Collinsella* species from dataset MSSS200R_15. 239

A.10 Subtree of GTDB r214 species tree containing all *s__Collinsella* species within GTDB r214 and a subset of GTDB r207 containing only species covered by MetaPhlAn 4. 239

A.11 Subtree of GTDB r214 species tree containing all *s__Collinsella* species within GTDB r214 and a subset of GTDB r207 containing only species covered by MetaPhlAn 4. 240

A.12 Read alignments with protal for reads simulated from all marker genomes mapping outside of their original species cluster and filtering out alignments with MAPQ <4 (see Section 5.2.5 for details about data). All 80,789 are stratified on the x-axis and sorted based on the percentage of correct alignments. 240

A.13 Read alignments with protal for reads simulated from all marker genomes mapping outside of their original species cluster and filtering out alignments with MAPQ <4 (see Section 5.2.5 for details about data). Species contained in MSSS200 are stratified on the x-axis and (as opposed to Fig. A.14) incoming alignments are stratified based on whether they originate from the species, or not. Incorrect alignments for a species hence quantifies the amount of alignments from reads of other species. 241

A.14 Read alignments with protal for reads simulated from all marker genomes mapping outside of their original species cluster and filtering out alignments with MAPQ <4 (see Section 5.2.5 for details about data). Species contained in MSSS200 are stratified on the x-axis and sorted based on the percentage of correct alignments. 241

A.15 A, Increase in monophyly score for protal from before to after filtering to the same samples as StrainPhlAn 4. B, Increase in monophyly score with respect to the fraction of retained strains from before to after filtering. There is a significant correlation between strain loss and increase in mean monophyly score (p=0.016, pearson correlation) 242

A.16 Phylogenetic trees for *s_ _Neisseria gonorrhoeae*. A, phylogenetic tree constructed from protal's MSA. B, phylogenetic tree constructed from protal's MSA, subset to shared trees with StrainPhlAn 4. C, phylogenetic tree constructed from StrainPhlAn4's MSA. 242

List of Tables

2.2 An overview of selected tools for taxonomic profiling. Fixed pre-built under **database** means a database is provided with the tool. Custom means the user can provide their own set of genomes as reference. The tools in the Table are MEGAN [105], mothur [239], QIIME [40], MetaPhlAn [246], mOTUs [272], UPARSE [62], Kraken [308], LotuS [96], CLARK [200], MetaPhlAn 2 [277], Kaiju [173], Centrifuge [123], DADA2 [39], Bracken [159], KrakenUniq [32], Kraken2 [307], mOTUs2 [179], MetaPhlAn 3 [19], mOTUs3 [232], MetaPhlAn 4 [26]. 32

3.1 Map file required for benchmarking taxonomic profiling. All columns are mandatory. 52

3.2 Benchpro map file for strain-level evaluation. Rows are samples and columns contain additional information on file paths and others. The meta-file can be either a .xlsx or as tab-delimited text file. Provided to benchpro with *'–meta META_FILE'* 53

3.3 Benchpro species meta-file for strain-level profiling. Rows are samples and columns contain additional information on file paths and others. The meta-file can be either a .xlsx or a tab-delimited text file. Provided to benchpro with *'–meta META_FILE'* 53

5.1 The following metrics are computed by protal for each species from all read alignments and are used as input for the random forest. . . . 109

A.1 Number of species for utilized benchmark datasets CAMI Human, CAMI Mouse, and CAMI Human for NCBI, GTDB r207, and GTDB r214. 176

A.4 All (isolate) genomes from humgut that are part of the strain-analysis dataset and their GTDB r214 species annotations generated with GTDB-tk 2.32. 179

A.6 All genomes that are part of the MSSS200R dataset and their GTDB r214 species annotations generated with GTDB-tk 2.32. Genomes were downloaded from MGnify Genomes v2.0 (ftp://ftp.ebi.ac.uk/pub/databases/metagenomics/mgnify_genomes/human-gut/v2.0/). 212
A.2 Species and their abbreviations used throughout the thesis. 233
A.3 Species in the MSSS200C Dataset and their mean vertical coverage with standard deviation across all 200 samples. Each species is represented in each sample with exactly one strain. 236
A.5 Per sample mean vertical coverage (VC) and standard deviation (SD) for MSSS200R, as well as number of species. 243

Chapter 1

Introduction

1.1 Problem statement

1.1.1 The complexity of metagenomics and taxonomic profiling

The vast amounts of produced metagenomic sequencing data is responsible for major insights in how bacteria interact with each other or their environment [204], how they associate with health and disease in humans[2], and how they colonize the human host [98]. Reference-based taxonomic profiling is an integral method for reconstructing taxonomic profiles from complex metagenomes. Further, the shift towards *de novo* methods for assembling draft genomes led to an increase in diversity represented in taxonomic databases which massively benefits reference-based methods [26, 232] However, tools vary in performance and do not always seamlessly integrate with standard taxonomies and different taxonomies and databases can have great impact on profiling results [26]. While independent benchmarks of taxonomic profiling exist, they do not include the latest tools and further do not distinguish between the effect of tool performance and uncertainty introduced by the utilized taxonomy [242, 176, 177]. While metagenomic profilers often use the NCBI taxonomy, GTDB has established itself as an alternative to NCBI in the recent years, but is often still poorly supported by the most used tools, and hence hampers integration to other tools and research using GTDB. Further, the currently most accurate tools are significantly slower than the fastest[307, 26, 232], and both have only improved little regarding speed in the past few years [277, 26]. On strain-level, this is even more pronounced as the fastest tools that are able to quantify within-species diversity do not scale linearly with the number of samples.

1.2 Thesis overview

This section provides a brief overview of the thesis. As a means to to cope with the vast amount of data in metagenomics, the aim of this thesis was to improve the speed and accuracy of taxonomic profilers and strain-resolved tools. This was achieved by

implementing both existing and novel concepts around k-mer and alignment-based methods, as well as in-depth benchmarking of existing tools. Chapter 2 introduces the importance of microbiology research, and the shift towards metagenomics caused by the advent of sequencing technologies. Core concepts around bacterial structure, genetic blueprint, genetic diversity, as well as bacterial taxonomy, phylogeny and the notion of species and strains are explained to provide the foundation for metagenomics. This is followed by a short overview of sequencing technologies and a brief summary of bioinformatics tool and their concepts, focusing on metagenomics. Further, the difference between algorithmic improvements and implementation details is demonstrated with the example of alignment algorithms.

Chapter 3 introduces benchpro, a software for benchmarking species-level and strain-level tools on simulated datasets. In this chapter, first details about the workflow and benchmarking metrics are introduced, followed by a comparison between different taxonomic profilers on species-level on various datasets and benchmarking of one strain-resolved tool.

Chapter 4 presents varkit, a k-mer based taxonomic profiler with a novel concept to detect SNPs by using k-mer match patterns between query and database. First, the methodology of k-mer based SNP calling is introduced. This is followed by results about SNP calling sensitivity and a benchmark on species-level using benchpro.

Chapter 5 introduces protal, an alignment-based taxonomic profiler leveraging both unique and inexact matching k-mers, an integrated alignment approach using a novel data-structure, and a reference-guided approach to MSAs. This chapter first introduces the workflow and then explains building the index from GTDB sequences, the alignment algorithm, taxonomic profiling using random forests and reference-guided MSAs to proceed to strain-level.

Chapter 2

Background

2.1 Microbial communities

Microbes prosper in a multitude of environments from parts of the human body such as skin, gut or the oral cavity, over soil to even the most hostile places on earth such as black smokers with temperatures above 250°C [17]. These microbes do not exist in isolation; rather, they form complex communities known as microbiota. The microbiota in the context of its environment including both biotic and abiotic factors is called the microbiome.

Microorganisms, or microbes, are microscopically small single-celled organisms living in solitude or as part of larger colonies of cells. While microbes encompass all domains of life and their definition includes all archaea, bacteria, and some eukaryotes, the focus of this thesis will be solely on bacteria. Eukaryotic microbes are generally less abundant in most environments, making them more challenging to study and leading to their under-representation in genome databases. While archaea can occur at higher abundances in specific environments, their representation in databases still lags behind bacteria due to challenges such as their inherent difficulty in in-vitro cultivation and the resulting lower quality and quantity of reference genomes. Additionally, archaea present unique computational challenges, including taxonomic complexity, distinct genomic features, and limited benchmarking resources, which complicate their integration into profiling tools. Since the computational tools developed and presented in this thesis depend on genomic reference sequences for microbiota analysis, the focus was placed on bacteria, which benefit from both a higher quantity and better quality of reference sequences. Although GTDB, the primary resource of genomes utilised in this thesis, includes archaeal references and genome markers, extending the tools to also profile archaea was beyond the scope of this thesis.

Microbes were first described by Antonie van Leeuwenhoek in 1674, as what he referred to as 'little animalcules' [135]. Driven by curiosity, he crafted the first single-lensed microscope, which allowed him to unravel and describe the fascinating

world of microscopic organisms. To this day, van Leeuwenhoek is referred to as the father of microbiology and microscopy and his pioneering work set the foundation for modern microbiology. Microbial research initially focused on pathogenic microbes causing sickness affecting large parts of the population which lead to prominent bacterial pathogens such as *Mycobacterium tuberculosis* or *Bacillus anthracis*, causing the deadly diseases tuberculosis and anthrax, already being discovered and described in the 19th century. This also explains the importance of the discovery of penicillin in 1928, which marked the beginning of the golden age of antibiotics and finally provided a means to combat bacterial infections that had previously claimed thousands of lives [73]. Yet, until the second half of the 20th century, research on microbes was limited to observational methods, neglecting the underlying genetic components and their intricate evolutionary relationships. This changed in the second half of the 20th century with the invention of DNA-DNA-hybridization and Sanger sequencing [236, 92], facilitating analysis of the underlying blueprint of microbes, ultimately defining traits that were previously only observable. Like all living organisms, microbes have a genetic blueprint, their DNA, which encodes genes that define all of their observable traits and is inherited from generation to generation.

Before the advent of DNA-based techniques like Sanger sequencing, microbiome research relied heavily on culturing-based approaches, where microbes were grown in the laboratory on selective media. Scientists would isolate and characterise organisms based on their morphology, metabolic activities, and staining properties. However, these methods were inherently limited, as the vast majority of microbial species are not easily cultivable under standard laboratory conditions. As a result, early studies could only capture a small, biased fraction of the true microbial diversity present in a given environment [7].

Although Sanger sequencing was a groundbreaking innovation, its high cost and low throughput initially limited its application in large-scale studies of microbial communities, such as those found in environmental or host-associated microbiota. Over time, advances in sequencing technology increased both throughput and automation. A key innovation was the replacement of radioactive-labelled nucleotides with fluorescent-labelled dideoxynucleotides, which enabled safer handling and the simultaneous detection of all four DNA bases in a single reaction [218] allowing for automating sequencing. In 1987, Applied Biosystems introduced the first automated DNA sequencer, which used gel electrophoresis coupled with fluorescent dyes to detect and differentiate the four DNA bases. The subsequent replacement of gel-based systems with capillary array electrophoresis further improved the speed, resolution, and scalability of Sanger sequencing, setting the stage for the next generation sequencing (NGS) machines.

Researchers like Wilson & Blitchington [302], Suau et al. [270], and Bonnet &

Collins [30] were pioneers in utilizing these methods to explore the composition of complex microbial communities, with a particular focus on the human gut. These studies revealed that traditional culture-based methods captured only a small fraction of microbial diversity, highlighting the presence of a vast 'unculturable majority'. By sequencing cloned 16S rRNA genes from PCR-amplified community DNA, these researchers were among the first to use molecular techniques to identify novel microbial taxa and to begin estimating the immense richness of microbiotas in situ.

The introduction of the 454/Roche pyrosequencing platform in 2005—the first widely adopted next-generation sequencing (NGS) technology—marked a major leap over traditional Sanger sequencing by enabling larger-scale, parallel sequencing without the need for electrophoresis [168, 231]. Compared to Sanger methods, 454 offered much higher throughput and longer read lengths, making it particularly attractive for early microbial diversity studies. However, despite these advances, 454 sequencing remained limited by relatively high costs, low scalability, and susceptibility to homopolymer errors, which impaired sequence accuracy in genomes rich in repetitive regions. Consequently, microbiome studies using 454 were typically restricted to sequencing marker genes such as the 16S rRNA gene rather than full metagenomes (see 2.4.3 for more details on amplicon sequencing). Briefly, while amplicon sequencing enables taxonomic stratification typically up to the genus level, whole-genome sequencing is required to achieve species- or strain-level resolution or to analyse functional potential (see 2.3 for details on bacterial taxonomy).

The subsequent development of Illumina sequencing further transformed the field by offering vastly higher throughput, lower cost, and greater sequencing accuracy. Although the initial read lengths were shorter than those of 454, the sheer volume of data combined with improved base-calling precision enabled much deeper profiling of microbial communities and helped rendering whole-metagenome shotgun sequencing feasible at scale. This technological shift revolutionized microbiology research by allowing comprehensive analyses of microbial composition, function, and ecology across diverse environments and hosts.

It is important to note that whole-genome sequencing of microbiomes only gained traction from around 2010 [174]; prior to that, sequencing efforts typically focused on the more accessible and cost-effective to sequence 16S rRNA gene, a phylogenetic marker widely used for taxonomic identification (see 2.4.3 for details on amplicon sequencing). Whole-genome sequencing of microbiomes only became a routine practice around 2015. This was primarily due to earlier limitations in cost and sequencing technology, as well as a lack of computing power and adequate and robust tools to process the data [124].

As for the human gut microbiome, an awareness and understanding of its role

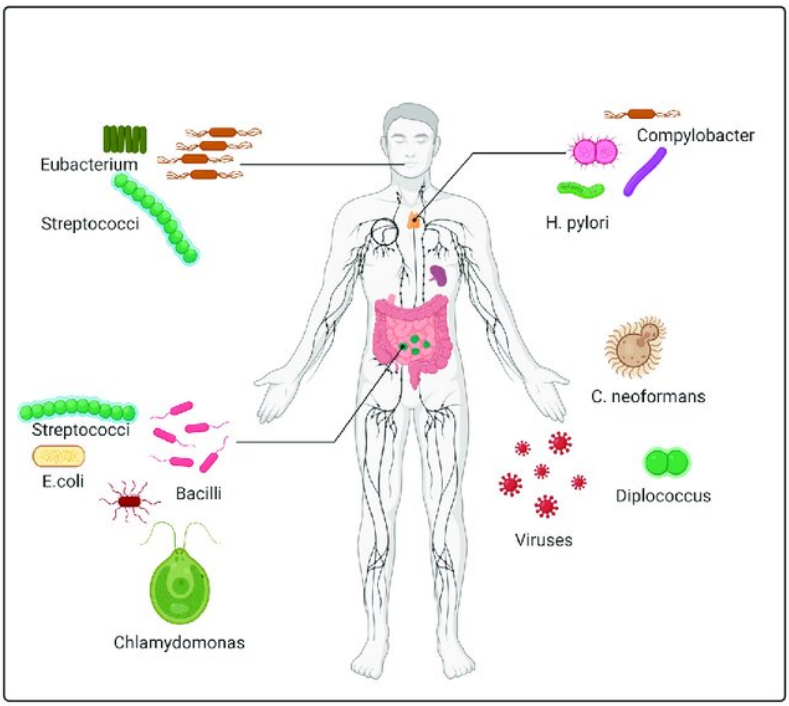

Figure 2.1: Schematic of different human microbiomes with common representatives. Image taken from Choudhry et al. under the license CC BY 4.0 [45]

in human health is still emerging. Complex bacterial communities harbour almost every part within and on our bodies and play important roles both in maintaining physiological homeostasis and in influencing disease processes (Fig. 2.1). These microbiomes contribute to digestion, modulate the immune system, and protect against pathogenic organisms, emphasizing their integral role in human health and disease.

For instance, in the gut, microbiota contribute to digestion by fermenting complex carbohydrates that are otherwise indigestible by human enzymes, producing short-chain fatty acids (SCFAs) like butyrate, acetate, and propionate, which serve as important energy sources and have anti-inflammatory effects [74]. Specific taxa such as the genus Bacteroides and various members of the phylum Firmicutes are key players in this process [158]. Moreover, the microbiome modulates the immune system by interacting with intestinal epithelial cells and immune cells; for example, segmented filamentous bacteria have been shown to promote the maturation of Th17 cells, a critical subset of T helper cells involved in mucosal immunity [109]. Additionally, commensal microbes competitively inhibit colonization by pathogenic organisms by occupying niches and producing antimicrobial compounds, a phenomenon known as colonization resistance [38].

Large-scale sequencing of bacterial communities as part of international efforts like the Human Microbiome Project (HMP) or MetaHIT helped to shed light on a previously understudied field. By gathering and analysing human fecal (or other bodysites) microbial sequencing data of thousands of individuals [63, 106] researchers

have gained insights on the human gut microbiome in health and disease.

The Human Microbiome Project (HMP) made foundational contributions to our current understanding of microbiotas. Initiated in 2007, the HMP systematically mapped the microbial diversity across different human body sites, including the gut, skin, oral cavity, and urogenital tract [280]. By employing standardized protocols for sampling, sequencing, and analysis, the project created a comprehensive reference framework for what constitutes a "normal" microbiota in healthy individuals. This effort revealed the vast inter-individual variability in microbial composition, established the concept of body site-specific microbial communities, and emphasized the functional redundancy among different microbial taxa. Furthermore, the HMP provided critical insights into the early links between microbiota alterations and human diseases, thereby catalysing a major paradigm shift in biomedical research that recognized the microbiome as a key determinant of human health. Building upon the foundational work of the HMP, the MetaHIT project extended the scope of microbiome research by focusing on the functional potential and genetic richness of the gut microbiota.

The MetaHIT (Metagenomics of the Human Intestinal Tract) project, launched in 2008 in Europe, significantly advanced our understanding of the human gut microbiota through the application of whole-genome shotgun sequencing [174]. By generating one of the first comprehensive gene catalogues of the human gut microbiome, MetaHIT demonstrated that the gut microbiota harbours an immense genetic diversity far exceeding that of the human host. The project introduced critical concepts such as microbial gene richness and the existence of a core set of microbial genes shared across individuals. Moreover, MetaHIT studies revealed major associations between microbial gene content and human health, particularly highlighting the reduced gene richness observed in individuals with obesity, type 2 diabetes, and inflammatory bowel diseases. By shifting the focus from merely cataloguing species to understanding microbial functional potential, MetaHIT laid the groundwork for functional microbiome research and established a new paradigm for studying host-microbe interactions at the genomic level. The MetaHIT consortium introduced key concepts such as the existence of a "core microbiota," referring to microbial species (and features) commonly shared across individuals, and "microbial gene richness", which describes the diversity of microbial genes within the gut microbiome [174, 11].

Both the HMP and MetaHIT made their datasets openly available to the scientific community, providing invaluable resources that continue to support microbiome research, method development, and comparative studies worldwide.

A key insight into the gut microbiome that emerged was that each hosts' microbiome is unique in regards to strain composition, diversity and abundance. Sheer numbers can help us grasp the complexity: the human gut microbiome contains $10^{13} - 10^{14}$ microbial cells, which is the same order of magnitude as cells that exist in the entire body with 3×10^{13} [247], and distributes across an estimated 250-400

species of bacteria in the human gut[157].

The increased research focus has led to the discovery of links between the gut microbiome and host diseases, such as obesity [15, 156, 107, 294], IBD (Inflammatory Bowel Disease) [165, 139, 121, 41, 1, 163], Parkinson's disease [230, 18], cardiovascular diseases [60, 188, 90, 75], type 2 diabetes [252], cancer [83, 47, 170], depression [119], and rheumatoid arthritis [320].

Many of these diseases are mediated by the immune system and characterised by a state of chronic low-grade inflammation, which is believed to be influenced by gut microbiome composition and function [292, 3]. Research into the microbiomes of metabolically healthy versus unhealthy individuals has revealed characteristic differences, including reduced microbial diversity, lower abundance of SCFA-producing bacteria (such as *Faecalibacterium* and *Roseburia*), and increased prevalence of pathobionts in unhealthy states.

Furthermore, the gut microbiome plays a critical role in the maturation of the immune system, shaping immune responses through mechanisms such as the induction of regulatory T cells by *Clostridia* species, the promotion of IgA production for mucosal immunity, and the development of gut-associated lymphoid tissue (GALT) [324]. These interactions help establish immune tolerance to commensal microbes while ensuring robust responses to pathogens [87]. For example, *Bacteroides fragilis* facilitates immune system training through the production of polysaccharide A, which is essential for maintaining immune homeostasis [**mazmanian_immunomodulatory_2005**]. Immune homeostasis refers to the balanced state in which the immune system can effectively defend the host against infections while avoiding excessive or inappropriate immune responses that could lead to chronic inflammation or autoimmune diseases.

Microbiome research does not solely focus on diseases—there is a continuous effort to characterise and define a 'healthy microbiome' by analysing the microbiomes of metabolically healthy individuals [248, 166]. Through this, it has become clear that the microbiome is shaped not only by pathological conditions in the host but also by factors such as dietary changes, drug intake, and environmental influences [260]. Although the term "healthy microbiome" is frequently used, including in marketing contexts, defining it scientifically remains challenging [114]. Efforts to delineate a healthy microbiome are complicated by factors such as high interpersonal variability, the influence of geography, diet, age, and genetics, as well as the absence of a single universal microbial composition that could be considered 'optimal' across different populations [114].

Dysbiosis refers to quantitative and qualitative disease-related changes in the gut microbiome, including alterations in metabolic activity, microbial diversity, and the composition of beneficial and pathogenic microbes [27, 56]. The term 'dysbiosis' is, however, subject to criticism, as it suggests the existence of a known and definable bacterial composition that causes disease. More accurately, dysbiosis describes

a disrupted microbiome associated with disease, where the microbial changes are hypothesized to be at least involved in the disease process.

'Dysbiosis' refers to quantitative and qualitative disease-related change patterns in the gut microbiome, describing changes in metabolic activity, microbial diversity, and microbial composition related to beneficial and pathogenic microbes [27, 56]. However, the concept of dysbiosis has been criticized, as it implies the existence of a clearly defined, "healthy" microbial composition whose disruption leads to disease. In reality, dysbiosis is more accurately understood as a descriptive term for microbial alterations observed in the context of disease, where the microbiome is hypothesized to modulate disease severity or contribute to disease processes, rather than necessarily being the direct cause [194].

In order to understand the impact of the gut microbiome on host health, it is essential to understand the functional role and key interaction points between the host and its microbial community.

A dysbiotic microbiome can lead to a loss of regulatory immune effects on the gut mucosa, associated with inflammatory and immune-mediated diseases such as rheumatic arthritis, metabolic syndrome, neurodegenerative disorder and malignancy as well as increased susceptibility to pathogen invasion or commensal-to-pathogen transitioning [101, 165, 324, 225]. Understanding the already complex host-microbiome interaction is further confounded by the constant change in gut microbiome composition.

Both age and diet are among the most influential factors driving shifts in gut microbiome composition throughout life. During infancy, the gut microbiota is initially shaped by mode of delivery and early feeding practices (breastfeeding vs. formula feeding), and gradually diversifies post-weaning [**backhed_dynamic_2015**]. In adulthood, dietary patterns such as high fiber or high fat intake significantly modulate microbial diversity and metabolic output [310]. Aging is associated with a decrease in microbial diversity and a shift towards a more inflammatory microbial profile [16], which may contribute to immunosenescence and increased disease susceptibility in the elderly [51]. Thus, both age-related and diet-induced changes represent important variables that must be considered when studying host-microbiome interactions.

Although considered robust to large-scale perturbations, certain events can cause a dramatic change in the gut microbiome. Hildebrand et al. have shown that an antibiotic intervention can cause low-abundant species to bloom to monodominance and result in a persistent shift in the microbiome [97]. In addition to antibiotics, a wide variety of external factors such as environment, diet, medication, and geographical region, impact the microbiome and, by extension, influence the human host. Further, host genetics play a significant role in shaping the gut microbiome by influencing factors such as immune system function, mucosal barrier properties, and the production of antimicrobial peptides, thereby affecting which microbial taxa are

able to colonize and persist [85, 261].

An effort to categorize and simplify the gut microbiome towards better understanding and analysis resulted in the concept of enterotypes, distinct generalized configurations of taxonomic compositions, introduced by Arumugam et al. in 2011 [11]. Three key configurations, mainly based on relative abundances of the genera *Bacteroides* (Phylum: *Bacteroidetes*), *Prevotella* (Phylum: *Bacteroidetes*) and *Ruminococcus* (Phylum: *Firmicutes*), were reported to capture different states of composition. This led to the assumption that a limited number of well-balanced host-microbial symbiotic states exist.

However, subsequent research has challenged and refined the initial three-enterotype model [100, 67, 287]. Later analyses showed evidence for a fourthenterotype structure, distinguishing between two subgroups within the Bacteroides-driven enterotypes — Bacteroides 1 and Bacteroides 2 — alongside Ruminococcaceae and Prevotella enterotypes. This expanded model captures greater interindividual variability and suggests that the microbiome configurations are more nuanced than originally proposed. A recent study further supports this refinement by introducing the concept of enterosignatures, emphasizing the dynamic and gradual nature of microbiome community structures rather than strictly discrete clusters [76]. Additionally, recent findings highlight continued efforts to revisit and update the enterotype framework in light of new large-scale datasets [120].

The heightened focus on microbiome research has been significantly driven by advancements in sequencing technologies, along with subsequent computational developments leading to the numerous mentioned discoveries. However, before exploring the technical and computational aspects of modern microbiology research, it is essential to understand the fundamental structure of bacteria, their genetic blueprint, and their taxonomic classification.

2.1.1 Microbes, their Structure, and DNA

Bacteria are single-celled microorganisms belonging to the prokaryotic domain. All bacteria are enclosed by a cell membrane, mostly also by a cell wall, and have a single loop of DNA at their center, called the chromosome (Fig. 2.2). However, there are notable exceptions, such as *Streptomyces coelicolor*, which possesses a linear chromosome rather than the more common circular bacterial genome [23]. In contrast to eukaryotes, the chromosome is not enclosed by a nucleus and instead exposed in the cytoplasm, and prokaryotes also lack membrane-bound organelles. Most bacteria reproduce asexually through a process called binary fission, wherein the bacterial cell divides into two equal halves, each of which contains a complete copy of the genetic material, or DNA. *Streptomyces coelicolor*, again, poses an exception as it reproduces through the formation of aerial hyphae—filamentous structures formed by certain

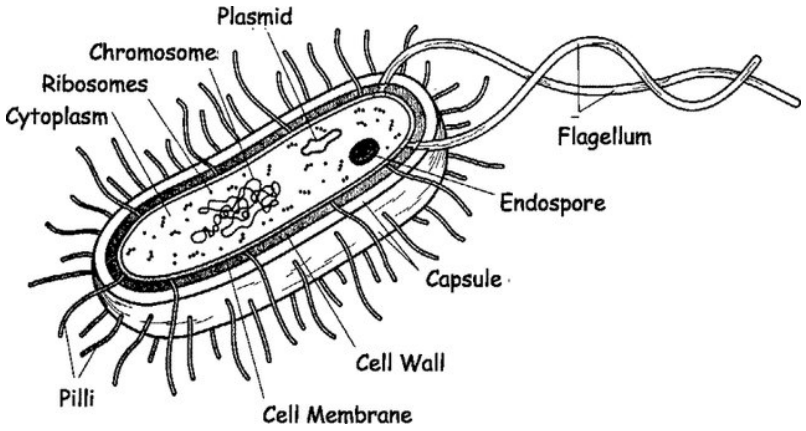

Figure 2.2: Schematic drawing of a bacterial cell. All bacteria are surrounded by a cell membrane, delimiting the space into within the cell and the outside. Most bacteria have an additional cell wall. The inside of the cell is filled with the cytoplasm, containing different organelles such as the ribosomes. The chromosome, a single circular piece of DNA, encodes the genetic blueprint. Image is taken from Hiremath et al. from 2012 [233].

bacteria and fungi—and chains of spores, rather than by binary fission, during its reproductive phase [72]. DNA, short for deoxyribonucleic acid, is a double helix of two strands of complementary nucleotide base-pairs. Connected to a backbone of sugar and phosphate, the four bases Adenine (A), Cytosine (C), Guanine (G), and Thymine (T) link the two backbones by forming complementary base-pairs (bp) of A-T and C-G. Hence, reading one strand in reverse order and replacing each base with its complementary base will yield the other strand. Both strands are directed with one end being 5' and the other 3'. The reading direction is from 5' to 3'. Additionally to the chromosome, bacteria can have plasmids, smaller rings of DNA, encoding additional information such as antibiotic resistance genes.

Genes are the DNA's fundamental unit of information and their activity is regulated in an orchestrated manner. Coding regions contain genes that encode for a protein. On reading, DNA transcribes into RNA (ribonucleic acid) which is a single-stranded counterpart to DNA. RNA, uses ribose instead of deoxyribose and the base Uracil (U) in place of Thymine (T), pairing with Adenine. RNA exists in various forms: messenger RNA (mRNA) serves as an intermediary storage of genomic information, with some regions being translated into proteins while others function directly as mRNA. Ribosomal RNA (rRNA), encoded by the 16S, 23S, and 5S rRNA genes in prokaryotes, is essential for protein synthesis. The process of copying DNA into a separate strand of RNA is known as transcription, whereas the conversion of mRNA to proteins is termed translation.
Sufficiently long RNA sequences can form a three-dimensional structure, or tertiary structure, through bonds between complementary bases located in different regions of the sequence. Transfer RNA (tRNA) exemplifies this capability, adopting a charac-

teristic cloverleaf secondary structure and an L-shaped tertiary conformation critical for its role in protein synthesis, where it delivers specific amino acids to the ribosome according to the mRNA template. The relationship between the nucleotide sequence and protein synthesis is governed by the genetic code, in which three consecutive nucleotides (a codon) specify a single amino acid. Due to the redundancy of this coding system, multiple codons can encode the same amino acid—a phenomenon known as degeneracy or redundancy of the genetic code. This degeneracy provides a buffer against mutations, as some changes in the third nucleotide of a codon may not alter the encoded amino acid, thereby preserving protein function.

2.2 Genetic variation, Evolution, and Phylogeny

Genetic variation in microbial populations is grouped into (i) single nucleotide polymorphisms (SNPs), (ii) structural variants (SVs) such as insertions and deletions (indels) and (iii) genomic rearrangements or horizontal gene transfer (HGT). In biological populations, evolution can be loosely defined as change in inherited genetic material over several generations, linked to natural selection and genetic drift.

On one hand, genetic drift describes the change of genetic features in a population by random chance, i.e., the decrease of genetic variety through an event that reduces the population size. On the other hand, natural selection acts on phenotype level: advantageous characteristics lead to successful propagation of the corresponding alleles that can then become fixated in the population, provided selection exerts a stronger force than genetic drift. Detrimental characteristics reduce the fitness of an organism, decreasing the reproductive success of the associated allele. As a result, such alleles are likely to decline in prevalence and may eventually disappear from the population. Contrarily, strong genetic drift can also fixate detrimental alleles in a population. Note that the usefulness of a characteristic depends on the environment it occurs in; different factors such as PH, temperature, available resources, as well as competition within a population making it more or less beneficial in different circumstances. New functionality can be acquired through *de novo* emergence of genes, through mutations and within genome structural variation, and from other organisms through horizontal gene transfer (HGT). The ability to detect these events is a requirement to explain changes in phenotype as well as to construct a phylogeny.

Single Nucleotide Polymorphisms and Structural Variants SNPs are point mutations (single base changes) in the nucleotide sequence and can happen spontaneously, typically during genome replication or by mutation-inducing agents. Point mutations can be considered (i) neutral, if it is part of an intergenic region or is a silent mutation (a mutation that does not change the encoded amino acid), (ii) detrimental, if it causes a loss of functionality by altering a gene's sequence or its regulatory region or (iii) beneficial, if a change or gain in functionality causes a better

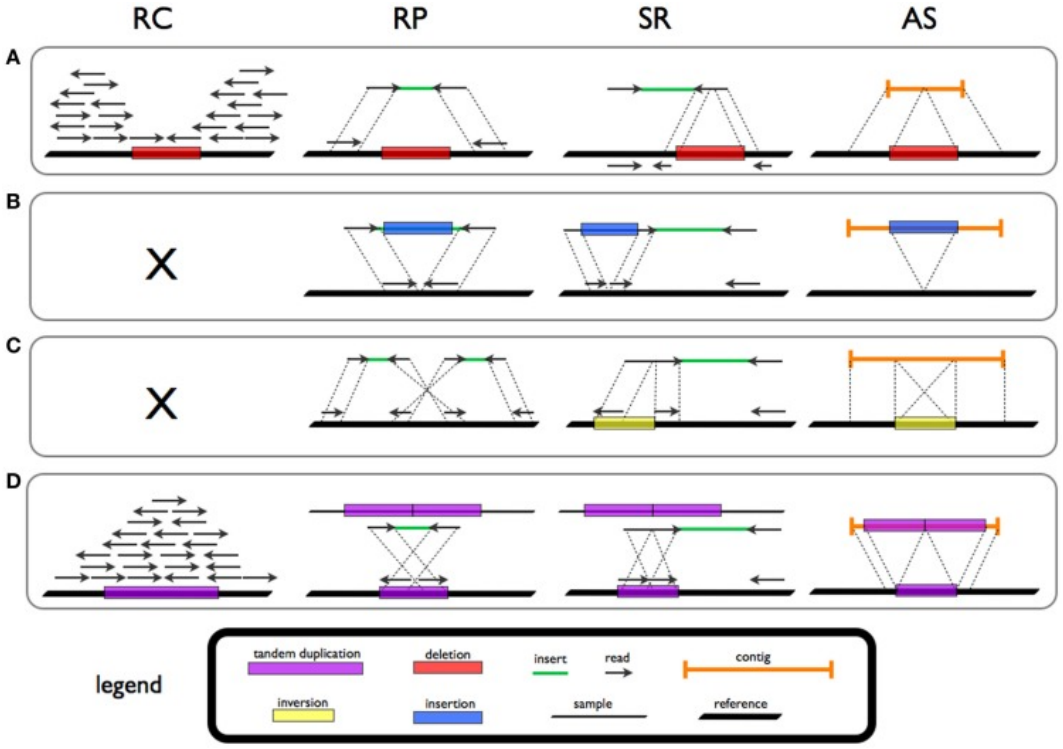

Figure 2.3: This figure shows different structural variants and how to detect them with paired-end reads. RC refers to read count, RP to read-pair, SR to split-read and AS to assembly. **A** describes a deletion event, **B** an insertion, **C** an inversion and **D** a tandem repeat such as microsatellites. The figure is taken from Tattini et al. [273].

adaptation to its habitat. For example, increased pathogenicity can be advantageous for a pathogen if it enhances transmission or survival within a host. In Plasmodium (the malaria parasite), mutations that increase virulence—such as those promoting faster replication or evasion of host immunity—can improve the parasite's fitness by ensuring higher propagation rates before host death or immune clearance [29]. Similarly, in Staphylococcus aureus, antibiotic resistance mutations are detrimental in drug-free environments but become highly beneficial under antibiotic pressure, allowing resistant strains to outcompete susceptible ones [267].

SNPs are frequently utilised in genotyping for the purpose of strain delineation. Further, the density of SNPs in a genome can be used to calculate the average nucleotide identity (ANI), a measure for sequence similarity. An ANI >95% between two genomes is the accepted standard, that indicates both genomes belong to the same species [86]. However, this threshold can vary for different species [50] (see 2.3.2 for more detail on species definition and 2.3 for more information on bacterial taxonomy). Contrary to SNPs, structural variants (SV) describe mutations affecting longer stretches of DNA, typically larger than 50 bp (base pairs) , and can emerge due to cellular mechanisms such as DNA recombination, DNA repair and DNA replication. Figure 2.3 describes four types of structural variants and how paired-

end reads aligned to a reference can be used for their detection. Insertions (Fig. 2.3B) and deletions (Fig. 2.3A), commonly referred to as indels, describe the gain or loss of one or several nucleotides at a certain position. Frameshift mutations are indels that cause the disruption of a protein by shifting the reading frame. Other structural variants are genomic recombinations that can lead to inversion (Fig. 2.3C), amplification (Fig. 2.3D) and translocation of DNA segments and typically arise as a result of double-strand breaks involving at least two different locations with following (erroneous) re-ligation as part of DNA repair mechanisms. Structural variants are evolutionary drivers as they increase the genetic variance through a) disrupted genes, b) new genes [185] and c) fused genes with chimeric gene products [189] and are able to affect gene expression by changing a gene's position on the chromosome [223].

Structural variants such as microsatellites are used as molecular markers to infer phylogeny or for genotyping [303, 5]. Microsatellites belong to the group of (short) tandem repeats and describe stretches of repetitive DNA motifs. These motifs are 1-6 bp in length and repeat 2-15 times. Microsatellites are highly unstable in copy number and are a result of replication slippage during genome replication. Replication slippage involves the DNA polymerase to pause and dissociate from the strand. Before DNA replication is resumed, the newly synthesized strand separates from the template and hybridizes with a different repeat, causing a loop either in the template or the new strand and causes microsatellite expansion or shrinkage in the new strand [289]. Their evolutionary relevance is based on a mutation rate ($10^{-3} - 10^{-6}$ per generation) much higher than for point mutations that is around 10^{-10} per base per generation [141].

Horizontal Gene Transfer and Homologous Recombination While genetic information is usually passed on vertically from generation to generation, horizontal gene transfer (HGT) and homologous recombination (HR) describe intra-generational exchange of genetic material. Both mechanisms are important factors contributing to genetic variation and thus evolution. The three mechanisms behind HGT are transformation, the uptake of extracellular DNA from the environment, transduction, uptake of DNA through viruses, and conjugation, the transfer of DNA between two cells via a cell-cell connection. For the genetic information to persist, the DNA must be either incorporated into the genome via homologous recombination or illegitimate recombination, or be able to proliferate on a separate plasmid [275]. Consequentially, most HGT events are unsuccessful [275].

Homologous recombination is the exchange of closely related DNA sequences and is a normal part of DNA repair. As genetic information acquired by HGT is susceptible to stochastic loss, homologous recombination can increase the rate at which genetic information is shared within a species [140]. While the acquired information is often neutral or detrimental, acquiring mobile genetic elements (MGEs) through HGT can also have a positive effect; e.g. by carrying antibiotic resistance genes

(ARGs) which grants a selective advantage [271].

Phylogeny is a way to describe the evolutionary history among organisms. In a phylogenetic tree, the leaves represent extant organisms, while the lowest common ancestor (LCA) is the point where their root-to-leaf paths, or lineages, converge. The root node of a phylogenetic tree represents the universal common ancestor of all organisms in the tree. Algorithms that reconstruct phylogenetic trees are almost always assuming strict vertical inheritance of genetic information, which is the transmission of genetic material between generations. However, this assumption does often not apply to bacteria, as a significant amount of genetic material is exchanged between individuals of the same generation (see chapter 2.2), complicating the reconstruction of phylogenetic relationships. To address this, phylogenetic trees are often reconstructed using conserved genomic regions that are shared by all organisms and are unlikely to have undergone HGT. One of the most commonly used universal markers for bacteria is ribosomal RNA (rRNA). Due to its crucial function in the ribosome (translation), rRNA is highly conserved and ubiquitous across all species of bacteria, making it an ideal candidate for constructing phylogenies. Despite the frequency of HGT events in bacteria as well as homologous recombination in closely related strains, vertical evolution exerts a strong phylogenetic signal in bacterial genomes [12].

A group of organisms is monophyletic, if all organisms and their LCA form an exclusive group in the phylogenetic tree. Within a group of genomes, this signal is utilised, often in combination with ANI, to detect strain-sharing and strain-transmission events between hosts [98].

2.3 Describing Bacterial Taxonomy

Bacterial taxonomy, following the principles of Linnaean taxonomy [155], organises microbial species based on their genetic similarity into a hierarchical system of different ranks. The Linnaean system of taxonomic classification was developed by Carl Linnaeus in the 18th century as a means to organise and categorize organisms across all domains based on their degree of similarity [155]. Improving on previous systems, the Linnaean taxonomy puts an emphasis on empirical observation and classification and introduces binomial nomenclature, a composite of a generic term (genus) and a specific epithet, which uniquely identifies a species, a system which is still ubiquitously used in biology. The common taxonomic ranks today group the natural world into Kingdom, Phylum, Class, Order, Family, Genus, and Species, with increasing specificity.

While originally developed to organise plants and animals, Ferdinand Cohn demonstrated in 1872 that this system could be equally applied to bacteria to divide them into genera and species [224]. This was a huge step towards modern microbiology, as categorizing and naming microorganisms is crucial to provide a

common and precise language for microbiologists and clinicians [103]. At first, organisms were classified based on observable characteristics like morphology (shape, structure), pathogenicity, and growth requirements. The basis for modern bacterial nomenclature, *Bergey's Manual of Systematic Bacteriology*, was published in 1923 by David Hendricks Bergey and is used to classify bacteria based on shape, morphology, gram staining, ability to form endospores, motility, and mode of energy production [24]. The nomenclature was governed by the International Code of Botanical Nomenclature .

In 1961, McCarthy and Bolton presented a method to establish genetic similarity through DNA-DNA hybridization, which has since been used extensively in phylogeny and taxonomy [264, 263]. In this method, single-stranded DNA from two organisms is mixed, and the extent of hybridization—indicating genetic similarity—is measured; a similarity of 70% or higher typically indicates the same species, while <70% suggests different species. This allowed for directly comparing genetic similarity of two bacteria rather than relying on morphological and other observable metrics only. DNA-DNA hybridization was long regarded as the gold standard for delineating species among closely related strains. However, due to its technical complexity, reliance on skilled personnel, and high variability between replicates, it eventually became a bottleneck in taxonomic studies of closely related species [264].

In 1977, Carl Woese was first to sequence sufficient amounts of ribosomal RNA samples to propose Archaea as the third domain of life, thus pioneering both the use of the 16S rRNA gene in bacterial taxonomy and phylogeny reconstructions, and simultaneously revolutionizing our understanding of the microbial world. In the same year, Sanger sequencing was presented and thus sequenced microbial DNA was more easily accessible and could be used to determine evolutionary relationships between organisms [236]. This advancement initiated efforts to incorporate phylogenetic information into taxonomic classification. This approach, termed 'polyphasic taxonomy' by Colwell, integrated genetic and phylogenetic information with traditional observational metrics, revolutionizing bacterial taxonomy [54].

Similar to how Linnaean taxonomy and binomial nomenclature were first applied to plants, the naming of bacteria was initially governed by the International Code of Botanical Nomenclature (ICBN). Although a draft Code of Nomenclature for Bacteria and Viruses was approved at the 1947 International Congress for Microbiology, it was never formally adopted, and bacterial names continued to fall under the ICBN. In 1975, the International Botanical Congress formally excluded bacteria from the scope of the ICBN. A fully independent framework for bacterial nomenclature came into effect with the publication of 'The Approved Lists of Bacterial Names' in 1980 [255], which established the official starting point for valid names under the International Code of Nomenclature of Bacteria (ICNB) and marked a reset in bacterial nomenclature. The ICNB—renamed the International Code of Nomenclature of Prokaryotes (ICNP)—was later published in its 1990 Revision [138],

marking the full institutionalization of bacterial nomenclature as a system separate from botanical rules.

One key difference between the ICBN and the ICNP lies in their concept of the type to describe a new species. In botany, the type is typically a physical specimen—such as a dried plant—preserved in a herbarium. In contrast, in microbiology, the type is a living culture, known as a type strain, which must be deposited in at least two publicly accessible culture collections in different countries. This type strain is required for the valid publication of a new species name under the ICNP since 2002 [199]. Before 2002, the deposition of a culture was strongly recommended, but exceptions existed.

'The Approved Lists of Bacterial Names' not only set the foundation for the ICNP and the List of Prokaryotic names with Standing in Nomenclature (LPSN), it also set the foundation for current taxonomic frameworks for bacteria such as the NCBI Taxonomy (National Center for Biotechnology Information). While the ICNP governs the nomenclature—a set of rules of how to name organisms—taxonomy aims at classifying and grouping organisms based on genetic similarity into Ranks.

The NCBI Taxonomy project was initiated in 1991 alongside the development of the Entrez information retrieval system, with the goal of linking nucleotide and protein sequences to the scientific literature and unifying taxonomic classification across all domains of life. This effort spanned major sequence databases—GenBank, the European Bioinformatics Institute (EBI), and the DNA Data Bank of Japan (DDBJ)—all of which later formed the International Nucleotide Sequence Database Collection (INSDC) [69, 52, 79]. A central challenge in this undertaking was the integration of the independently maintained taxonomies from these resources, which required substantial coordination and harmonization.

In 1997, the INSDC members agreed to resolve taxonomic issues—such as nomenclature errors and classification discrepancies—before releasing new sequence data. Under this agreement, all member databases (not just GenBank) began submitting new organism names to NCBI Taxonomy for review prior to publication [69]. In return, NCBI committed to displaying only taxa linked to publicly available sequence entries. Effectively, the NCBI Taxonomy for prokaryotes contains both species that are validly published and have a type strain, as well as candidate species with placeholder names such as those with *Candidatus* prefix. Many of those placeholder names originate from MAGs which are not cultivated but inferred computationally, and contribute to a better representation of microbial diversity. As of 2011, the NCBI taxonomy database listed 11,110 prokaryotic species with a formal scientific name, most of which were represented by at least a 16S rRNA gene sequence [69].

Next-generation sequencing (NGS) was introduced with the 454/Roche platform in 2005, marking a major step forward from Sanger sequencing. Although cheaper than Sanger sequencing, its high cost per base and poor accuracy in homopolymer tracts limited its practicality for routine bacterial genome sequencing [231, 104]. It

was only with the release of Illumina's Genome Analyzer in 2006 [249]—offering higher accuracy and substantially reduced costs—that whole-genome sequencing of bacterial isolate cultures gradually gained wider accessibility [279, 133]. These advancements in sequencing technology—especially by Illumina—facilitated sequencing the collective bacterial DNA of communities and thus enabled metagenomics on larger scopes, such as the HMP or MetaHIT projects [174, 175, 133, 280].

Consequently, an increasing number of putative novel species have been identified based on genomic comparisons, such as similarity of the 16S rRNA gene, conserved housekeeping genes, or the whole genome [206, 172]. However, under the current rules of the ICNP, valid publication of a new species name requires the deposition of a type strain in at least two public culture collections. As a result, genome-based species discovered from metagenomic or uncultured samples cannot be validly named under the Code. To accommodate such organisms, the provisional status *Candidatus* is used. In practice this means that a *Candidatus* name may be proposed when a pure culture (type strain) is unavailable, but sufficient data—usually genomic and other phenotypic or ecological information—exist to characterise the organism [198]. *Candidatus* names represent provisional taxa that do not have formal standing in prokaryotic nomenclature until they are validly published in accordance with the ICNP guidelines [108]. In addition to validly published taxa and *Candidatus* taxa with placeholder names, the NCBI Taxonomy also includes other taxa based solely on proposed or auto-generated placeholder names derived from sequence entries in the INSDC databases. Potential new taxa are then phylogenetically placed based on the sequence data submitted to one of the INSDC nucleotide sequence databases, as nomenclature and classification issues must be resolved before any sequence data is publicly released [240].

As of 2024, the NCBI Taxonomy encompasses 24,821 bacterial species with validly published names, 2,034 with the prefix *Candidatus*, and 106,071 with no validly published or *Candidatus* name [1]. An additional 420,024 bacterial species remain unclassified as their sequence information does not allow for unambiguous phylogenetic placement in the NCBI Taxonomy. Even though modern taxonomy uses phylogeny as a framework, there are cases of taxa with clinical relevance, where phylogeny differs from taxonomy. This is the case for the genera *Shigella* and *Escherichia*, which phylogenetically are in the same genus, yet are distinct in taxonomic genera. On the other hand, *Shigella* and *Escherichia coli* exhibit different pathogenic profiles and are treated differently in clinical and public health context [20]. This shows there are good reasons for keeping the distinct taxonomic naming. The distinct placement in the NCBI Taxonomy can be traced back to the original 'Approved Lists of Bacterial Names' that both genera *Shigella* and *Escherichia* were part of [255].

As previously mentioned, technological advances in sequencing and computational tools as well as cost reduction of sequencing caused a shift towards whole-

[1] https://www.ncbi.nlm.nih.gov/Taxonomy/taxonomyhome.html/index.cgi?chapter=statistics

genome sequencing of whole microbiomes. This shift has resulted in a dramatic increase in the number of bacterial assemblies within the INSDC from 2,433 in 2010, to 714,835 in 2020 [148], with the majority being metagenome-assembled genomes (MAGs) derived from community sequencing rather than from genome assemblies of isolates. Genome assembly is the process of reconstructing a complete genome sequence from short DNA reads obtained by sequencing a pure culture of a single organism, resulting in an isolate genome. In contrast, a metagenome assembly involves assembling draft genomes—MAGs—directly from complex environmental samples containing DNA from multiple organisms, enabling recovery of genomes also from (currently) unculturable, potentially unknown microbes. While MAGs provide valuable insights into microbial diversity, they are generally considered less complete and accurate than isolate genomes. The methods for assembling MAGs and the challenges associated with them will be discussed in more detail in section 2.5.2.

The SeqCode (Code of Nomenclature of Prokaryotes Described from Sequence Data) was developed as an alternative to the ICNP to address a growing need in microbial taxonomy: the formal recognition of prokaryotic taxa that are known only from sequence data and lack a cultured type strain. This is particularly important in the post-genomic era, where advances in metagenomics and single-cell genomics have uncovered vast microbial diversity that cannot yet be cultured, leaving many taxa excluded from valid naming under the ICNP [48]. The ICNP strictly requires deposition of a viable, pure culture in at least two public culture collections to designate a type strain, which is often not feasible for environmental microbes. In contrast, the SeqCode permits genome sequences as type material, provided that they meet defined quality standards, thus allowing for the valid naming of uncultured taxa [298]. Formally launched in 2022, the SeqCode represents a parallel and complementary nomenclatural system, not in competition with the ICNP, but designed to bridge the gap between modern sequencing-based taxonomy and classical nomenclature [94]. As of early 2025, the SeqCode registry (https://seqco.de) is active and accepting submissions, and a growing number of taxa—particularly metagenome-assembled genomes (MAGs) and single-amplified genomes (SAGs)—have been proposed under its framework, though its adoption across the broader taxonomic community is still ongoing and being closely monitored.

The SeqCode process for proposing a new prokaryotic taxon involves submitting a high-quality genome sequence as the type material, followed by writing a formal taxonomic description that includes genomic and phylogenetic evidence. The proposal is then registered in the SeqCode Registry, where it undergoes a community review period of at least 30 days. After any necessary revisions, the name is validated and made official, with the option to publish through SeqCode or a peer-reviewed journal. While the SeqCode allows for taxonomic naming without cultured type strains, it also maintains compatibility with ICNP rules. For MAGs, it is recommended to have more than one MAG per species to ensure genomic representation, but

it is not a strict requirement [**minimum_information**].

Considering the bigger picture, the enormous increase of MAGs in reference databases still poses challenges. The quality of MAGs is measured in terms of completeness and contamination and can be hampered by low read coverage in the sequencing data, low-complexity regions that are difficult to assemble, as well as closely strains in the sequence data which lead to chimeric assemblies (see 2.5.2. These limitations . Limiting databases to isolate genomes is not an option as this would drastically reduce the taxonomic diversity represented in public databases—most bacteria are still considered unculturable or simply have not been cultured as isolates. Hence, taxonomic classification of MAGs is important to establish context and a base for communicating scientific results. At this scale, manually curating taxonomic nomenclature is increasingly tedious and impractical, given the vast amount of novel genetic information and the historical use of phenotypes for taxonomic classification, which facilitated the need for a different approach.

2.3.1 Modern DNA-based Taxonomy

In 2016, ProGenomes [172] was released as a prokaryotic genome resource and taxonomy with a focus on providing extensive meta data such as functional annotation and habitat information. In 2018, the Genome Taxonomy Database (GTDB) was introduced as a solution to existing challenges concerning polyphyletic taxa and the increasing amount of available MAGs [208]. The GTDB taxonomy is a fully automated taxonomic system for bacteria and archaea based on a phylogeny derived from DNA sequences, while still respecting existing taxonomic nomenclature and further offering normalized taxonomic ranks on the basis of relative evolutionary divergence (RED) [309]. Automating the process of generating phylogenies and subsequently developing a concordant taxonomy offers significant advantages. It addresses existing discrepancies between phylogeny and taxonomy that have arisen from classifications based on 16S rRNA or observational methods [317, 20], which have often been maintained for the sake of consistency, despite obvious inaccuracies.

Automating this process can resolve these issues, resulting in a more robust and accurate taxonomy. As an on-going census of microbial diversity, GTDB provides annual updates and went from 143,512 bacterial genomes spanning 23,458 species clusters in the initial release in 2018 to 584,382 genomes spanning 107,235 species clusters in R09-RS220 (2024) [2]. Almost 50% of the incorporated genomes are based on MAGs or SAGs (Single Amplified Genomes), and over 70% of species clusters are represented by MAGs, indicating the vast biodiversity only accessible through metagenomic sequencing. Even though both NCBI and GTDB source their genomes from GenBank and RefSeq, they differ in underlying philosophy and methodology, leading to discrepancies in taxonomic classification. A prominent example are the

[2] https://gtdb.ecogenomic.org/stats/r220

aforementioned genera *Shigella* and *Escherichia*, which GTDB merged into the same genus, *Escherichia*.

The GTDB taxonomy constructs species clusters by centering them around species each represented by one or more genomes that pass quality control measures [209]. For each species, a representative genome is selected based on metadata and quality metrics. Genomes are then assigned to these species clusters using whole genome ANI and alignment fraction (AF), with an ANI threshold of 95%. Genomes that cannot be assigned to any existing species are sorted by quality. Starting with the highest quality genome as a new species representative, remaining genomes are assigned using ANI and AF, repeating this process until all genomes are assigned to a species cluster, either as representatives or members.

The phylogeny in the GTDB taxonomy is based on 120 ubiquitous single-copy proteins, ensuring taxonomic groups form monophyletic lineages with normalized phylogenetic distances between ranks across lineages [208]. The selection of genetic markers is critical to the success of any DNA based approach at structuring bacterial evolutionary history and taxonomy; they must be ubiquitous across all bacterial species to construct a comprehensive phylogeny while providing sufficient resolution to distinguish between species. The 16S rRNA gene, although widely used in amplicon sequencing, is often not present in MAGs, and other marker regions are more practical [49]. As the 16S rRNA gene is highly conserved across closely related microbial taxa, which is why a >98.5% sequence identity threshold is often used for species delineation in 16S-based studies, compared to the >95% average nucleotide identity (ANI) threshold typically applied to whole-genome comparisons. However, due to this high conservation, short DNA reads obtained from sequencing (e.g., Illumina) often cannot be unambiguously assigned to specific species or strains during metagenomic assembly. This ambiguity leads to challenges in reconstructing full-length 16S rRNA genes in MAGs, resulting in their frequent absence from MAG datasets [319].

GTDB's phylogeny is constructed from a multiple sequence alignment (MSA) of concatenated marker protein sequences from all dereplicated and quality-filtered genomes. In the absence of a standardized approach, internal ranks are assigned based on RED to provide uniform placement of taxonomic ranks across all lineages [309, 208]. This addresses a significant issue in the NCBI taxonomy, where taxonomic ranks are not based on evolutionary distances. Additionally, many taxa in the NCBI taxonomy are polyphyletic, meaning that members of the same taxon do not share a recent common ancestor. GTDB resolves this by splitting polyphyletic taxa into distinct monophyletic groups with suffixed taxonomic labels. This approach is particularly beneficial for profiling microbial communities from sequencing data, as accurate profiling relies on phylogenetic signals. Using a taxonomy not based on phylogeny can result in erroneous classifications, emphasizing the importance of GTDB's phylogenetically consistent taxonomy [26].

2.3.2 Of species and strains

The prevalent species definition in the last century was based on a list of phenotypic characteristics of a microorganism, that allowed researchers to systematically categorize bacteria [24]. As phenotypes are reflected in the bacterial genome, DNA-DNA hybridization quickly became the gold-standard for delineating species by measuring the degree of genome similarity [226] and move away from solely relying on observable traits.

However, as bacteria reproduce asexually, the classical species definition of the animal kingdoms does not apply, which raised the question whether bacterial species truly existed. With the broad availability of sequencing technology, researchers identified a natural gap in diversity at the species level by measuring ANI, the pairwise similarity in nucleotide sequence of two genomes, between bacterial genomes at different taxonomic ranks. The ANI gap shows that bacteria within a species often are >95% ANI similar, while between species ANIs are mostly <90% [226, 111], which has been interpreted at supporting the existence of a 'bacterial species'. Therefore, 95% genome-wide ANI is widely accepted as species boundary (Fig. 2.4).

For highly conserved marker regions like the 16S rRNA gene, a 97% ANI threshold is commonly used, though a 98.5% ANI is more consistent with whole-genome species definitions [262, 264, 229]. The 97% threshold was initially chosen as it aligned with DNA-DNA hybridization values above 70%, the gold standard for species delineation at the time [262]. However, this threshold often grouped species that should be distinct, prompting its revision to 98.5% ANI for more accurate species delineation.

The discrepancy between the rRNA 16S gene ANI threshold to the whole genome one arises because the 16S rRNA gene evolves more slowly than most other genomic regions due to its essential role in ribosome function and strong structural constraints, making it highly conserved across taxa. As a result, higher sequence similarity in the 16S gene is needed to reflect the same level of divergence captured by whole-genome ANI.

A bacterial strain is a taxonomic unit below the species level, typically characterised by 99.9-99.99% species identity (Fig. 2.4). The name derives from clinical microbiology, where "straining" was used to grow bacterial strains on an agar plate. However, in literature the definition of strains can vary widely, ranging from a hundred or more, to as little as one nucleotide difference. The definition of a strain often also depends on the research question. Single nucleotide variations can result in phenotypic changes, thus being classified as a different strain [53]. For studying strain-transmission, 'same strain' refers to bacteria that have been transmitted from one host to another in a recent event, with "recent" encompassing time-frames from days to decades [283, 98]. In very closely related strains, the signal-to-noise ration of computational approach and sequencing errors limits the achievable resolution and hence influence the definition of strain [250].

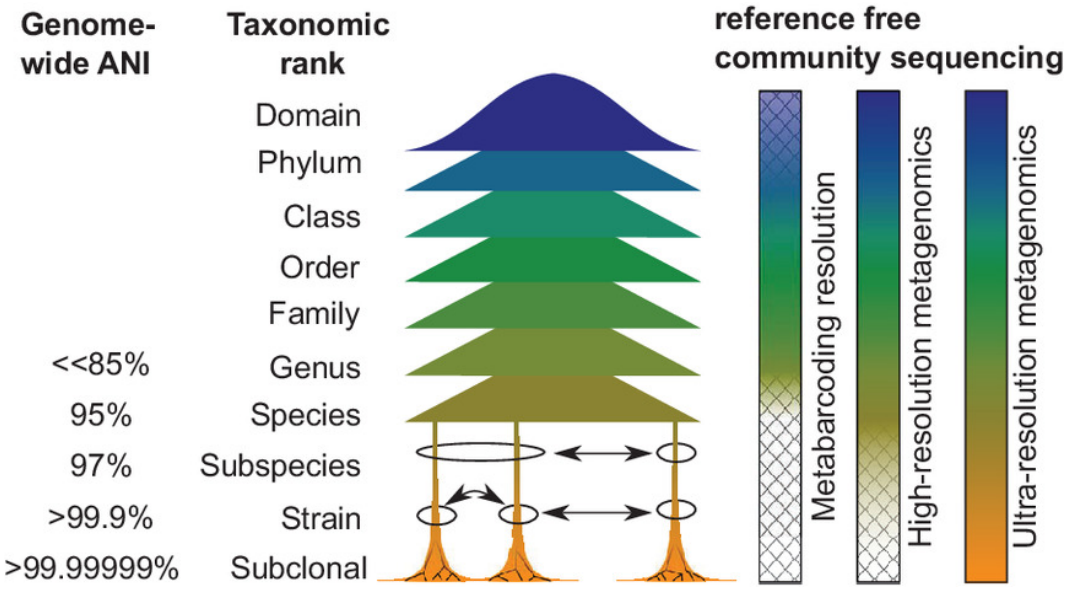

Figure 2.4: Average nucleotide identity (ANI) between prokaryotic genomes of different taxonomic ranks. Image is taken from Hildebrand [95]

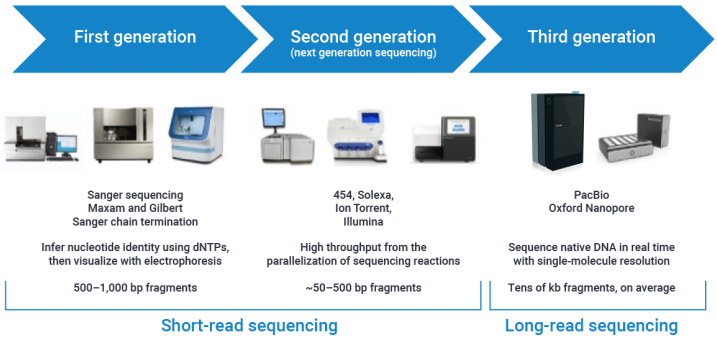

Figure 2.5: Overview over different sequencers and sequencing technologies. Image taken from https://www.pacb.com/blog/the-evolution-of-dna-sequencing-tools/

For the purposes of this thesis, a strain is defined by >99.99% Average Nucleotide Identity (ANI), both between whole-genomes and marker genomes, and the term is sometimes used interchangeably with 'genome of organism'. However, strains do no only differ by SNPs. The pan-genome of a species defines the complete set of genes of all strains within.

2.4 How to study microbial communities?

2.4.1 Sequencing technologies

Until the mid-20th century, studying microbes was confined to observational methods. In current microbiology, the analysis of the genetic material of microbes and

microbiomes through sequencing has become the primary focus. Genomic sequencing enables the determination of the order and identity of the four nucleotide bases in a sequence.

The development of Sanger sequencing in 1977 marked a significant advancement in sequencing technology, facilitating the sequencing of fragments up to 800 base pairs in length [236]. Sanger sequencing employs a sequencing-by-synthesis approach, wherein fluorescently labelled nucleotides sequentially complement single-stranded DNA. Each nucleotide incorporation emits a base-specific light signal, which is detected by a photosensor to reconstruct the genomic sequence.

The very first high-throughput sequencer, the 454/Roche, was released in 2005 [231]. Like Sanger sequencing, the 454/Roche also utilises sequencing-by-synthesis. However, unlike Sanger sequencing, which could only sequence one molecule at a time, the 454/Roche was capable of massively parallel sequencing, allowing the simultaneous reading of thousands of molecules. Hence, the 454/Roche was the first sequencer of what is known as **next generation sequencing**. Other next generation sequencers followed, most prominently the Genome Analyzer by Illumina, increasing the sequencing throughput from megabases (Mb) (1,000,000 bases) to gigabases (Gb) per run [249]. Illumina sequencing starts with binding DNA sequences to wells using adapter sequences. The immobilized DNA undergoes in-situ bridge amplification [22] to create clusters of multiple copies of the same sequence, also called template. Sequencing then proceeds in cycles: each cycle, fluorescently tagged nucleotides complement the sequence by one using DNA polymerase. Each cycle, the wells are excited with a light-source, which them emits a base-specific light signal detected by a photosensor that is interpreted by a computer. Starting with an initial read length of 50 bp with the Genome Analyzer, current Illumina models commonly produce 2x150bp paired-end reads (HiSeq X, HiSeq 3000/4000, MiniSeq, iSeq 100) but also offer longer read-lengths up to 2x300bp paired-end reads (MiSeq, NextSeq[3]). Paired-end reads are achieved by reading the same DNA template from both 5' and 3'. The advantage over single-end reads are the increased DNA sequence length and that both reads can span even a greater distance than their combined length. However, Illumina sequencing is not without errors. Sequencing errors are caused by phasing, a process when single DNA templates within a cluster go out of sync, leading to noise and ambiguity in the base-specific light signal read. Phasing can be caused by high GC-content, homopolymer runs (a long sequence of the same base), imperfect conditions in temperature as well as faulty reagents. Sequencing machines quantify the per-base error-rate in quality scores.

Succeeding next generation sequencing and what was dubbed **third-generation sequencing**, machines were able to sequence 10kb and longer, instead of the 2x300bp

[3]https://knowledge.illumina.com/instrumentation/general/instrumentation-general-reference_material-list/000002826

previously possible. Commercially available from 2011, Pacific Biosciences' (PacBio) single molecule, real-time (SMRT) sequencing-by-synthesis technology is able to produce much longer reads exceeding 20 kb [64]. In SMRT sequencing, the long double-stranded DNA fragments are ligated to adapters (so-called bells) on both ends to produce long circular templates. After adding primer and DNA polymerase, the molecules are put into individual wells on the SMRT cell. While immobilized, the DNA polymerase incorporates labelled fluorescent nucleotides. The emitted light on incorporation is measured in real-time and interpreted by a computer. PacBio sequencing has two sequencing modes. In circular consensus sequencing (CCS), templates are sequenced multiple times in a circular fashion which leads to higher quality long-reads, achieving error rates as low as Illumina's short read sequences (10^{-3}). In continuous long read sequencing (CLR) each circular template is only sequenced once and thus produces longer reads (half of the reads are > 50 kb).

Initially released in 2014, Oxford Nanopore Technologies' (ONT) sequencing is the other of the two major third-generation sequencing methods, producing long-reads. As opposed to Illumina or PacBio, ONT sequencing is not based on sequencing-by-synthesis [253]. Instead, they use a motor-protein that docks—when capturing a long double-stranded DNA molecule—with a nanopore that is fitted into a membrane with high electrical resistance. When applying a potential across the membrane, molecules passing through the nanopore cause a characteristic change in electrical current. Due to the electric field, the DNA naturally threads through the nanopore and the motor-protein acts as a molecular brake to slow down the molecule. Changes in the electrical current as the DNA passes through the nanopore are interpreted by a computer to determine the nucleotide sequence. The cheapest and most commonly used ONT sequencer is the MinION; a portable device for real-time sequencing with a laptop, allowing for on-site sample analysis without a full-fledged lab. Initially suffering from an accuracy as low as 70%, ONT have successively increased the accuracy to 99% by optimizing the chemistry, nanopores, and base calling software.

2.4.2 Read Errors

Sequenced samples come with a per-base error rate which is encoded in the so-called Phred scores. Phred scores are logarithmically related to the per-base error probabilities and are often represented as ASCII characters. To convert the Phred score from the characters in fastq files, a platform specific offset is subtracted from the ASCII code of the character. The offset varies between technologies: old Illumina has an offset of 64 but nowadays 33 is most common and universally used by i.e. Illumina, ONT and PacBio. Equation 2.1 shows how the probability of an error Q is computed from the Phred score P.

$$Q = -10 \log_{10} P \tag{2.1}$$

An awareness for sequencing errors is crucial to a thorough analysis of sequencing data. However, errors not only arise during sequencing. DNA damage, leading to wrong base readings, can be introduced at any step in the workflow. During sample preparation, faulty handling (exposure to UV radiation, wrong storage temperature) can lead to DNA alterations that are passed along through the entire subsequent workflow. Heating and formalin fixation (a stabilization method for clinical applications) increase the rate of spontaneous cytosine deamination and lead to transversion mutations (mutating from a purine to pyrimidine or vice-versa) in PCR amplification [10, 58]. Different methods for DNA fragmentation can also lead to deletions and substitution errors [126, 235]. In the (optional) PCR enrichment steps, the DNA polymerase may incorporate a wrong base; this is known as incorporation error. In Illumina there is an additional PCR step to form sequence clusters in the wells. During sequencing, context-driven DNA polymerase incorporation errors, phasing, damaged DNA, overlapping clusters, issues with cluster formation and more are potential error sources [214, 237, 125]. The per base error rate for substitutions in HiSeq, an Illumina sequencer released in 2010[4], is approximately 3.3×10^{-3} [237]. For this error rate, one substitution error in a read pair with 2x150bp is expected. Note, however, that errors are not uniformly distributed across the read but occur preferentially towards the end of a read. When analyzing these errors, base-related preferences in substitution as well as motifs preceding the error have been identified [237, 162] . Unless explicitly mentioned, the following approaches will focus on methods developed for short-read analysis.

2.4.3 Approaches for sequencing complex microbial communities

Broadly speaking, there are two different sequencing methods for analysing the DNA of microbiomes. Amplicon sequencing, also called 'metabarcoding', is the targeted sequencing of a gene or genes, such as 16S or 23S, through specific primers. (Shotgun) Metagenomic sequencing describes sequencing the collective genetic material of a microbial community, called the metagenome [93]. It follows that metagenomics is "the genomic analysis of a population of microorganisms", a term that was coined by Handelsman et al. in 1998 [93] (Fig. 2.6 A). In literature, however, metagenomics is frequently used as an umbrella term for both whole genome and amplicon sequence analysis. Similarly, metatranscriptomics describes the analysis of mRNA sequencing and metaproteomics refers to the analysis of proteins of whole microbial communities; however, this thesis focuses solely on metagenomics.

[4]https://investor.illumina.com/news/press-release-details/2010/Illumina-Announces-HiSeqTM-2000-Sequencing-System/

Amplicon Sequencing

Both amplicon sequencing and metagenomics are different in taxonomic resolution, their ability to resolve functional potential, sequencing cost, computational requirements, and complexity of analysis. In amplicon targeted sequencing, or short amplicon sequencing, a molecular marker such as 16S ribosomal RNA (rRNA) for prokaryotes, or 18S rRNA or ribosomal internal transcribed spacers (ITS) for eukaryotes, is amplified using polymerase chain reaction (PCR) and subsequently sequenced. Due to their functional conversation, 16S, 18S, and ITS genes have a low mutation rate and are present in all microorganisms [304]. The 16S region is about 1,550 bp long and comprises both conserved and variable regions. Because of the limited read length in NGS sequencing, only part of the variable regions V1-V9 is accessible. The taxonomic resolution of this short sequence is limited to genus (sometimes species) level [195]. However, recent approaches using long reads from third generation sequencing for amplicon sequencing can be used to resolve microbes at species and sometimes even strain-level [113].

Amplicon reads naturally align at the primer region which allows for *de novo* similarity-based clustering into operational taxonomic units (OTUs). OTUs can serve as proxies for *de novo* generated genera, with the similarity threshold for clustering defining the taxonomic rank. The established threshold of 97% identity is commonly used to delineate species-level clusters [262]. Annotation of OTUs with the help of databases such as SILVA [317], RDP [293], Greengenes [57], or our KSGP [88] establishes context with respect to known taxa, allowing the comparison of results to those in literature. Common tools for OTU clustering are, for example, CD-HIT [78] or UPARSE [62], and are often integrated into pipelines to automate the whole analysis process [201]. By clustering amplicons at an ANI threshold, noise from sequencing errors is 'drowned out'. Alternative to OTUs, amplicon sequence variants (ASVs) aim to denoise sequences from sequencing errors while retaining true variation. First coined and implemented in DADA2 [39] in 2016, ASVs are now a widely used alternative to OTUs. Yet, in direct comparison ASVs often still produce more false positives than OTU-based methods [201].

Amplicon sequencing is still the most cost-effective way to study the taxonomic composition of microbiomes, and is especially useful for environmental samples with mostly unknown biodiversity. Compared to metagenomics, a low cost-factor and established easy-to-use tools [201], amplicon sequencing still is an accessible option for reference-free taxonomic analysis with moderate resolution (Fig. 2.4). Beyond this, some tools even extrapolate functional capacity from amplicon sequencing data [296]. However, it's important to approach this interpretation with caution, as inferred functional capacity may not precisely reflect actual biological function. Moreover, it's noteworthy that the difference in functional capacity among microbes within the same genus can be extensive. While amplicon sequencing data analysis is

not without flaws [220, 237], the commonly used pipelines are well tested and problems like copy number variation, chimeric reads, and host contamination are known and can mostly be accounted for.

Metagenomics

Metagenomic sequencing data allows for all the analyses possible with amplicon data, and many more, but is a) more costly and b) computationally more demanding. Taxonomic profiling of metagenomes is either conducted through reference-based or reference-free methods, sometimes also referred to as assembly-free and assembly-based methods. Reference-based methods use alignment (or comparison) of individual reads against an often pre-built database, representing a set of pre-selected genomes and thus taxa, to collect evidence and reconstruct the taxonomic profile (Fig. 2.6 B). This will be discussed in greater detail in the next section (2.5) and in Chapters 4 and 5, where I present two approaches I developed during my PhD. Metagenomic data further allows for functional profiling, as genes can be detected directly, not only via the known genomes of detected taxa, yielding a comprehensive picture of functional capacity in the genome. Further, metagenomic sequencing data allows for a much higher taxonomic resolution than amplicon sequencing, with strain-level analyses possible with modern pipelines. This can be used to study strain-transmission, HGT and SVs, and genome-wide association studies (GWAS) enable searching for single genes or variants associated with diseases.

In terms of complexity, reference-based methods are in between amplicon sequencing analysis and reference-free metagenomics. Reference-free (or assembly-based) methods are necessary to study the genome of unknown microbes (Fig. 2.6 B). Assembly-based workflows start with the *de novo* assembly of reads into longer contiguous sequences (contigs). Subsequently, the contigs are distributed into bins based on sequence properties and abundance, to group contigs from the same genome. Notably, *de novo* assembly alone has a runtime approximately 50- to 100-fold greater than that of reference-based profiling [322]. Metagenomic assembly will be discussed in more detail in section 2.5.2.

2.5 Computational microbiome analysis

Along with the advent of sequencing, a plethora of bioinformatic algorithms and tools have been developed for tasks such as similarity-based pairwise and multiple sequence alignment (MSA), *de novo* genome assembly from sequencing data, prediction and extraction of genes, reconstructing phylogenetic trees from MSAs, identification of SNPs between sample and reference genome (SNP calling), quality filtering of sequencing data, and many more. Gradually, these tools were adapted for analysing microbial sequencing data, and over time, specialized tools and pipelines for meta-

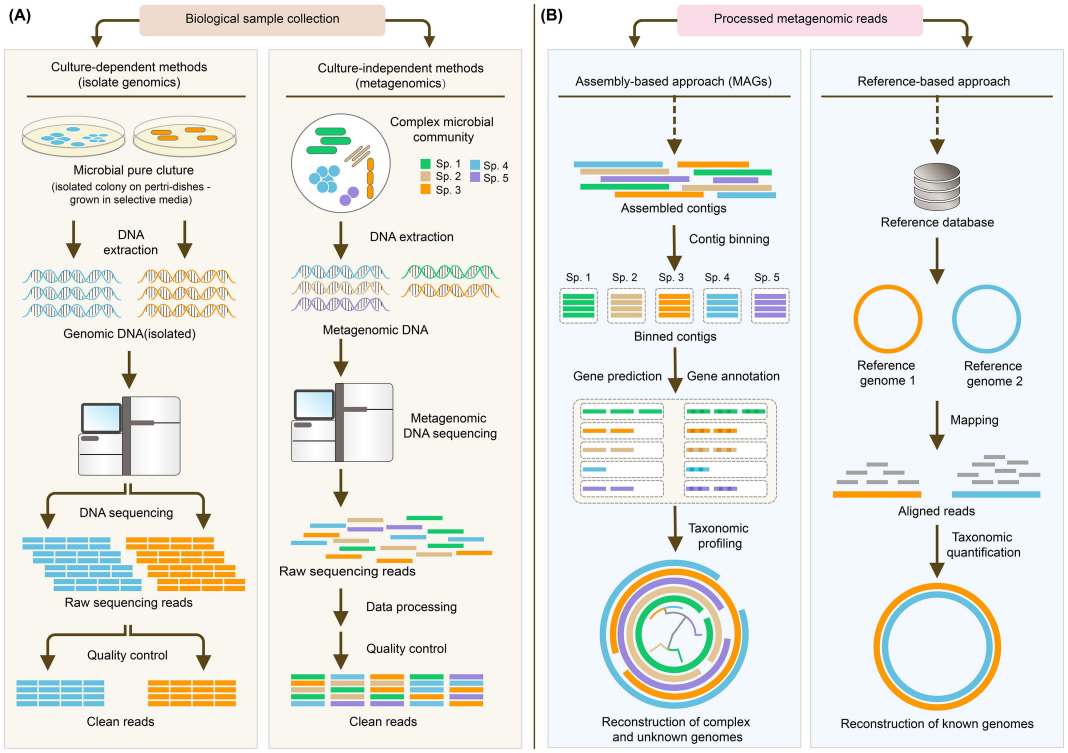

Figure 2.6: A, Schematic depiction of culture-dependent and culture-independent methods. Metagenomics emerged from the necessity to study microbial communities *in vivo* as opposed to *in vitro*, to study the microbe-microbe interactions as well as microbe-host interactions. B, Schematic of the two major metagenomic approaches. The assembly-based approach aims to reconstruct draft genomes of known and unknown microbes from sequencing data to capture both known and unknown microbial diversity. Reference-based methods rely on databases with reference genomes to profile metagenomes and analyse the known portion of microbial diversity. Assembly-based approaches are orders of magnitude more complex in terms of computational requirements as well as quality control. Image is taken from Yang et al. [313].

genomics and amplicon sequencing analysis emerged, as I will discuss later (Table 2.2).

Probably the most common bioinformatic task across all bioinformatic disciplines is sequence comparison. One of the first tools for this task that is still developed and used today, is BLAST (Basic Local Alignment Search Tool) [6]. Published in 1990, BLAST improved the speed of similarity-based sequencing search via a seed-and-extend algorithm (refer to Section 2). Further, it was the first tool to introduce a mathematical framework for quantifying the probability that a hit occurred by chance (E-value). In general, alignment tools such as BLAST allow for building custom reference databases to search against, and output the best scoring hit, calculated based on similarities and differences such as mismatches, insertions, and deletions (See 2.2). Early tools for metagenomics and amplicon sequencing analysis often used BLAST for its highly sensitive and accurate sequence search [105, 246, 96] .

However, many taxonomic profilers eventually replaced BLAST with aligners of a new generation: aligners that offered substantial improvements in speed over BLAST while maintaining a similar sensitivity and accuracy. Bowtie [137], bwa [146], and the later published DIAMOND [37] are still among the most used bioinformatics tools. Examples for this transition are mOTUs, MetaPhlAn and MEGAN, which switched from BLAST to bwa, bowtie2, and DIAMOND, respectively.

2.5.1 Taxonomic profiling

To reconstruct the taxonomic composition in a sample, called taxonomic profiling, read alignment is often an integral part. Many taxonomic profilers like MetaPhlAn 4 and mOTUs 3 (and some of their earlier versions) utilise read alignment against a custom curated database to predict which taxa are present in a sample. This reference-based profiling is fast, but heavily dependent on which taxa are represented in the database. One strategy here involves marker genes, pre-selected areas of reference genomes, which exhibit a strong phylogenetic signal to improve taxonomic accuracy, and help to reduce the size of databases. MetaPhlAn 4 uses a set of species-specific marker genes (for each species), which are selected for occurring in every member of the species (coreness), but not in genomes outside of the species (uniqueness). mOTUs3, on the other hand, uses universal marker genes, genes that are present across all bacteria. In comparison, species-specific markers often exhibit higher accuracy, but are difficult to accurately determine for species with only one or two reference genomes available. On the other hand, universal marker genes are easy to extract, but often less accurate in clearly distinguishing between two related species.

The incredible success of tools like Kraken and Kraken2 (4204 and 3671 citations, respectively, as of 13th June, 2024 [5]) further showcased a demand for fast algorithms and analysis software. This was partly because sequencing cost had dropped and more data was generated every year [183], but also due to the growing amount of sequencing data deposited on public repositories. Faster tools and novel algorithms facilitated large-scale meta-analyses, efforts to analyse data from multiple studies and cohorts [210]. An effort that initially required resources of multiple labs [11] is now possible for a single lab [76]. With the growing amount of available sequencing data and taxonomic diversity in reference databases, the technical challenges were not only limited to performance. Due to the vast amount of unculturable bacteria, there was a shift towards MAGs, draft genomes of individual strains reconstructed from complex communities (See Section 2.5.2). This led to reference databases quickly harbouring more MAGs than assembled genomes from isolates. Table 2.2 provides an overview over selected tools for taxonomic profiling from metagenomic data and amplicon sequencing data.

[5] https://scholar.google.co.uk/ search terms 'Kraken' and 'Kraken2', cited by

Year	Tool	Type	Ref	DB	Method	Taxonomy	Databases	External Tools	Description
2007	MEGAN	WGS	DNA, Protein	Whole-genome	Alignment	NCBI	Custom	BLASTX, BLASTN, BLASTZ	Tool for profiling and analysing sequencing data. Alignments are done with BLAST, has a user interface.
2009	mothur	Amplicon	DNA	Amplicon	OTU clustering	RDP	RDP	Implements DOTUR, CD-HIT, RDP, UniFrac and more.	Pipeline for analysing and visualising amplicon sequencing data. Combines previously existing resources for OTU clustering, pairwise phylogenetic distance and taxonomic classification.
2010	QIIME	Amplicon	DNA	Amplicon	Alignment, RDP	RDP	RDP, Custom	CD-HIT, mothur, RDP, BLAST, FastTree, RAxML, MUSCLE and more	Pipeline for analysing and visualising amplicon sequencing data. Combines existing tools for OTU clustering, MSA, phylogenetic tree and taxonomic classification.
2012	MetaPhlAn	WGS	DNA	Clade-MG	Alignment	NCBI	Fixed pre-built	BLASTN	Taxonomic profiling from WGS data, uses BLAST to align reads against custom marker database. Database contains both species markers and markers on higher taxonomic levels (The database covers 1,221 Species from 2,887 genomes).
2013	mOTUs	Amplicon	DNA	Universal-MG	Alignment, Assembly, Clustering	Custom phylogeny, NCBI	Fixed pre-built	MOCAT, fetchMG, USEARCH	mOTUs is part of MOCAT and uses de novo read assembly and fetchMG to extract marker genes from metagenomic sequencing data. For their database, mOTUs uses 10 universal marker genes of the total 40 of fetchMG. Extracted markers are compared to markers in the database or de novo clustered. Covariance groups markers into linkage groups to resemble species (mOTU-LG), (1,753 species from 3,496 genomes).
2013	UPARSE	Amplicon	DNA	-	OTU clustering	-	-	-	De novo clustering of amplicons into OTUs.
2014	Kraken	WGS	DNA	WG, Custom	K-mer based	NCBI	Custom, RefSeq available.	-	Fast k-mer based approach for taxonomic binning. Reads are classified based on k-mer hits with respect to a prebuilt database. Non unique k-mers in the database point to the LCA of all their occurrences. Reads are classified by counting hits in the taxonomic tree and choose the lowest hit with the most support in the lineage. User can build databases from custom genomes.
2014	LotuS	Amplicon	DNA	16S, 18S, SSU	OTU clustering	SILVA, RDP, greengenes	greengenes, SILVA, RDP	UPARSE, RDP, BLAST, RDP, FastTree, ClustalOmega, usearch, uchime, uparse, BLAST+	Complete pipeline for amplicon sequencing data. Provides fast and extensive quality filtering and dereplication of amplicon sequences, offers multiple clustering algorithms and taxonomic databases.
2015	CLARK	WGS	DNA	WG	K-mer based	NCBI	Custom, RefSeq	-	Fast k-mer based taxonomic binning, uses target-specific k-mers in the database and removes all k-mers that are not unique at a certain target level (species, genus, etc).

2015	MetaPhlAn2	WGS	DNA	WG	Alignment	NCBI	Fixed pre-built	Bowtie2	Alignment against species-specific marker genes. Extends marker gene concept with quasi-markers by lowering the uniqueness requirement (marker gene must not be present in other species). The database covers more than 7500 Species from over 16,000 Genomes.
2016	Kaiju	WGS	Protein	WG	Pseudo-alignment	NCBI	Custom	-	Kaiju uses a data structure often used in alignment tools, the FM-index, without doing actual alignment (pseudo alignment). It translates all reads into its six reading frames and compares them against protein sequences in the database.
2016	Centrifuge	WGS	DNA	WG	Pseudo-alignment	NCBI	Custom	-	Centrifuge uses the FM-index to perform pseudo alignment (see Kaiju) against a pre-built database. It finds and identifies short k-mer seeds shared between read and reference and then extends those seeds until there is a mismatch. Hit species are scored with respect to the lengths of their exact matches and the highest scoring species are reported.
2016	DADA2	Amplicon	DNA	Amplicon	ASV	-	-	-	DADA2 introduced ASVs (Amplicon Sequence Variant). DADA2 denoises amplicon sequences into ASVs by removing sequencing errors based on a bayesian model.
2017	**Bracken**	WGS*	DNA	WG, Custom	Bayesian-reestimation	NCBI	Custom	Works with Kraken, Kraken2, and KrakenUniq output.	Transforms the Kraken (+ KrakenUniq and Kraken2) output into abundance profiles by using bayesian-reestimation.
2018	KrakenUniq	WGS	DNA	WG, Custom	K-mer based	NCBI	Custom	-	Counts unique k-mers to improve precision. User can build databases from custom genomes.
2019	**Kraken2**	WGS	DNA, Protein	WG, Custom	K-mer based	NCBI	Custom	-	Improved speed and memory (85% less memory than kraken, 8 times faster), accuracy similar to kraken. The user can build databases from custom genomes.
2019	mOTUs2	Amplicon	DNA	Universal-MG	Alignment	NCBI	Fixed pre-built	bwa-mem	mOTUs2 has a custom database with a phylogeny built via de novo clustering of 10 universal marker genes across genomes. The database comprises marker genes from both sequenced genomes and de novo assembled metagenomes. For profiling, all reads are aligned against the marker gene database with bwa-mem (The database covers over 7,700 Species from over 25,000 genomes).
2021	MetaPhLAn3	WGS	DNA	Species-MG	Alignment	NCBI	Fixed pre-built	Bowtie2	MetaPhlAn 3 has a custom database of species-specific marker genes, extracted from the ChocoPhlAn 3 database. During profiling, reads are aligned to the marker genes using bowtie2 (The database covers 13,475 species from 99,227 genomes).
2022	**mOTUs3**	WGS	DNA	Universal-MG	Alignment	NCBI, GTDB	Fixed pre-built	bwa-mem	mOTUs3 has a custom database containing the same 10 universal marker genes as previous versions (fetchMG). The previous database is extended by adding marker genes from MAGs representing previously unrepresented species (The database covers more than 33,000 species from over 600,000 genomes).
2023	**MetaPhlAn4**	WGS	DNA	Species-MG	Alignment	NCBI, GTDB	Fixed pre-built	Bowtie2	MetaPhlAn 4 has a custom reference database containing species-specific marker genes. They distinguish between uSGB and kSGB (unknown or known Species-level genome bin) depending on whether a bin is only comprised of MAGs (The database covers 26,970 species from 729,195 genomes).

Table 2.2: An overview of selected tools for taxonomic profiling. Fixed pre-built under **database** means a database is provided with the tool. Custom means the user can provide their own set of genomes as reference. The tools in the Table are MEGAN [105], mothur [239], QIIME [40], MetaPhlAn [246], mOTUs [272], UPARSE [62], Kraken [308], LotuS [96], CLARK [200], MetaPhlAn 2 [277], Kaiju [173], Centrifuge [123], DADA2 [39], Bracken [159], KrakenUniq [32], Kraken2 [307], mOTUs2 [179], MetaPhlAn 3 [19], mOTUs3 [232], MetaPhlAn 4 [26].

2.5.2 Expanding known diversity through Metagenomic Assembly

A prime example for the specialization of bioinformatics tools towards metagenomics is genome assembly. In the past, culture-dependent methods were used to grow individual organisms in isolation to then study their interaction [205]. However, 80% of the gut bacteria are considered 'unculturable' and relying on culture-based methods neglects a huge portion of the community [131]. Despite recent efforts and advancements in culturing bacteria, assembly-based metagenomics are still more accessible.

Metagenomic assembly

[35, 116]. As a means to shift away from culture-dependent workflows towards culture-independent microbiome analysis, metagenome assemblers were developed to allow for reconstructing draft genomes from high complexity sequencing data, spanning hundreds of bacterial species - a task conventional sequence assemblers were not capable of.

Metagenome assembly faces several intrinsic challenges that affect genome recovery, contiguity, and interpretability—especially for conserved and complex genomic regions such as the 16S rRNA gene. One major issue is that 16S rRNA genes are typically multi-copy and highly conserved, making them difficult to assemble using short-read sequencing. Their similarity across species often leads to collapsed or chimeric contigs, resulting in underrepresentation or misassembly of this crucial taxonomic marker [319]. Repetitive elements, such as transposases and rRNA operons, introduce similar problems by confounding the assembler's ability to resolve copy number and structural arrangement, especially in species-rich environments [242]. Additionally, strain heterogeneity—the presence of closely related but genetically distinct strains—introduces conflicting signals during assembly, leading to fragmented contigs due to low allelic consensus [221]. Coverage bias further complicates assembly: low-abundance taxa are often under-sampled and poorly assembled, while dominant organisms may produce highly uneven coverage that disrupts contig extension and scaffolding [190]. Furthermore, high-GC content regions are known to be systematically underrepresented in Illumina data, skewing both assembly and downstream gene prediction [36].

The first dedicated metagenome assemblers date back to 2011 and performed significantly better than previous general purpose assemblers on metagenomic datasets [212, 28, 186]. One of the most popular metagenome assemblers today is MEGAHIT [143] (5450 citations as of 19th June 2024 [6]), as it massively improved computational requirements as well as assembly completeness and contiguity over other assemblers. Further improvements on assembly can be achieved, for example by co-assembly of samples that are from a single host, as a high level of strain retention within an indi-

[6] https://scholar.google.co.uk

vidual is expected [291]. This approach artificially increases the sequencing depth for low-abundant microbes, however, may lead to ambiguous and fragmented assemblies in species with high strain-level variability [241].

Metagenomic binning

After assembly, binning tools group contigs together in 'bins' based on their genomic origin. Binning tools typically use a combination of sequence properties and abundance-based estimates of single contigs to group these together in a genome "bin". Sequence properties such as tetra-nucleotide frequencies [117] or GC-content [281], exhibit a species specific signal that is also present in the individual contigs. However, short contigs are inherently hard to reliably assign to bins, as with decreasing length the signal-to-noise ratio degrades rapidly [243]. Abundance-based binning works with the premise that contigs from the same genome have similar abundance within a sample and contigs between multiple samples exhibit a similar abundance pattern across a genome [175].

Currently, popular binning tools are CONCOCT [5], MetaBat [115], and SemiBin [203]. Additional tools like DAS Tool [251] and metawrap [282] refine the output of other binners based on genome completeness and contamination.

MAG quality

Estimating genome completeness and contamination is a key process in building high-quality MAGs from metagenomes. The tool CheckM marks an important development as it improves quality control by estimating contamination and completeness of MAGs [207] and has recently been superseded by CheckM2 [44]. CheckM analyses lineage specific, single-copy marker genes, to assess the degree of contamination and completeness. GUNC, MetaQUAST, MAGpy, and MAGpurify are complementary tools used to assess and improve the quality of metagenome-assembled genomes (MAGs), each targeting different aspects of contamination and chimerism [197, 178, 269, 187]. GUNC evaluates MAGs for taxonomic inconsistencies by checking for discordant lineage signals across genes, making it particularly effective at identifying chimeric and contaminated bins that traditional tools like CheckM might miss. MetaQUAST is an assembly evaluation tool that compares contigs to reference genomes (when available), identifying misassemblies, including chimeric contigs, structural errors, and unaligned regions. In contrast, MAGpurify specifically targets contaminant and chimeric contigs within MAGs. It uses multiple signals—such as GC content, tetranucleotide frequencies, read coverage, and taxonomic assignments—to score and filter out suspect contigs, improving MAG integrity. MAGpy is a scalable pipeline for automatic MAG annotation that takes multiple MAGs to check their quality, detect chimeras, suggests a taxonomy and places them in a phylogenetic tree. Together, these tools improve MAG reliability for downstream taxonomic and

functional analysis.

Alongside the development of tools for quality control of MAGS, a standard has been established as a guideline for minimum quality requirements for MAGs. The Minimum Information about a Metagenome-Assembled Genome (MIMAG) standards, proposed by the Genomic Standards Consortium [274], provide guidelines for evaluating and reporting the quality of MAGs. MIMAG defines three quality tiers—high-quality, medium-quality, and low-quality—based on metrics such as genome completeness, contamination, presence of rRNA genes, and the number of tRNAs. For example, high-quality MAGs must be >90% complete, <5% contaminated, and include the 23S, 16S, and 5S rRNA genes along with at least 18 tRNAs. These standards are widely adopted by taxonomic databases like GTDB and frameworks like SeqCode to determine the eligibility of MAGs for inclusion and potential naming. By enforcing these thresholds, MIMAG helps ensure that only reliable genome reconstructions are used in downstream analyses, including taxonomy, functional annotation, and evolutionary inference.

Advances in metagenome assembly and binning tools, along with the development of quality assessment methods and community standards, have increased confidence in MAGs, leading to their gradual integration into reference databases. A good example of this development is GTDB [206]: For the first version of GTDB (r89, 2019) only 15% of all ∼146,000 genomes were MAGs, a number which grew to ∼45% of ∼597,000 genomes (r220, 2024) within five years. Moreover, this progress has driven a rapid expansion of species representation in taxonomic profilers, which have updated their databases by incorporating MAGs from diverse environments. However, assembly-based pipelines also face challenges beyond computational demands. Low-abundant taxa are difficult to assemble, which is why some approaches combine reference-based and reference-free methods in taxonomic profiling to increase sensitivity. Despite the importance of MAGs to cover taxonomic diversity, reference-based pipelines are needed for fast and sensitive classification as they detect lower abundant taxa, which lack the sequencing depth for assembly, at a fraction of the time.

2.5.3 Strain-level analysis

Historically, the species definition emerged as a result of distinct phenotypic properties between bacteria. However, many species equally exhibit intra-specific phenotypic diversity [285], necessitating an increased taxonomic resolution to strain-level. While functions are phylogenetically conserved to a certain degree, environmental preferences show a greater contribution to the pan-genome's (the collective genome of organisms within a species) variance than phylogenetic inertia, which is the limiting factor of previous adaptions on future evolution [164]. Thus, classifying the

microbiome on genus or species level neglects potentially crucial strain-level differences within a species, such as pathogenicity or antibiotic resistance [142]. High resolution strain-level metagenomics is therefore the key to understanding not only how the microbiome changes over time, but also to investigate its role in disease and identify its varying functional capacity. Further applications include detecting transmission events of strains between two samples (such as two human hosts) that can only be reliably detected if the sequencing resolution is adequate to identify a common strain [314].

To understand how strain-level analyses can be conducted, it is important to revisit the drivers of genetic diversity within a species. Genetic diversity emerges from DNA replication errors, DNA repair errors, and mutagens, which causes SNPs, insertions, deletions, but also structural variants such as inversions, duplications and insertions. Mechanisms for exchange of genetic material, such as HGT or homologous recombination, consolidate novel genetic diversity through genetic exchanges within a species, thus decreasing genetic diversity. However, structural variants are hard to detect with short-reads [297] (See Chapter 2.2), whereas long-read sequencing would allow for an easier SV detection. For example, pan-genome based methods distinguish strains based on their gene content, but often require metagenome assembly or sufficient reference genomes available [321] . This approach was used by Zhu et al. in a reference-based approach to investigate inter-individual gene content of bacterial strains of the same species [325, 91].
However, the most widely used methods to achieve strain-resolution are SNP based, allowing for quantification of the genetic distance between entire genomes or specific marker regions. SNPs are easily identified from short-read alignments against a reference genome using tools such as bcftools mpileup [144], freebayes [80], snippy [245], or GATK haplotypecaller [216]. For example, Hildebrand et al. used *de novo* assembly and SNPs calling in species core genes to investigate strain-retention and dispersal across >1,000 species and >5,000 samples [98]. It is also important to distinguish between *de novo* and reference-based strain-methods. GT-Pro, for example, has a database built around SNPs from known genomes. It can genotype strains, but ultimately lacks the resolution to incorporate unknown SNPs and other genetic variation not represented in the reference database. Hence, strain-tracking is possible, but limited to known variants and thus will not match the accuracy and resolution of methods that can *de novo* detect SNPs.
Research often assumes for simplicity that the human gut is predominantly colonized by a single (dominant) strain per species, with at most a few strains per species (conspecific strains) [306]. This assumption is common in strain-resolved tools, which typically consider only one strain per species in a single metagenome. While this simplification is useful in many contexts, it may not fully capture the true complexity of the gut microbiome, where multiple, potentially low-abundant, strains in addition to the dominant strain can coexist in certain environments [81, 306, 301]. How-

ever, several tools are specialized on disentangling con-specific strains. StrainScan, for example, employs a tree-like data structure for k-mers between many closely related strains to accurately disentangle conspecific strains. Unfortunately, StrainScan needs extensive database building for just one species and requires a multitude of reference genomes to work [150]. ConStrains on the other hand requires only one reference genome per species, rather linking SNPs with similar allele frequencies to disentangle strain signatures; however, it requires a high coverage to be accurate [160]. In practice, most strain-level tools which are not specialized for analysing conspecific strains rely on a strong signal of a dominant strain and ignore potential subdominant strains.
Strain-level pipelines can be separated into complete pipelines, starting from raw sequencing reads, and pipelines which require prior read alignment or assembly, as well as tools using whole-genome alignment vs. marker regions. MIDAS2 [323] starts from raw reads by identifying species in a sample using marker gene alignments, to then identify SNPs for all abundantly present species across the whole genome. Further, MIDAS offers extensive pangenome analysis. StrainPhlAn 4 builds on the MetaPhlAn 4 marker gene alignments to identify SNPs, reconstruct a consensus sequence, and build an MSA across samples on shared species [26]. inStrain [196] takes read alignments as an input and does not provide a reference database. Unlike MIDAS2 or StrainPhlAn4, which are consensus based tools, inStrain allows for ambiguous matching by including includes minor SNP alleles into ANI computation. While this approach is sensitive for detecting shared SNPs, it may also lead to an increased false-positive SNP detection rate. SameStr [215] uses the same strategy and compares strains with respect to all possible variants, both minor and major alleles. Unlike inStrain, SameStr builds on top of both mOTUs3's universal marker genes and MetaPhlAn 4's marker genes, offering an alternative to StrainPhlAn 4 to calculate distances, but does not reconstruct a consensus for MSA generation. metaSNV2 also uses read alignments as input to delineate strains in metagenomes and further provides a recommended database (ProGenomes2) [55, 286] . metaSNV2 stands out by being able to detect subspecies clusters, distinct phylogroups between species- and strain-level.

The choice which software to use boils down to the available computational resources, research question, and usability. Across all mentioned tools, StrainPhlAn 4 is reported to be fastest due to its utilization of marker genes instead of whole-genomes [284].

2.5.4 Long-reads and hybrid assembly improve MAGs

Utilizing long-read sequencing for metagenome assembly holds significant potential for enhancing the quality of metagenome-assembled genomes (MAGs) by overcoming limitations of short-read assembly alone. Short-read assembly is difficult because a)

low complexity regions with simple sequence repeats are hard to assemble, especially because short-reads are often not long enough to span the entire region, b) short reads have shorter DNA sequence overlaps that give rise to matches by chance, and c) closely related strains or highly-conserved regions between species (e.g. 16S rRNA gene) are hard to disentangle for the same reason as a). Therefore, MAGs from short-reads are often fragmented due to those repetitive and hard to assemble regions.

Long reads result in higher confidence alignment, which can help to span repetitive genomic regions, and can even lead to circular assemblies (a full contiguous assembly of the whole chromosome) with minimal contamination. Additionally, long-reads from the PacBio platform have a higher per-base accuracy and can be used alone for genome assembly, as demonstrated with metaMDBG [21]. Other common tools for assembling long-read data include Flye [127], Canu [128], and HiCanu [191], all of which are widely used for assembling microbial genomes from PacBio or ONT data. In contrast, Oxford Nanopore reads—despite their longer lengths—still have a higher per-base error rate compared to Illumina reads [154]. For these datasets, hybrid assembly approaches are preferred, as they combine the long-range continuity of Nanopore reads with the high accuracy of short Illumina reads. Tools like Unicycler [300], MaSuRCA [326], and hybridSPAdes [9] are commonly used for such hybrid assemblies, allowing for more accurate reconstruction of metagenome-assembled genomes (MAGs). These approaches are currently the most effective way to generate high-quality MAGs from Nanopore-based sequencing [299].

2.6 Computational concepts and data structures

Bioinformatics emerged as a solution to the vast amounts of data and the unique challenges that come with analysing biological data. Many existing concepts from statistics and other fields were adapted to problems in the biological domain [318, 71], other methods were developed to work with sequencing data and other bioinformatics inputs [259, 192].

The performance of bioinformatics software is determined by two major factors. First, the chosen algorithm determines the runtime complexity and how the runtime grows with input size. For example, the standard algorithm for optimal local alignment, the Smith-Waterman algorithm [259], has a quadratic runtime complexity depending on query length n, denoted as $\mathcal{O}(n^2)$ in Big O notation. Big O notation gives an upper bound for how the time and space requirements grow as the input grows (Fig. 2.7). The recently published wavefront alignment algorithm (WFA) improves the algorithmic complexity from $\mathcal{O}(n^2)$ to $\mathcal{O}(ns)$, with s being the similarity between query and reference. This means that for identical sequences, the runtime scales linearly with input length but gradually gets worse with decreasing similarity and in the worst case scenario (no similarity) it exhibits the same runtime as the

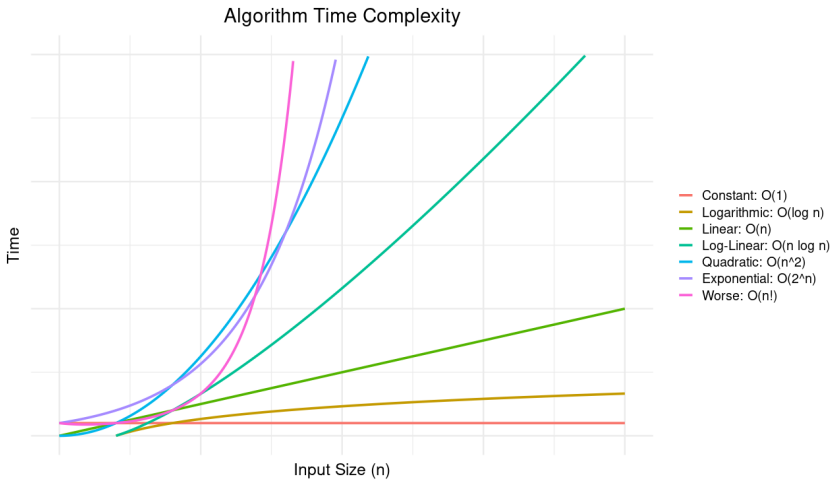

Figure 2.7: Visualization how the number of computations for algorithms with different runtime complexities scales with input size. Image generated using R.

Smith-Waterman algorithm.

The second factor in computational efficiency is about implementation specifics, such as choice of programming language, parallelization efficiency, choice of data structures, programming language, libraries, and memory efficiency. Simply put: the range between the slowest and the fastest implementation of the same algorithm can be vast [311, 129]. The choice of programming language defines how well the developer can access the underlying low-level resources such as memory, to optimize the number of operations necessary for computation, available libraries, and many more. C++, for example, is a compiled language offering fine-grained control over hardware and memory, allowing for optimizations that would not be possible in most other languages.

This is often crucial, as the way data is laid out in memory affects the performance significantly. Most CPUs have a hierarchical memory system divided into local storage and RAM, both of which are separate from the CPU, and further L3 cache, L2 cache, and L1 cache, all of which are integrated into the CPU [258]. The access times for memory increases in the order L1 cache, L2 cache, L3 cache, RAM, local storage, whereas the storage capacity decreases in reverse order. L1 cache, for example, has the fastest access time with ∼1-2 ns and usually has a capacity of 2-64 KB, while the main memory is ∼50-100 times slower but has a size of 16 GB in a typical personal computer. Hence, if data that is actively used by an algorithm can be laid out in consecutive blocks in memory, the number of accesses decreases and the algorithm is faster. Frequent jumps between far away memory locations slow down the program execution time. Different sorting algorithms, for example, have been shown to exhibit difference performances due to cache access patterns [132].

Sequence alignment

These performance differences can also be seen in popular bioinformatics algorithms such as Smith-Waterman for local alignment. Smith-Waterman is a dynamic programming algorithm for local sequence alignment that requires a matrix of size $m \times n$, with m being the size of the reference, and n being the size of the query (or read). The matrix is filled in by comparing nucleotides between query and reference and uses a set of parameters to score mismatches, start of gaps, and extension of gaps (practically indels). This leads to the algorithm having a runtime complexity of $\mathcal{O}(nm)$ which can be simplified to $\mathcal{O}(n^2)$ if $n > m$ is assumed. It has to be noted that Smith-Waterman results in an optimal alignment, meaning no better solution can be found given the input scoring parameters. Modern aligners use variations of this algorithm. Bowtie2 [136], for example, first uses a seeding approach where seeds of length 16 bp are extracted in distances of 10 bp and exact matches of those seeds are searched in the reference via their internal data structure, the FM-index (see next paragraph). Bowtie2 then uses a SIMD (single instruction multiple data) accelerated heuristic of Smith-Waterman around exact matching seeds, to complete the alignment between read and reference. SIMD is built into most modern CPUs and allows for doing the same computation for multiple data values in one CPU-cycle, as opposed to many cycles. SIMD can yield significant performance benefits, but is hard to implement and also requires multiple implementations to allow for execution on different CPU models. bwa-mem2 [169] implements a seed-and-extend algorithm around the FM-index as well ('mem' stands for maximum exact matches). It uses a banded Smith-Waterman alignment algorithm, which is restricted to only fill in the matrix in a certain distance around the diagonal - a heuristic which speeds up the process, but does not guarantee optimal alignment. Further, bwa-mem2 also heavily employs SIMD acceleration to speed up certain operations. Heuristics such as computing only parts of the matrix are often employed to speed up the algorithm [151].

Data structures in sequence alignment

The two most common data structures for read alignment and sequence search in bioinformatics are hash tables and the FM-index (Full-text index in Minute space) [71]. While the FM-index is a complex search structure with conceptual similarities to suffix-trees with great compression characteristics, a hash table is a conceptually simple data structure storing key-value pairs for fast lookup.

The FM-index is a compressed full-text index based on the burrows-wheeler transformation (BWT) . The burrows-wheeler transformation is a permutation of the input text where all possible rotations of a text are stored in a matrix which allows for better data compression. BWT has some similarities with suffix trees [295], as each rotation of the input is also the start of a suffix. The FM-index con-

ceptually combines BWT with suffix-arrays to yield a data structure that allows for high compression and fast querying. It is not surprising that the FM-index found application in bioinformatics, as many of its problems can be boiled down to text search and storage. Well known bioinformatic implementations of the FM-index are in the aforementioned alignment tools bwa [146] (Burrows-Wheeler aligner) and bowtie [137]. The FM-index has linear time complexity $\mathcal{O}(n)$ for queries, depending on the size n of the query. However, the FM-index has a poor spatial memory layout (cache), causing many random access patterns to the cache, which are known to be 'slow' [112]. Although sequence aligners like bowtie and bwa are fast, many more recently published aligners such as Minimap [145], accel-align [312], and strobealign [234], achieve higher speeds while using custom hash tables instead.

A hash table, often also called hash map or dictionary, is an organised data structures of unique key-value pairs that allows for constant time lookup of a value given its key. Constant time means that the complexity of a lookup is independent of both input size and the size of the data structure. In k-mer based tools like Kraken, hash tables are constructed in advance using a collection of reference sequences, with k-mers serving as keys and sequence IDs as corresponding values. When presented with a metagenome dataset for analysis, each read undergoes classification by searching for its k-mers within the pre-built hash table. The read is then assigned to the taxonomic identity associated with the most closely matching sequence based on k-mer hits.

The significance of hash tables in this context lies in their linear time complexity for lookup operations. This allows tools to utilise extensive reference databases without impacting runtime performance. However, the use of large reference databases does introduce memory usage considerations, as the hash table must be loaded entirely into memory. While publicly available hash table libraries for C++ exist, they often lack sufficient memory efficiency, thereby presenting a challenge for applications dealing with sizable datasets. In chapters 4 and 5 I will present two custom implementations of a hash table, tailored towards fast and efficient k-mer lookups.

Chapter 3

Benchmarking of metagenomic profilers with benchpro

3.1 Introduction

Metagenomic studies provide insight into microbial communities, their functions, and their interactions within diverse environments. Integral to studying these communities is to reconstruct their taxonomic composition to identify phenotypic changes associated with certain microbial taxa in a host-related setting. This necessitates accurate reconstruction of composition profiles by metagenomic profilers to reduce false predictions that can confound the statistical signal. The reasons for false predictions in reference-based profiling are manifold and include sequencing technology, sequencing errors, quality filtering, choice of reference, and underlying taxonomy.

With amplicon sequencing, for example, copy number variation of amplicons in a single organism can lead to the false detection of multiple taxa and create a false signal in the following analysis [89]. The distinction between these is important as, for example, incomplete reference databases or overly harsh quality filtering of low-abundant taxa prevents discovering associations relating potential keystone taxa. Ultimately, the challenge is to find a balance in quality filtering to remove the false signal without losing the true signal. Aside from this, limited computational resources, expertise, or time already dismiss certain tools that do not match the requirements in computational efficiency and usability. A firm understanding of performance and biases in taxonomic profilers is crucial to strike the balance between sensitivity, precision, and invested computational and human resources.

Choosing the right tool, however, is not easy, especially as benchmarks rarely go in depth into the root causes of false predictions and thus make it hard for the user to identify whether the tool is applicable to their specific research scenario. Consequently, an unbiased comparison of tool performance requires comprehensive independent benchmarks covering a broad range of applications. CAMI, short for critical assessment of metagenome interpretation, is an effort towards this by compar-

ing metagenomic tools for example for taxonomic profiling and taxonomic binning. Synthetic metagenomes as gold standard datasets simulate a wide range of microbiomes ranging from soil, over ocean to multiple host environments, allowing the authors to compare the performance of multiple profilers across a variety of use-cases. Using synthetic datasets as opposed to mock communities is crucial as the complexity of real-world samples is higher than what is cost- and labour-efficiently achievable with mock communities. In their comparisons, the CAMI team uses the NCBI taxonomy as a common denominator and as a result the benchmarks are obfuscated by the uncertainty of mapping between taxonomies and phylogenies. Tools like MetaPhlAn 4 and mOTUs3, take a different approach and come with proprietary databases with their own phylogeny and taxonomy, which has to be mapped into the NCBI space for benchmarking.

In the following, I will present benchpro, a set of tools for benchmarking metagenomic tools for species profiling as well as strain-resolved metagenomics. For benchmarking taxonomic profilers, benchpro allows for easy analysis of large datasets across multiple tools for taxonomic classification, impact of sample richness and abundance threshold filtering, and abundance prediction while supporting both the NCBI and GTDB Taxonomy. For benchmarking using the GTDB taxonomy, benchpro further provides analysis within the phylogenetic context of GTDB for each profile, allowing for disentangling of different causes for false predictions and their consequences.

For strain-level benchmarks, benchpro measures the monophyletic score, an ANI independent assessment of degree of monophyly in phylogenetic trees, cluster error in phylogenetic trees, and errors in MSAs for samples carrying the same strain. Further, benchpro compares predicted phylogenetic trees to a gold standard tree to assess similarity both with respect to topology and branch lengths.

3.1.1 Contribution of this Chapter

The scope of benchpro, in contrast to existing approaches [242, 177], is to not only compare tools on a performance level, but also to dissect sources of errors. This approach benefits both users and developers by providing a deep understanding of a tool's performance across different scenarios, thereby driving development and enabling users to make informed decisions. Benchpro specifically targets false predictions in taxonomic profiling, analyzing these errors within a shared phylogenetic context between predicted and gold-standard profile. Benchpro further provides universal benchmarks for strain-level tools that stratify strains across samples and use multiple sequence alignments and strain-resolved phylogenetic trees as intermediates, while existing benchmarks mostly focus on SNPs or use real-data with no gold-standard [8, 196].

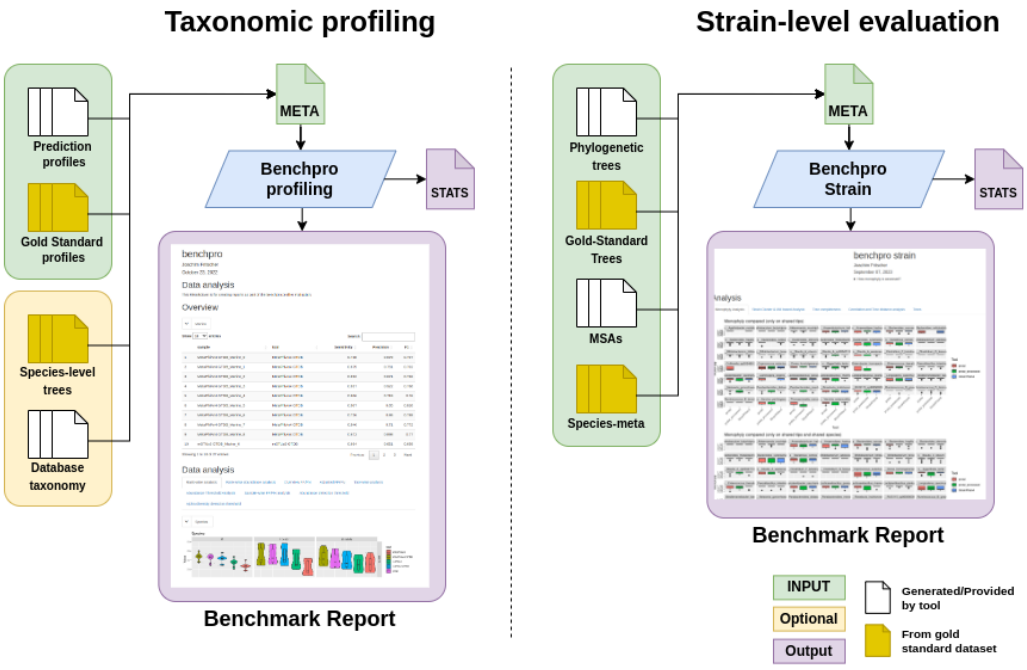

Figure 3.1: Benchpro's workflow listing required input, optional input, and output. Taxonomic profiling requires gold-standard and prediction profiles for each tool, dataset, and taxonomy. Optional files are phylogenetic tree for GTDB, and a list of all species in the database for each tool. File paths and optional data is configured in the META filer. For strain-level benchmarks, benchpro requires phylogenetic trees and MSAs for each tool and species across all datasets. Further, gold-standard phylogenetic trees and strain-resolved information for each dataset is required.

3.2 Methods

3.2.1 Workflow

Benchpro is a tool to benchmark metagenomics software for taxonomic profiling and strain-resolved analyses (Fig. 3.1). For evaluation of taxonomic profiling, benchpro takes a set of generated predicted profiles along with their gold standard profiles (for format see 3.2.3) to compute several metrics, such as F1-score, precision, sensitivity, or Bray-Curtis similarity (see Section 3.2.2 for details). Optionally, the user can provide gold-standard species-level trees and a list of species a tool can predict, to put predicted and gold standard profiles into a phylogenetic context, facilitating a more in-depth comparison. All files are organized within a meta-file, which is used as main input for benchpro (see Section 3.2.3). For strain-level evaluation, the tool-generated inputs are trees and MSAs for each species. Dataset inputs are gold-standard trees as well as a meta-sheet providing additional information about the strains per-species. For both species- and strain-level benchmarking, the output is a report generated with R markdown, which contains a visualization of the results - stratified by tool, dataset, taxonomic rank and metric.

3.2.2 Metrics

Evaluation of Binary Classification

Taxonomic profilers are binary classifiers, which means that for a given sample the taxa are put in one of two groups: present or absent. Benchmarking the binary classification performance assesses how well a profiler performs at predicting the presence or absence of taxa with respect to a gold standard profile. For each predicted profile and its corresponding gold standard, I compute true positives (TP) as the number of correctly predicted taxa, false positives (FP) as the number of incorrectly predicted as present taxa, and false negatives (FN) as the number of taxa incorrectly predicted as absent. Note, that there are no true negatives in this setting, as this would encompass all taxa absent from the gold-standard profile that a tool can predict. Some tools further output taxa labels for GTDB that indicate uncertainness, e.g. *'s__'* or *'Incongruent [g__Marinobacter]'*, and those are not counted as FP. From TP, FP, and FN, I can compute sensitivity, precision, and F1-score as follows.

$$\textbf{Sensitivity} = \frac{TP}{TP + FN} \tag{3.1}$$

$$\textbf{Precision} = \frac{TP}{TP + FP} \tag{3.2}$$

$$\textbf{F1} = \frac{2 \cdot TP}{2 \cdot TP + FP + FN} \tag{3.3}$$

Adjusted Evaluation of Binary Classification

False negative predictions are caused by either insufficient vertical coverage (sequencing depth), or because the database lacks coverage of the taxa. In the latter case, it is important whether there is a closely related species (with a different label) which is detected instead, or if the tool is unable to pick up the bacterial signal at all (Fig. 3.2). For clarification: reporting a different taxonomic label compared to a gold standard is generally a false prediction (especially in a clinical context). However, there are cases where labels do not match between gold-standard and predicted profile because the species are clustered at a different resolution or do no match between both taxonomies. In this case, false negatives (FN) that are absent from the tool database (undetectable) are often in close phylogenetic proximity of a false positive (FP) (see Fig. 3.2 C for clarification). This then indicates that there is a difference in phylogeny and species definition leading to a labeling issue, but this is not necessarily a biological error. To account for this, benchpro uses the GTDB taxonomy as a common taxonomy to identify those false positive, false negative pairs.

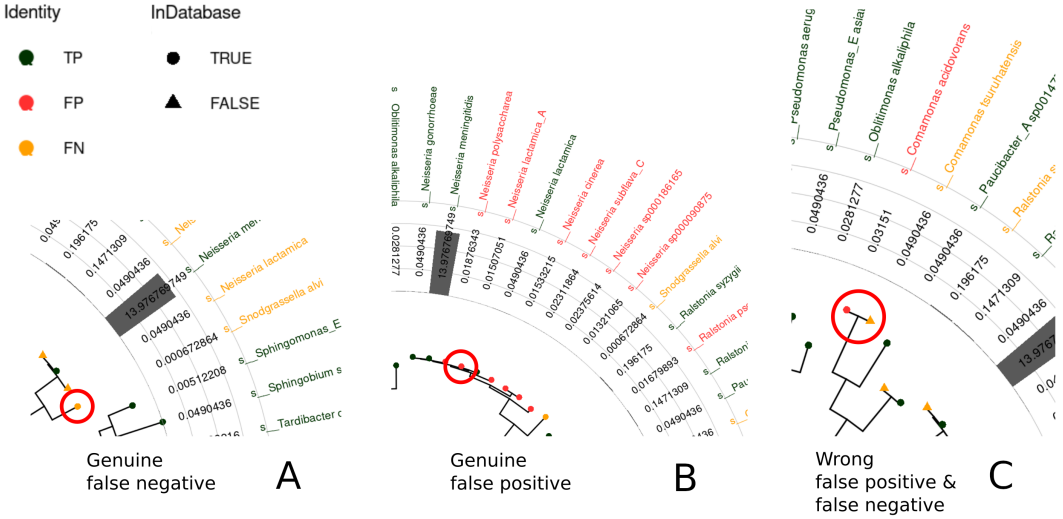

Figure 3.2: Tree visualization of TP, FP and FN of predicted and gold-standard profile. Each tip is a true positive, false positive, or false negative prediction. A, A genuine false negative, often because the abundance is below the detection threshold for the tool, but the taxon is covered by the database. B, A genuine false positive caused by false signals of nearby false positive predictions. C, A wrong false positive and false negative. The false negative taxon is not covered in the tool database, so its merged with its closest neighbouring false positive if the distance is under a threshold (distance of 0.04).

The algorithm starts with pairing each FP with their closest related FN based on phylogenetic distance that is not contained in the tool database. Next, the pairs are processed in order of phylogenetic distance, starting with the least distance pair. A pair is merged if the FN is not already merged with another FP and if the distance is below a certain threshold. Merged FN are relabeled as TP and merged FP are relabeled as false false positives (FFP). In the evaluation I selected a tree distance of 0.04 as a threshold for FP-FN-pairs to be considered for merging after assessing the pairwise distances of FN species absent from the tool database and their phylogenetic closest FP species within the same dataset (see A.3). Additionally, profilers sometimes output placeholder values such as *'s_ _ '* or *'Incongruent [g_ _ ...]'*. These taxa cannot possibly be TPs and reversely should not be counted as FP. Therefore, they are relabelled as TN and thus removed from all metrics. The adjusted counts are then used to recompute F1-score, sensitivity, and precision and are listed as a new rank 'Species Adj' below 'Species'.

Abundance Prediction

Beyond predicting presence and absence of taxa, profiling also entails predicting the (relative) abundance of each taxon in the community. For benchmarking, benchpro applies different metrics on the predicted and the gold standard abundances. I

compute Bray-Curtis similarity (BC), L2 distance (also called euclidean distance), and Pearson-Correlation (PC) on the predicted and gold-standard abundances. Taxa present in only one of the profiles (FP, FN) are set to 0 in the respective other. Additionally, to isolate the abundance prediction and not penalize for FP and FN predictions, the metrics L2 and PC are also computed on abundances for only TP predictions. This is indicated with a trailing '-TP' (PC-TP, L2-TP). The metrics compute as follows.

$$\textbf{Bray-Curtis similarity} = \frac{2 \cdot \sum min(P_i, G_i)}{\sum (P_i + G_i)} \tag{3.4}$$

$$\tag{3.5}$$

$$\textbf{L2} = \sqrt{\sum (P_i - G_i)^2} \tag{3.6}$$

$$\tag{3.7}$$

$$\textbf{Pearson-Correlation} = \frac{\sum (P_i - \overline{P})(G_i - \overline{G})}{\sqrt{\sum (P_i - \overline{P})^2 (G_i - \overline{G})^2}} \tag{3.8}$$

Computing monophyly scores as an ANI agnostic metric

Microbial strain-tracking is important for understanding colonization and transmission patterns within and between (host) environments. For two samples carrying the same strain, we can expect near perfect sequence identity (>99.9%) and, given a tree with all samples, that they form a monophyletic cluster. Strains tracked between samples, often have <100% ANI. Besides actual monoclonal nucleotide variation between genomes of the same strain, this is caused by errors in sample preparation, sequencing errors combined with low coverage, misaligned reads, as well as conspecific strains and results in below 100% ANI in pairwise sequence comparison of the same strain. To circumvent this problem, benchpro uses a monophyly score as an ANI agnostic benchmark (see Fig. 3.3). It captures the structure and grouping of samples rather than ANI values. With tools using different genomic regions (whole-genome vs. universal marker genes vs. species-specific marker genes) for comparison, ANI values can be misleading, which leads to structural metrics such as monophyly becoming more important. Note, that the informative value of monophyly is limited to datasets with sufficient strain-sharing between samples and increases with sample-size and number of closely related strains. Given a strain-level tree for a species and samples as leaves with known strain-identity, I compute the monophyly score for strain A as the proportion of samples carrying A and the total number of leaves under the LCA of all samples carrying A as shown in Fig. 3.16.

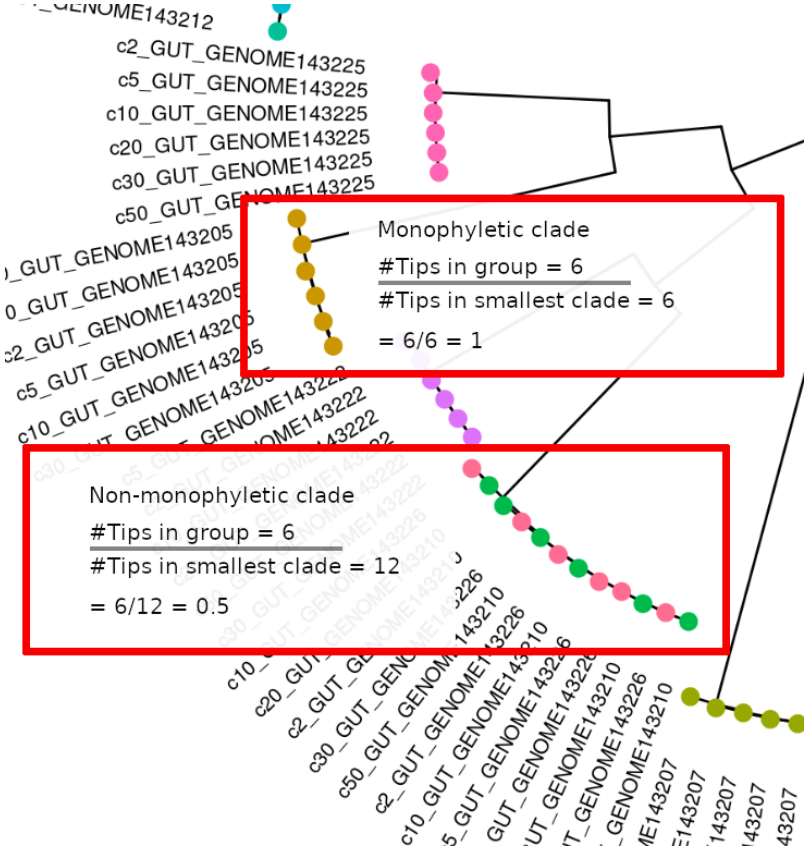

Figure 3.3: Here, the monophyly score describes the purity of a phyletic clade. It is computed by dividing the number of samples carrying strain A by the number of leaves in the smallest clade containing all samples carrying strain A. The monophyly score is constrained to be between 0 and 1, with one being perfect monophyly score.

Computing Maximum Cluster Error in Phylogenetic Trees

Strains in a phylogenetic tree should form monophyletic clades. If two different strains have high similarity it is often not possible to maintain monophyly. Given strain A and strain B have indistinguishable high ANI. In this case, the pairwise phylogenetic distances of samples within strain A and between strain A and B should be close to zero (Fig. 3.18). To quantify this error in a phylogenetic tree for a given strain, I introduce the metric Max Cluster Error (MCE) . The following formula explains MCE. $PairwiseWithin(A)$ are all pairwise distances between all samples carrying Strain A and similarly $PairwiseBetween(A)$ are all pairwise distances between a sample carrying strain A and all other samples outside of strain A.

$$\mathbf{MCE(A)} = -1 \cdot (max(PairwiseWithin(A)) - min(PairwiseBetween(A))) \tag{3.9}$$

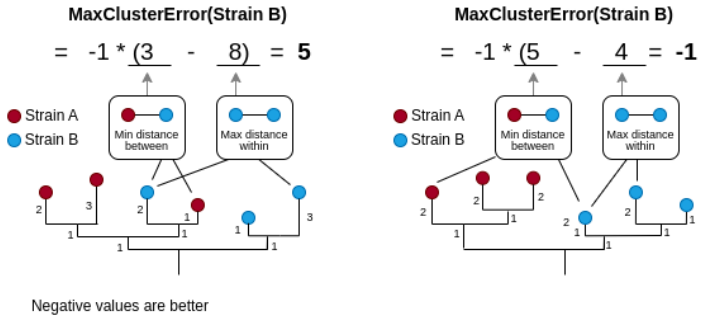

Figure 3.4: MaxClusterError (MCE) compares pairwise phylogenetic distances within a strain to pairwise phylogenetic distances to other strains. High values of MCE indicate that two strains are not separated in the tree. Further, MCE quantifies large topological incongruities in the phylogenetic tree by reporting strains with high pairwise distances within its members. MaxClusterError (MCE) is a tree-dependent, subjective measure that reflects the relative placement of strains in a phylogenetic tree. While its absolute values may vary with tree topology, MCE is a useful comparative measure for assessing the clustering performance of different tools. When computing MCE for strain A, all other strains are considered for the pairwise between distances.

Computing the Error in MSAs

As with the monophyly score, knowing the strain-identity of each sequence in an MSA and having multiple samples carrying the same strain, we can both assess the error rate within a group of sequences representing the same strain and its information content across all sequences. The error rate is computed as all positions in a group of samples carrying the same strain where two different bases (ignoring '-' and 'N') occur divided by the number of positions with at least a coverage of two. The information content, is the total number of positions with more than one nucleotide per position, again ignoring '-' and 'N'. Intuitively, comparing sequences in the MSA representing the same strain, I expect 100% sequence identity and any deviation indicates an error. Similarly, to delineate strains within an MSA, I require sufficient genomic variation to tell two strains apart. If the within group error rate exceeds the information content in the MSA, it is not possible to robustly reconstruct strain-phylogenies.

Benchmarking Phylogenetic Accuracy

To measure the quality of per-species tree reconstruction, benchpro compares each tree to its Roary [**page_Roary_2015**] gold standard constructed from all utilized strains of a given species (see Section 3.2.5). Roary is a bioinformatics tool designed for pan-genome analysis of prokaryotic genomes. It identifies both accessory genes and core genes—those shared among a set of user-defined conspecific genomes—to reconstruct their phylogenetic relationships. While each predicted tree has as many leaves as samples, the gold standard tree only has one leaf per strain (genome) in

the dataset. For comparison, each group of leaves (samples) in the predicted tree carrying the same strain is removed and its LCA is kept as representative for this strain. Predicted and gold standard trees can be compared either with respect to edge lengths or tree topology. Benchpro uses the phangorn (v2.12.1) package [238] to compute the topological distance metrics, normalized Robinson-Foulds distance (RFnorm) [228] and Steel and Penny distance (SP) [266], and further the weighted normalized Robinson-Foulds distance (wRFnorm) [228], weighted Steele and Penny distance (wSP) and Kuhner & Felsenstein distance (KF) [161], which additionally account for edge lengths.

Given two phylogenetic trees, the Robinson-Foulds (RF) distance counts the number of branch bi-partitions that are not shared between two trees, hence the maximum distance is the sum of the number of bi-partions of both phylogenetic trees. The normalized distance is computed by dividing the Robinson-Foulds (RF) distance by the maximum achievable distance, resulting in a value constrained between 0 and 1. The weighted RF distance incorporates branch lengths by incrementing the score based on the absolute difference in branch lengths between two trees and for each bi-partition, instead of simple counting [228]. Similarly, the Kuhner-Felsenstein (KF) distance sums up the squared differences instead of absolute values, giving more weight to longer branches [130]. The path distance as proposed by Steel and Penny [266], first computes all leaf-to-leaf difference for both trees, and then sums up the absolute differences for each path between trees.

3.2.3 Input Format for Benchpro

Gold-standard and Predicted Profiles

Benchpro accepts taxonomic profiles in CAMI format [242] and in tab-delimited files. With both formats, the two columns of interest are lineage and abundance, the column number of which can be specified in the map file (see Section 3.2.3). Profiles must either specify taxa only for a single rank or specify all taxa. If the profile contains only taxa of a single rank, e.g. species, the abundances for all higher ranks will be inferred with help of the taxonomic lineage. The taxonomic lineage can be either pipe-delimited (2|23|123...) or semicolon-delimited (d__Bacteria;p__Firmicutes;...") and names are either integers or strings with or without the GTDB tax prefix 's__' (s for species).

Meta Files for Organizing Input

To provide a simplified framework for complex benchmarks with multiple samples, data sets, and tools, benchpro takes map files for benchmarking taxonomic profiling and strain-resolution. Map files are tabular files (.xlsx or .tsv), organizing information on utilized tools, samples, paths to profiles, trees, MSAs, and other files. In the map file for benchmarking taxonomic profiling, there is an entry (row) for each

Table 3.1: Map file required for benchmarking taxonomic profiling. All columns are mandatory.

Column	Description
ID	Unique identifier in the whole table, e.g. protal_sample1
sample	Sample name
Dataset	Dataset name
Tool	Tool name
Profile	Path to predicted profile
ProfileColumns	Columns in the profile with the format id|gtdb_lineage|abundance . (default 0|1|2, indicating that lineage is in column 1 and abundance in column 2, starting with column 0. If the id is missing, this can be indicated with an X, e.g. X|0|1).
GoldStd	Path to gold standard profile
GoldStdColumns	Same as "ProfileColumns" for GoldStd.
GoldStdTree	Tree for the utilized taxonomy. E.g. GTDB r214 for protal or GTDB r207 for MetaPhlAn 4. Can be set for each sample individually.

predicted profile. Each entry specifies which tool generated the profile for which sample and where to find the predicted and the gold-standard profile. The exact specifications are described in Table 3.2.

In contrast to per-sample benchmarks for taxonomic profiling, the strain-level evaluation revolves around per-species MSAs and phylogenetic trees. In the strain-level map file each entry (row) contains information about tool, dataset, species, path to the tree, path to predicted reconstructed tree including information on the tool, the tree construction tool and the species, as well as where to find the newick tree, the MSA and a species meta-file. Trees must be in newick format and MSAs in fasta format.

For a species present across multiple samples, the species-meta-file contains information about the vertical coverage and the specific strain contained in each sample. This information must be supplied with the gold standard dataset and cannot be derived by the user unless the user has created the dataset. With this information, benchpro can deduce which tree tips should form monophyletic groups.

3.2.4 Output Format

The output of benchpro is a report (.html) generated with R Markdown. The report contains plots for different metrics, stratified by dataset and tool. For species-level benchmarks the report can be generated in a single command from the command-line. The strain-level scripts are currently still in development.

Table 3.2: Benchpro map file for strain-level evaluation. Rows are samples and columns contain additional information on file paths and others. The meta-file can be either a .xlsx or as tab-delimited text file. Provided to benchpro with *'–meta META_FILE'*

Column	Description
Name	Unique name of tree
Species	Species name, e.g. s__Agathobacter_rectalis
Tool	Tool name (gold-std in case this entry is the gold-standard tree for this species)
TreeTool	Tool for tree creation (e.g. IQ-tree or raxML)
Tree	Path to newick tree file
MSA	Path to MSA file
AnalysisMSA	Path to MSA analysis if it does not exist, it will be generated by benchpro from the MSA supplied in column MSA
Meta	Path to species meta-file, containing information which strains each sample contains.
SpeciesTree	Path to species level tree. Can be set to NA
AvailableSpecies	Path to file containing a line for each species that can be detected. Can be set to NA

Table 3.3: Benchpro species meta-file for strain-level profiling. Rows are samples and columns contain additional information on file paths and others. The meta-file can be either a .xlsx or a tab-delimited text file. Provided to benchpro with *'–meta META_FILE'*

Column	Description
ID	Matches sample name in strain map and in trees
genome	Name of genome used to simulate reads
species	Species name
Coverage	Vertical coverage of reads for this species in this sample
Path	File path to genome fasta

3.2.5 Datasets

Taxonomic Profiling

For species-level benchmarking I used the datasets "Toy Human Microbiome Project Dataset" (CAMI Human), "Toy Mouse Gut Dataset" (CAMI Mouse), and "Marine" (CAMI Marine) from the 2nd CAMI challenge (https://data.cami-challenge.org/). The gold standard profiles were converted to GTDB r207 and GTDB r214 by running GTDB-tk v2.32 on the source genomes and using the abundance values provided in the *'abundance<sampleid>.tsv'* files for CAMI Human and the *'distributions/distribution_<sampleid>.txt'* files for CAMI Mouse and *'genome_to_id.tsv'* to map the genome fasta files with their respective genome identifier.

Strain-evaluation

To benchmark the strain-level performance by measuring monophyly and within-group error in MSAs, the dataset needed to resemble time series data where some samples share the same strain and other samples contain different strains at a varying rate of relatedness. As a source for genomes I chose humgut, a comprehensive collection of 289,232 genomes and MAGs common to the human gut [99]. To select a set of species with a variety of high quality genomes available, I filtered the humgut database for isolate genomes only and selected 50 species with 15-60 isolate genomes each. As the existing taxonomic annotation was done with the older GTDB r95 version, I re-annotated the pool of 1428 isolate genomes with GTDB-tk (v2.32, default parameters) and GTDB r214. Due to changes in taxonomic assignments, the re-annotated genomes now span 67 species in version r214. After this process I removed species that had less than 15 genomes available and was left with 1348 genomes spanning 46 species. These genomes were used as the basis to simulate 200 samples, each containing a single strain per species across all 46 species. Although taken from a collection of bacteria commonly found in the human gut, the genome selection was not designed to mimic the gut environment by representing a commonly found diversity. Instead, the goal was to pick species which had a sufficient number of isolate genomes available to avoid the often intransparent quality issues regarding chimeric contigs and bins found in MAGs. All isolate genomes names within humgut with their GTDB r214 species classification used can be found in Table A.3. Table A.3 contains information on mean vertical coverage and standard deviation per species across samples.

For each sample and species the genome is randomly selected from the pool and the vertical abundance values are normalized to be between 1 and 50 following a negative binomial distribution. The paired reads with 2x150bp are simulated with art_illumina (v2.5.8) [102] using the parameters *'ss HS25 -i <input.fna> -p -l 150 -f <coverage> -m 200 -s 10 -o <output_prefix>'*. The core genome and gold-standard MSA for each species was constructed with Roary (v3.13.0, default parameters) **[page_Roary_2015]** and *'–mafft'* [118] on all prokka (v1.14.6, default parameters from Roary) [244] gene predictions from source genomes within that species. iqtree (v2.2.0.3) [180] was used to construct the gold-standard phylogeny of all strains from Roary's MSAs within a species with *'-s <MSA> -fast -m GTR'*. This dataset is used in Section 3.3.3 and Section 5.3.5.

3.2.6 Metagenomics

Taxonomic profiling

MetaPhlAn 4 (v4.0.2) was used to profile the CAMI datasets and SPECIES46 datasets with database *'mpa_vJan21_CHOCOPhlAnSGB_202103'* using default parameters. MetaPhLAn 4 uses bowtie2 with *'–sensitive'* to align the reads against

species-specific markers. mOTUs3 (v3.0.1) uses bwa2 for alignment against universal marker genes of a custom database . Kraken2 (v2.1.3) was used with the standard RefSeq database covering bacteria, archaea, viruses, and the human genome[1]. Bracken (v2.9) was used for abundance estimation with a read threshold (*'-t'*) of 450. Kraken2+Bracken output was mapped from NCBI to GTDB with a custom mapping file created from the GTDB r214 metadata files [2]. The species-level NCBI taxonomic identifier (Column 74 in the metadata file) was mapped to the GTDB lineage (Column 17) where GTDB and NCBI species names matched. If this was not possible, the NCBI taxonomic identifier was mapped to the GTDB lineage with the most occurrences. In case of a tie, an arbitrary GTDB lineage was selected out of the tying members. MetaPhlAn 4 was mapped to GTDB using the provided script *'sgb_to_gtdb_profile.py'* and NCBI results were obtained with the parameter *'–CAMI_format_output'*. mOTUs3 was mapped to GTDB using a custom script and their mapping file *'mOTUs_3.0.1_GTDB_tax.tsv'* and NCBI results were obtained with the parameter *'-C parenthesis'*.

Strain-level profiling

StrainPhlAn 4 (v4.0.2) was run with default parameters by first extracting the marker regions for each sample with *'sample2markers.py'* from the read alignments, then extracting the marker regions from the database as reference with *'extract_markers.py'* from all SGBs (MetaPhlAn 4 species) found in the samples, and finally running the main command StrainPhlAn 4 with *'–marker_in_n_samples 20'* to get per species MSAs. Protal was used with default parameters to get taxonomic profiles. IQ-Tree was used to infer phylogenetic trees from multiple sequence alignments with *'iqtree -s <MSA> -fast -m GTR'*.

3.2.7 Runtime and memory benchmark

For the runtime comparison, all tools are run on the CAMI Airways dataset with 10 samples of 2x5GB uncompressed paired-end fastq files. Included in the comparison are Kraken2+Bracken, mOTUs3, MetaPhlAn 4, and StrainPhlAn 4. The tools were run with the parameters described in the previous section 3.2.6. All tools were run with 16 threads and the benchmark was conducted on AMD EPYC 9654 96-Core Processor @ 3.70GHz with a separate local SSD storage. All reads were copied over to the local SSD storage to minimize the effect of other processes on the cluster using IO. The databases were not copied over, but each program was run three times and the best run was taken as final result. After the first run, the utilized database is cached, which leads to loading time being faster for the two subsequent runs. StrainPhlAn 4 was only run once, but does not load a large database as opposed to

[1] https://benlangmead.github.io/aws-indexes/k2, from 1/12/2024
[2] *'ar53_metadata_r214.tsv'* and *'bac120_metadata_r214.tsv'*

the other tools. Tree construction is not included in the benchmark for StrainPhlAn 4, as it is the user's choice which tool to use.

3.2.8 Implementation Details

Benchpro's implementation is comprised of both data generation and visualization. Statistics for taxonomic classification, from predicted and gold standard profiles, are implemented in Python 3.10. Visualisation and R Markdown generation is implemented in R. Strain-level analyses are implemented in R and R Markdown. Trees are read and handled with the ape package. Tree comparison is done with the phangorn package as described in 3.2.2. All plots are generated with ggplot2.

- R version 4.5.0,
- Other packages: ape 5.8-1, BiocManager 1.30.25, cowplot 1.1.3, dplyr 1.1.4, DT 0.33, forcats 1.0.0, ggExtra 0.10.1, ggplot2 3.5.2, ggpubr 0.6.0, ggtree 3.16.0, ggtreeExtra 1.18.0, knitr 1.50, lattice 0.22-5, maps 3.4.2.1, PerformanceAnalytics 2.0.8, permute 0.9-7, phangorn 2.12.1, phytools 2.4-4, plotly 4.10.4, quantmod 0.4.27, reshape2 1.4.4, stringr 1.5.1, tibble 3.2.1, tidyquant 1.0.11, tidyr 1.3.1, TTR 0.24.4, vegan 2.6-10, xts 0.14.1, zoo 1.8-14

3.3 Results

3.3.1 Overview

The following results section is divided into two parts: benchmarking taxonomic profiling (Fig. 3.5) down to the species level and separate strain-level benchmarking (Fig. 3.6).

Taxonomic profiling is evaluated using MetaPhlAn 4, mOTUs3, and Kraken2+Bracken on the CAMI datasets (see Section 3.2.5; Fig. 3.5). First, species-level classification performance is assessed in terms of F1-score, sensitivity, and precision (Fig. 3.5 A). These metrics are then re-evaluated after applying an abundance threshold to filter out low-abundance FP predictions, testing whether such post-filtering improves overall performance (Fig. 3.5 A). Genus-level performance is also examined using the same metrics (Fig. 3.5 B). Finally, to assess how accurately each tool estimates species-level abundances, predicted and gold-standard profiles are compared using L2 error, Bray-Curtis dissimilarity, and Pearson correlation (Fig. 3.5 C).

Next, the species-level results are re-examined within a phylogenetic framework using the GTDB r207 and r214 reference trees, with a particular focus on FNs that are caused by taxa absent from some tool databases. For each tool, FNs

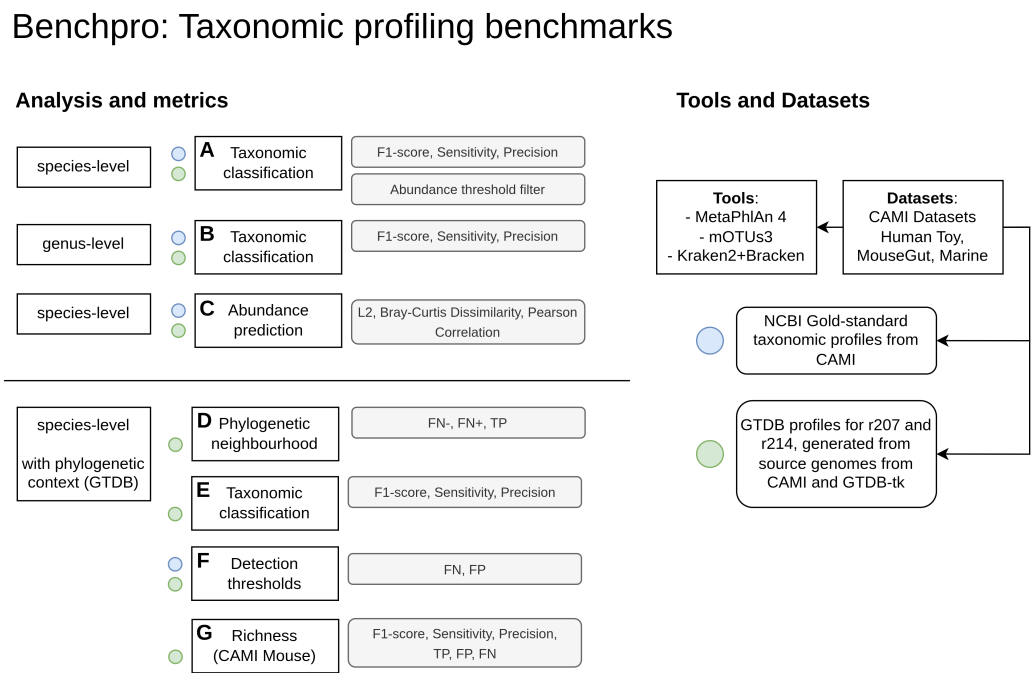

Figure 3.5: Overview over datasets, tools, and analyses regarding taxonomic profiling in this chapter.

are classified as FN- or FN+, depending on whether the species is present in the respective database. False positives (FPs) are then analysed in relation to their closest phylogenetic neighbours among FN-, FN+, and true positives (TPs), as well as their relative abundances, to explore potential sources of misclassification (Fig. 3.5 D). Subsequently, FP-FN- pairs are reassessed to derive adjusted species-level performance scores (Fig. 3.5 E). Detection thresholds are also examined across tools and datasets to (a) estimate the minimum abundance at which true positives are reliably detected, and (b) quantify high-abundance FNs that are missed (Fig. 3.5 F). Finally, the mouse dataset is used to assess how species richness in a sample impacts tool performance (Fig. 3.5 G).

The strain-level benchmarks assesses StrainPhlAn 4 on a custom dataset spanning 200 samples, each containing a single strain per 46 species. The aim is to assess StrainPhlAn 4's (and later protal in chapter 5) ability to sensitively and accurately resolve strains for species present in multiple samples. First, the sample and strain sensitivity is assessed and how it is affected by the vertical coverage of strains (Fig. 3.6 A). Next, monophyly score is used as a metric to assess how well samples with the same strain cluster together in the tree (Fig. 3.6 B). After this we look at monophyly and sensitivity together to see if they affect each other (Fig. 3.6 C). MCE is used to further quantify errors in non-monophyletically resolved strains (Fig. 3.6 D). Next, we assess whether errors in the MSAs support errors in the tree (Fig. 3.6 E). Lastly, per-species phylogenetic trees are compared to reference trees generated using Roary.

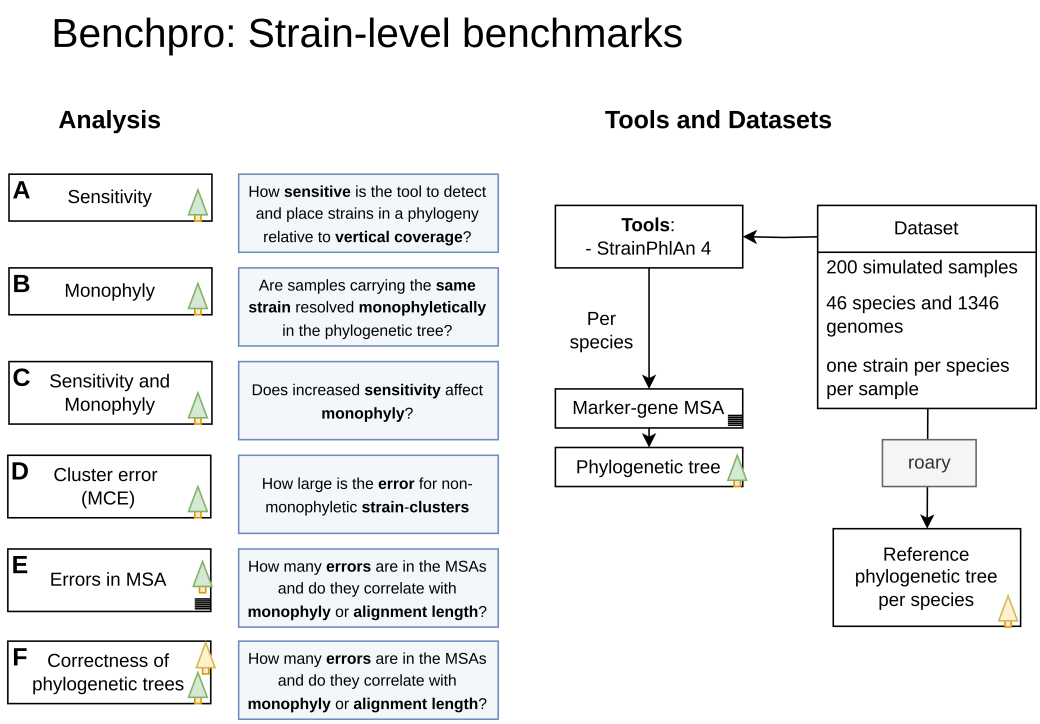

Figure 3.6: Overview over datasets, tools, and analyses regarding strain-level benchmarks in this chapter.

3.3.2 Taxonomic Profiling

Benchpro is a suite of tools designed for a comprehensive assessment of taxonomic profilers and strain-resolved tools. The following section shows a comparison between three popular taxonomic profilers, MetaPhlAn 4 [26], mOTUs3 [232], and Kraken2+Bracken [307, 159]. This evaluation sheds new light on the tools' profiling performance regarding false positives and false negatives, abundance prediction accuracy, abundance detection limits, and effect of abundance-based post-filtering. All tools were run on the CAMI datasets Human Microbiome, Mouse Gut, and Marine, and are evaluated using both the NCBI and GTDB taxonomy (For methods, see 3.2). Further, this comparison is not only between tools, but also between alignment-based vs. alignment-free approaches (MetaPhlAn 4 and mOTUs3 vs. Kraken2+Bracken) and a comparison between methods using species-specific marker genes, universal marker genes, and whole genomes as reference (MetaPhlAn 4, mOTUs3, and Kraken2+Bracken, respectively).

Fig. 3.7 shows the species-level benchmarking results between MetaPhlAn 4, mOTUs3, and Kraken2+Bracken using both the NCBI and GTDB taxonomy. For F1-score for species level on all datasets, MetaPhlAn 4 performs best with a mean of 0.927±0.04 and 0.852±0.057, for GTDB and NCBI respectively. mOTUs3 comes second with 0.841±0.055 for GTDB and 0.773±0.062 for NCBI. Last is Kraken2+Bracken with 0.427±0.157 and 0.451±0.15. As both MetaPhlAn 4

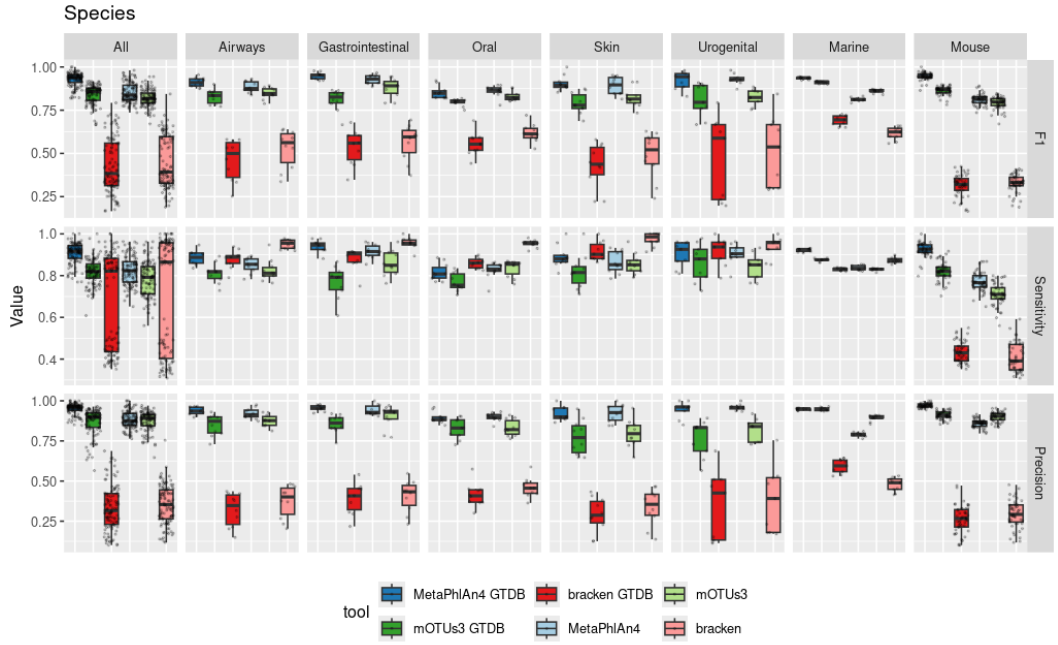

Figure 3.7: Species level profiling performance across samples, environments, and tools measured with F1-score, precision, and sensitivity. Each box plots represents a tool and each data point is a sample. Top to bottom, the panels display F1-Score, precision, and sensitivity and the column panels stratify between different environments with the left row being the summary across all environments.

NCBI and mOTUs3 NCBI have comparable sensitivity levels (0.823±0.068 and 0.782±0.084, respectively), the difference in F1-score is mostly due to the inferior precision of mOTUs3 (0.885±0.055 vs. 0.772±0.077 for MetaPhlAn 4 vs. mOTUs3). The GTDB phylogenetic context uncovers that FP predictions in mOTUs3 are mostly artifacts of surrounding close-by (and often highly abundant) true positives. An example of this signal can be seen in mOTUs3 GTDB profile for sample Marine_3. *s__Moritella viscosa* is correctly predicted as present, albeit with a higher abundance (0.31% true abundance, 0.88% predicted abundance). In addition, two FP species of the same genus—*s__Moritella sp001574435* and *s__Moritella marina*—and in the close phylogenetic neighbourhood are falsely predicted as present with lower abundances of 0.09% and 0.13%, respectively (see Fig. A.2).

Both MetaPhlAn 4 and mOTUs3 have a similar F1-score between GTDB and NCBI and for MetaPhLAn4, sensitivity and precision are also similar between taxonomies. However, mOTUs3 GTDB has a lower precision (0.87±0.082 for GTDB, 0.874±0.061 for NCBI) and higher sensitivity (0.818±0.062 for GTDB, 0.782±0.084 for NCBI). MetaPhlAn 4 GTDB performs better than MetaPhLAn 4 across all metrics (F1: 0.928±0.04 vs. 0.852±0.057, precision: 0.95±0.035 vs. 0.885±0.055, sensitivity: 0.907±0.05 vs. 0.823±0.068). Kraken2+Bracken GTDB performs worse than the other tools with an F1-score of 0.427±0.157, precision of 0.331±0.138, and sensitivity of 0.676±0.23. Kraken2+Bracken using NCBI performs better than with

GTDB and has a F1 score of 0.451±0.15, a sensitivity of 0.698±0.272, and a precision of 0.354±0.119. The performance difference between taxonomies is likely due to Kraken2+Bracken's output being native to NCBI - mapping to GTDB adds noise. This is not the case for the other tools, as both MetaPhlAn 4 and mOTUs3 have their own internal taxonomy and phylogeny and thus are neither native within NCBI nor GTDB. MetaPhlAn 4's better performance on GTDB suggest that its internal species clustering aligns more with GTDB than with NCBI. Kraken2+Bracken's high sensitivity is partly due to its whole genome approach, which allows for picking up a signal from the whole range of the genome, as opposed to MetaPhlAn 4's and mOTUs3's marker gene based approach, which greatly limits the amount of informative reads. Additionally, Kraken2+Bracken has no default filter for FP reads and thus gains its sensitivity at the cost of precision seen in the difference of their medians of ∼0.5 across all datasets.

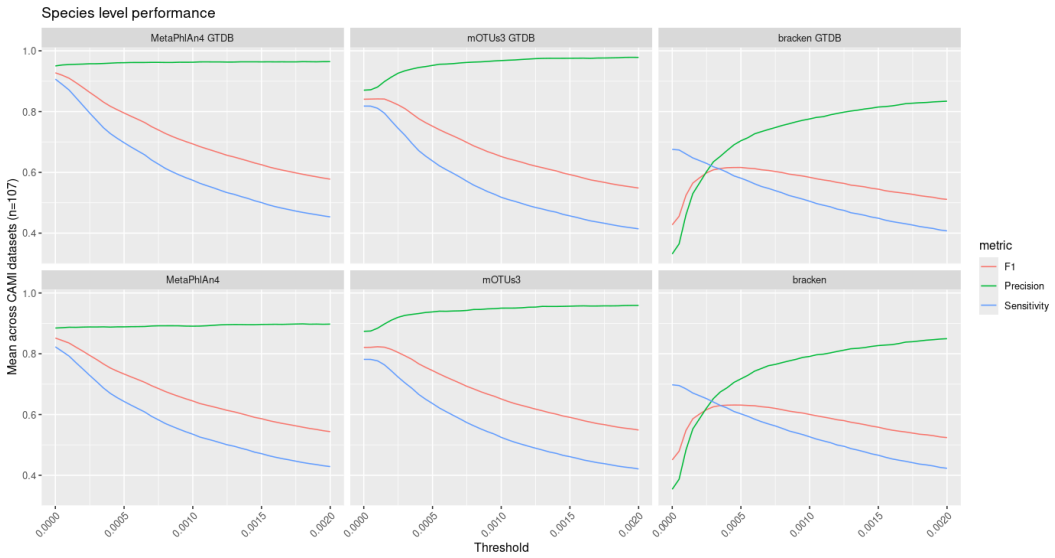

Figure 3.8: Change of F1-score, precision, and sensitivity on species-level on the y-axis across tools when applying different abundance thresholds for the predicted profiles on the x-axis. Each panel shows the performance of one tool. The y-axis shows the mean of F1-score, precision, and sensitivity and is calculated for each tool over 107 samples across all datasets (CAMI Human, Mouse, Marine).

To investigate whether applying an abundance threshold can increase the profiling performance on species-level, a sliding threshold was applied to recompute sensitivity, precision, and F1-score (Fig. 3.8). Kraken2+Bracken benefits the most from this threshold with a maximum mean F1-score of ∼0.85 and ∼0.8 for NCBI and GTDB respectively between a threshold of 0.0005 and 0.001, marking an increase of 0.3 from the unfiltered profiles. mOTUs3 benefits as well, with an increase of ∼0.05 for both taxonomies peaking at a threshold of ∼0.0003. MetaPhlAn 4 shows no increase in F1-score and has its best performance with no filter. It should be noted that this benchmark explores the best possible performance with an abundance threshold

determined with knowing the gold-standard. This is not achievable by the user, as the best abundance filtering threshold depends on factors like sequencing depth and complexity of the community. Therefore, there is no way to determine which threshold is the best on real data. MetaPhlAn 4 is the most user-friendly in this sense as the default parameters yield a good performance. It is also noteworthy that MetaPhlAn 4 is the most precise out of the three tools at any threshold.

Genus-level profiling performance

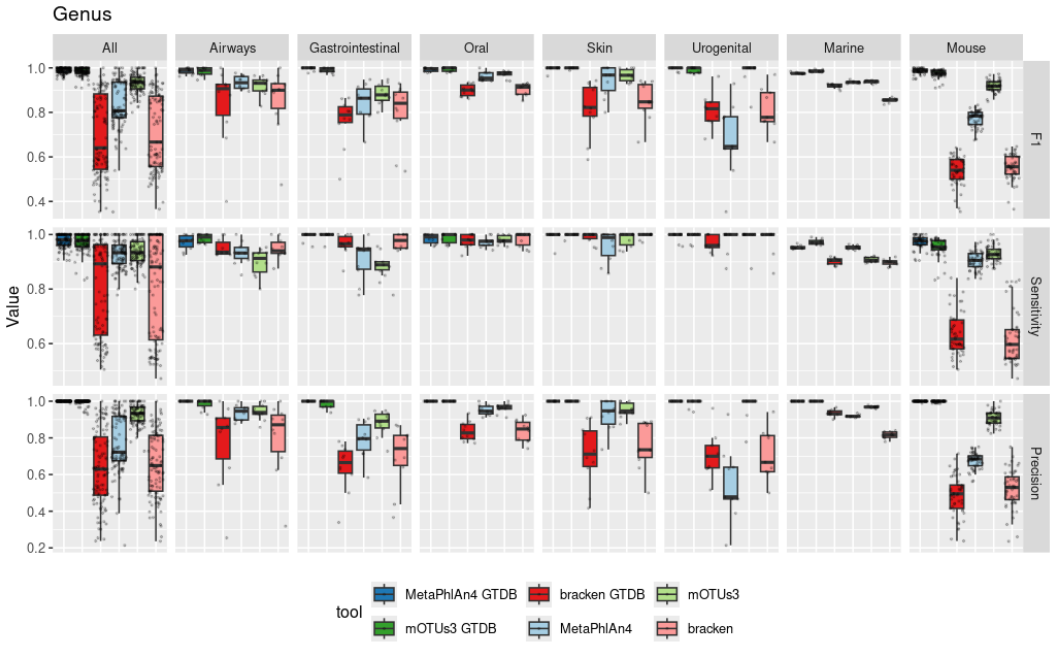

Figure 3.9: Genus level profiling performance across samples, environments, and tools measured with F1-score, precision, and sensitivity. Each boxplot represents a taxonomic profiler and each data point is a sample. From top to bottom, the row panels are F1-Score, precision, and sensitivity and the column panels stratify between different environments with the left column being the summary across all environments.

On genus-level and across all datasets, both MetaPhlAn 4 GTDB and mOTUs3 GTDB show an almost perfect F1-Score with 0.988±0.013 and 0.983±0.017, respectively (Fig. 3.9). With NCBI, the scores are significantly lower at 0.834±0.11 for MetaPhLAn 4 and 0.916±0.051 for mOTUs3. While the precision for both tools is almost 1 (1±0.001 for MetaPhlAn 4 and 0.995±0.014 for mOTUs3), they miss some genera (mean FN of 27.111±3.79 and 50.889±5.533 for MetaPhlAn 4 GTDB and MetaPhlAn 4 NCBI, and 43.556±5.681 and 53.222±6.078 for mOTUs3 GTDB and mOTUs3 NCBI), likely due to genera that are either only represented by low abundant species that are also not detected, or by genera that are missing or labelled differently in the respective tool databases (sensitivity of 0.977±0.024 for MetaPhLAn 4 and 0.971±0.032 for mOTUs3). Since the focus is on species-level profiling

and below, I did not investigate this further. Kraken2+Bracken GTDB is inferior in all metrics with an F1-score of 0.701±0.178, a precision of 0.637±0.2, and a sensitivity of 0.81±0.172. The value is about the same for NCBI with an F1-score of 0.711±0.17, a precision of 0.655±0.177, and a sensitivity of 0.81±0.172. These scores are not only due to the large number of FPs (mean of 31.533±26.123 for GTDB and 19.215±14.414 for NCBI) as otherwise the sensitivity would be higher.

Benchmarking the reconstruction of species-level relative abundances

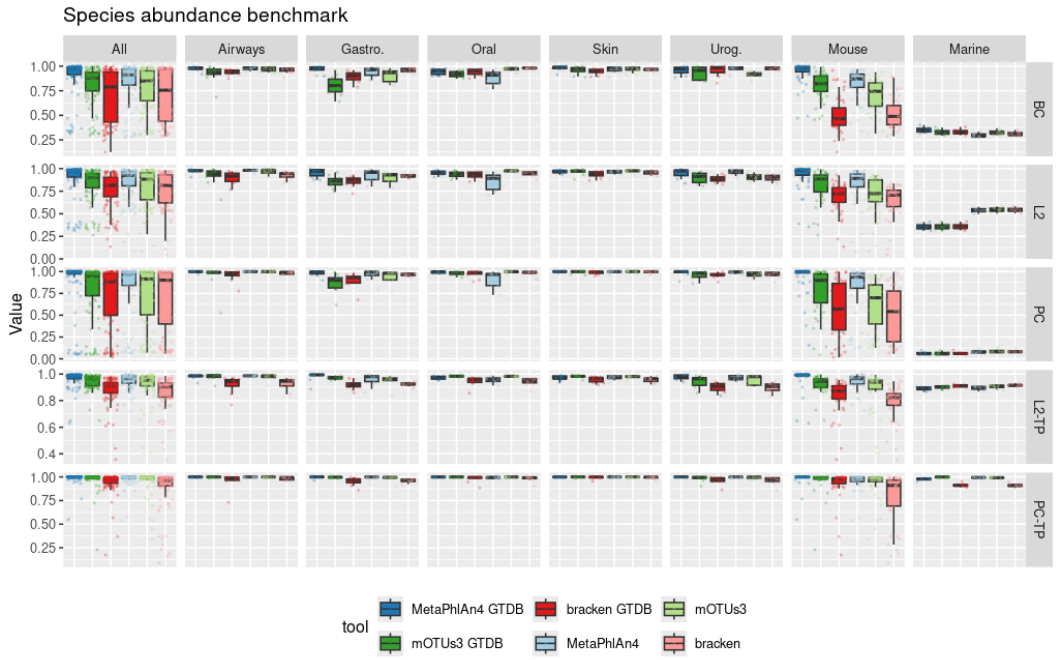

Figure 3.10: Benchmark to test the ability to correctly reconstruct microbial abundances. The column panels show different datasets, with the leftmost column being the summary across all; the row panels stratify different metrics. Each plot shows the performance on the y-axis stratified for all tools on the x-axis. Each data point in the boxplots represents a sample. The metrics displayed are Bray-Curtis similarity (BC), 1 - L2-error (L2) and Pearson Correlation (PC). The suffix '-TP' indicates that the metric has only been computed on TP abundances (see Section 3.2.2 for details).

Correctly predicting presence and absence of taxa is only one part of taxonomic profiling. Predicting the abundance values is equally important, as downstream statistical analyses rely on accurate abundance values for differential abundance analysis. I used two different ways to assess the performance of abundance prediction. The metrics Bray-Curtis similarity (BC), L2, and Pearson Correlation (PC) penalize not only for deviating abundance values, but also for missing or incorrect species. L2-TP and PC-TP only focus on TP taxa and thus do not penalize based on the classification performance (see Section 3.2.2 for details). Further, all of the abundance prediction metrics were only computed on unadjusted values.

MetaPhLAn 4 GTDB had the highest score across all tools and all metrics with a mean BC similarity of 0.944±0.136, followed by mOTUs3 GTDB with 0.929±0.134, and Kraken2+Bracken with 0.835±0.177. This order maintains for all other metrics. Out of all datasets, all tools showed the lowest accuracy at reconstructing abundances for the Marine dataset, with a mean BC between 0.295±0.02 and 0.349±0.027. However, for the metrics only comparing TP abundances all tools performed much better. For L2-TP, MetaPhlAn 4 GTDB had the lowest mean L2-TP of 0.89±0.013, and Kraken2+Bracken had the highest L2-TP with a mean value of 0.91±0.008. The reason for this is the abundance of plasmids in the dataset that cannot be detected with any of the tools, that are not included when only assessing TP abundance predictions. The dataset with the most variance across tools is Mouse for which MetaPhlAn 4 GTDB performed the best with a mean BC of 0.936±0.115, mean L2 of 0.924±0.141, and mean PC of 0.931±0.168. Kraken2+Bracken GTDB performed the worst for BC with a mean value of 0.511±0.179. Kraken2+Bracken was worst for L2 of 0.645±0.173 and PC of 0.509±0.303. Due to the lack of availability of genomes for the Mouse dataset, Fritz et al. reduced the quality requirement of genomes to 'scaffold', suggesting that the dataset is simulated from lower quality genomes [77]. Incomplete genomes might hamper with the internal abundance prediction algorithm of tools, for example in regions with low or no read coverage.

Adjusted TP, FP, and FN values on species-level lead to more accurate assessment of tool performance

In the previous benchmark, TPs, FPs, and FNs were determined by exact matching between taxonomic labels of prediction and gold profiles, disregarding the taxonomic or phylogenetic context. However, a closer inspection of FP and FN predictions within a shared phylogenetic context warranted questioning this strategy. For re-evaluating wrong predictions, FNs are divided into the two categories, detectable taxa and undetectable taxa, based on whether they can or cannot be predicted by the tool (see Section 3.2 for details). In several cases, undetectable FNs formed pairs with FP taxa in close tree distance, suggesting that the present organism was identified, but mislabeled due to a mismatching taxonomies and phylogenies. To quantify this signal, all GTDB prediction profiles have been re-evaluated by benchpro with respect to GTDB's species-level phylogenetic tree. By using the proposed algorithm to re-adjust species level benchmarking, the performance improved for all tools with the GTDB taxonomy. MetaPhlAn 4 benefits the most with removing 166 FP+FN pairs leading to a false prediction reduction of more than 60% and thus rendering 90% of FPs as an error of mapping. mOTUs3 follows with 118 pairs removed and a following reduction of ∼13% and Kraken2+Bracken has 68 FP+FN pairs removed leading to a 1.3% reduction in false predictions. The high relative reduction of false predictions for MetaPhlAn 4 is due to the low initial number of false predictions. All

removed FP+FN pairs are caused by misclassifications due to label mismatching. The low amount of remaining false predictions suggest an otherwise homogeneous mapping between the GTDB and MetaPhLAn4 phylogeny and that FP+FN explain most of the false predictions for MetaPhLAn4. For Kraken2+Bracken, on the other hand, only 68 FP+FN pairs indicate that the cause of FN and FP predictions is more substantial and only partly due to label mismatching. It happens that a single NCBI taxonomic identifier associates with genomes from different genera within GTDB and leading to an error that cannot be removed by the here introduced algorithm. The difference in F1-score between Kraken2+Bracken NCBI and GTDB shows the errors introduced by mapping the NCBI taxonomy to GTDB. However, the main source of false predictions in Kraken2+Bracken and mOTUs3 are spurious low-abundant species caused by high-abundant TP species causing signal in the closer phylogenetic neighbourhood. Although mOTUs3 has a higher reduction in false predictions, the majority of FP predictions remain after re-evaluation.

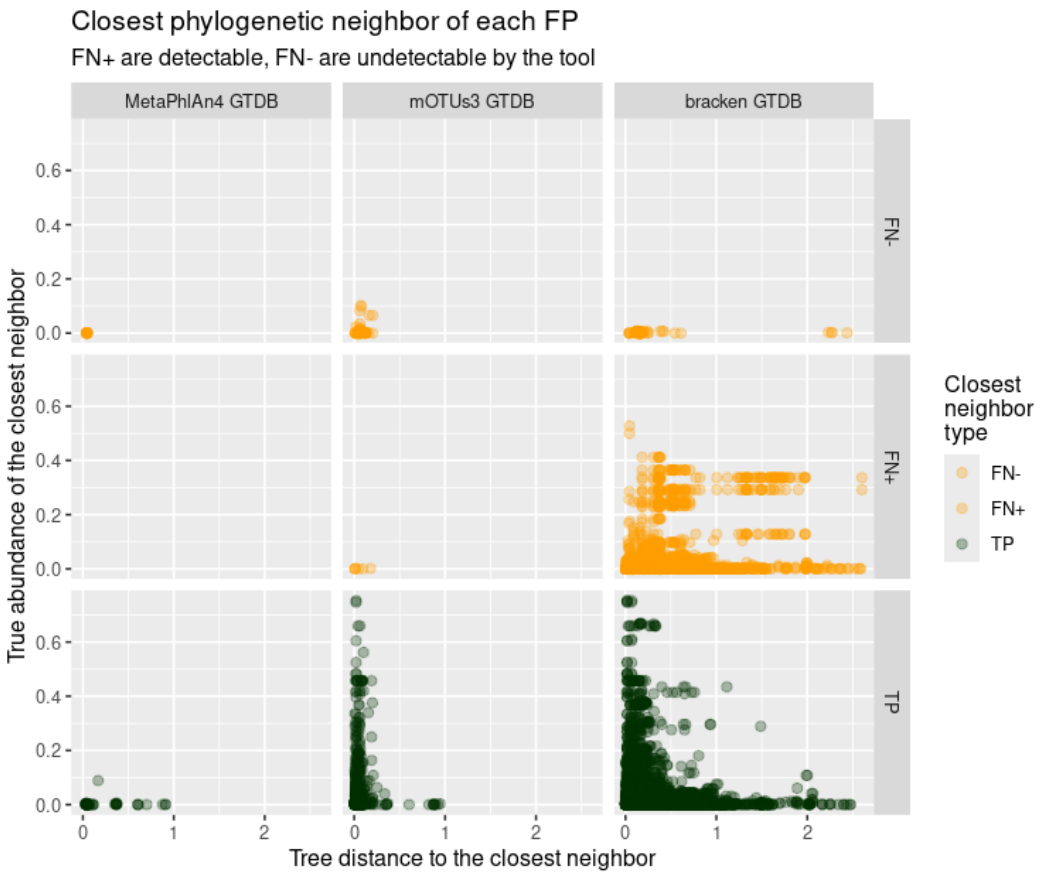

Figure 3.11: Each point is a FP prediction, plotted with respect to the phylogenetically closest TP, FN- or FN+ in the same sample. FN+ are false negatives that are contained in the taxonomic database of the tool. FN- are absent from the tool database. The x-axis shows the tree distance to the closest TP and the y-axis shows the true abundance of the TP, FN+, and FN-. TP, FP, and FN values are after re-evaluation.

After re-evaluating FPs and FNs, I analysed the remaining FPs with respect to their closest phylogenetic neighbour in the gold-standard profile (TP, FN+, FN-). Fig. 3.11 shows that FP predictions by mOTUs3 GTDB are mostly within close phylogenetic distance of TP predictions that are often also high-abundant. This signal suggests that high-abundant taxa in a dataset lead to FP predictions in their close phylogenetic neighbourhood. Kraken2+Bracken GTDB, shares the same signal, however, there are also FPs that are not in close phylogenetic distance to any TP. A potential explanation is that Kraken2+Bracken's database contains mobile elements shared via HGT, that are also shared with species less phylogenetically related. Additionally, the taxonomic species definition in NCBI is only an approximate representation of phylogenetic relationships, which can lead to discordance when mapping to the phylogenetically more consistent GTDB taxonomy.
MetaPhlAn 4 GTDB also shows 12 FP whose closest neighbour in the tree are FN-species. Although the majority of those were resolved during re-evaluating pairs of FP and FN species, some remain, as they did not meet the threshold of 0.04% phylogenetic similarity. Although the threshold was only determined through visual inspection of FP-FN- pair distances, this does not automatically mean the threshold should be increased. Raising the threshold further would permit greater phylogenetic divergence between the predicted and actual species, potentially inflating the perceived performance of the tool. Those 12 FP species have a mean phylogenetic distance of 0.043±0.003 to the next FN- species. MetaPhlAn 4 GTDB does not have any FPs that are closest to a detectable FN (FN+). mOTUs3 GTDB has 6 FPs closest to a FN+, three of which are*s__Anaerotignum lactatifermentan* falsely predicted as *s__Anaerotignum sp001304995*. This shows that the present strain is classified differently between the GTDB taxonomy and mOTUs3's classification. Kraken2+Bracken has a high occurrence (n=5315) of FPs in close distance to high abundant detectable FN+. When stratified by dataset it is apparent that the signal varies between dataset (see Supp Fig. A.1). The signal of FPs in close distance to high abundant FN+ is only present in the Mouse dataset. This signal is potentially an artifact of mapping from NCBI to GTDB (see Section 3.2.6), and also the Kraken2+Bracken's whole-genome approach.

After re-evaluating false predictions, F1-score, sensitivity, and precision were recomputed on the adjusted TP, FP, and FN counts. With the resulting adjusted scores of species level performance, MetaPhLAn 4 achieves an F1-score of over 0.9 for every single dataset and a mean F1-score of 0.972±0.02, marking an increase of almost 0.05 compared to the unadjusted scores. MetaPhlAn 4 GTDB's precision is at a remarkable 0.996±0.011 due to a low mean FP count of 0.673±1.72, which is reduced from 4.935±4.622 in the unadjusted benchmarks. mOTUs3 GTDB's F1-score increased by ∼0.02 to a mean of 0.89±0.073 after adjustment reduced false positives from 10.093±4.319 before adjustment to 8.327±3.592 after adjustment. Kraken2+Bracken only experienced a slight increase in F1-score from 0.427±0.157

Figure 3.12: Profiling performance across samples, environments, and tools measured with F1-score, precision, and sensitivity after re-evaluating false predictions. The boxplots represent different tools and each data point is a sample. From top to bottom, the row panels are F1-Score, precision, and sensitivity and the column panels stratify between different environments with the left column being the summary across all environments.

to 0.432±0.158 and is still behind the other tools. A general large amount of noise of FPs of 137.252±62.172 to begin with is reduced to only 136.551±62.133 after adjustment, explaining the little increase.

Zooming into the individual datasets, I see that there are no universally difficult datasets. Kraken2+Bracken GTDB, for example, struggles with the Mouse dataset (mean F1-score of 0.319±0.064), likely due to an insufficient coverage of species in the database, which is supported by the high mean FN count of 82.571±35.376. MetaPhlAn 4 GTDB performs consistently across all datasets, but excels especially in gut related datasets (Human Gastrointestinal, Mouse) where it has a perfect precision of 1±0 for Human Gut and an almost perfect mean precision of 1±0.001 for Mouse. mOTUs3 GTDB struggles with Skin and Urogenital samples but still has decent F1-scores with 0.818±0.041 and 0.837±0.06, respectively.

In another benchmark, I tested the detection of low abundant taxa in the sample as well as if high abundant taxa are missed (Fig. 3.13). In the adjusted benchmark, MetaPhlAn 4 GTDB captures all taxa that are above 1% relative abundance and detects taxa down to a relative abundance of 0.0007%. The most abundant taxon mOTUs3 GTDB missed had a relative abundance of 4.24% but can detect taxa down to 0.0003% relative abundance. Kraken2+Bracken GTDB missed taxa with a relative

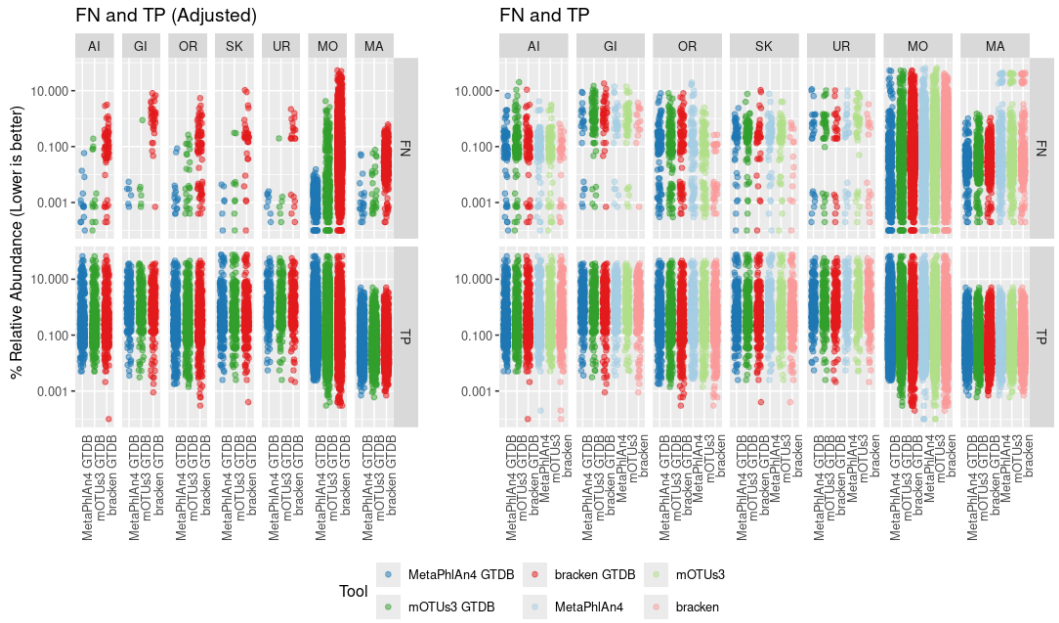

Figure 3.13: Detection threshold of species based on abundances. The y-axis displays the gold standard abundance of each species (point) across all datasets (horizontal panels, UR=Urogenital, GI=Gastrointestinal, MA=Marine, MO=Mouse, OR=Oral, SK=Skin, AI=Airways) and the x-axis stratifies tools (color). The left panel shows FNs and TPs after adjustment and the right panel shows the unadjusted values. The top panel shows the FNs and the bottom panel shows the TP. High abundant FNs indicate failure to detect important species while low abundant TPs show that a tool has a high sensitivity for low-abundant species. FNs are filtered for species that are detectable by the respective tool.

abundance up to 52.75% but has the lowest detection threshold with 0.0001% relative abundance. Considering that an abundance threshold of 0.01% for taxa would increase the precision and F1-score (Fig. 3.8) puts Kraken2+Bracken GTDB's low abundance detection threshold into perspective. Microbiomes from different environments typically have different complexities and species counts. To investigate how the profiling performance is linked to the species richness, I compared the profiler outputs for all samples within the Mouse dataset (Fig. 3.14). CAMI Mouse is the only dataset that exhibits a great variance in richness, so I can avoid the bias of comparing between datasets. It is important to note that all datasets have similar number of reads at varying species richness (see Fig. A.5). Firstly, while precision is increasing with species richness, sensitivity is declining. This evens out for mOTUs3 GTDB and MetaPhLAn 4 GTDB such that the F1-score is constant for different species counts. For Kraken2+Bracken GTDB, however, there is a notable increase in F1-score from ∼0.25 to ∼0.38 (linear regression line) caused by the increase in sensitivity. The general decline in sensitivity is caused by the constant read count across all samples. The more species there are in the dataset without an increase in depth, the more taxa will be pushed below the detection threshold. Kraken2+Bracken's

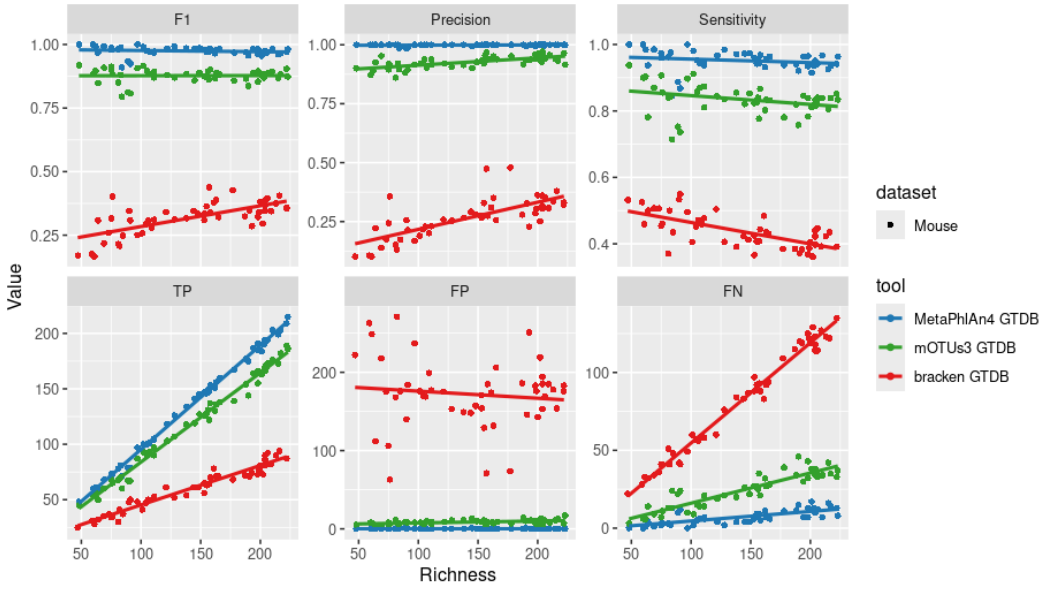

Figure 3.14: Richness with respect to F1-score, precision, and sensitivity. Each dot represents one sample in the Mouse dataset and all statistics are after adjustment on species level. Richness on the x-axis is calculated as TP+FN and the y-axis shows the value for the respective statistic in each panel.

increase in precision is most likely a result as with increasing taxa the likelihood of a random taxon to be a TP increases.

3.3.3 Strain-level Benchmarks

Benchpro offers dedicated benchmarks for strain-resolved tools, with a focus on tools that produce per-species MSAs and phylogenetic trees on multiple samples. These analyses facilitate insights like strain-tracking and strain-sharing. Pairwise distances—either computed directly from multiple sequence alignments (MSAs) or derived as cophenetic distances from phylogenetic trees—can serve as thresholds to infer whether two genomes represent the same or different strains. In longitudinal datasets, such as those from the human gut microbiome, additional assumptions can be made: because the gut microbiota tends to remain relatively stable over time, a high degree of strain sharing across timepoints within individuals is expected [257, 66]. These expectations can be used to validate the placement of samples within phylogenetic trees and to detect events such as strain replacements..

With this in mind, I designed 200 simulated datasets carrying the same 46 Species with the same and different strains. With this type of dataset where each strain is present in multiple samples at different coverages, I can determine tree monophyly per species to assess if samples with the same strain form pure clusters. For quantifying the resolution at which strain-clusters are separated, I incorporate tree distances measuring the intra-strain distances vs. inter-strain distances. To combine both metrics, I measure the monophyly of strains with respect to their closest

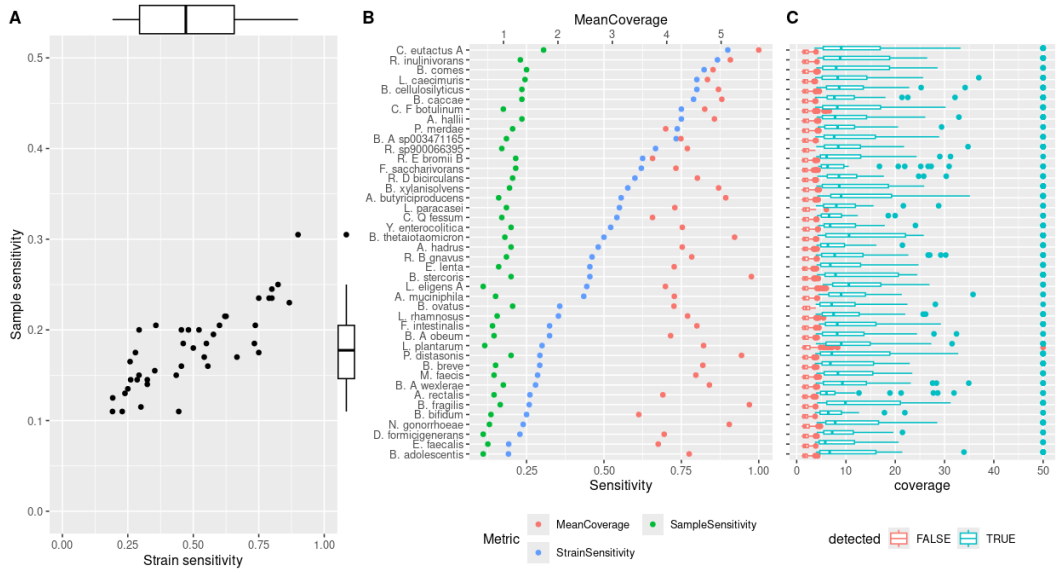

Figure 3.15: Strain and sample sensitivity of StrainPhlAn 4 on different species. **A** shows sample sensitivity plotted against strain sensitivity for each species. **B** also shows sample and strain sensitivity per species and includes the mean coverage across samples for this species. **C** resolves the gold-standard vertical coverage per sample in boxplots for each species, stratified by whether it is included in StrainPhlAn 4 phylogenies. Refer to Table A.2 for species abbreviations.

neighbour-strain outside of the strain-cluster. As tree distances serve as proxies for ANI values, this allows us to measure at what resolution a tool fails to correctly distinguish and place strains in a tree. As trees are computed from MSAs, I also compute metrics directly on the MSAs to quantify the error in sequences within the same strain and putting those errors into context of total nucleotide variability between all sequences. Lastly, Roary is tool to generate core-genome phylogenies from a set of (conspecific) genomes. Benchpro quantifies the distances between predicted trees and Roary reference trees both considering phylogenetic topology and distances. Benchpro automatically computes these metric from provided MSAs and phylogenetic trees given a test dataset and gold standard trees. Using StrainPhLAn 4, I generated MSAs and trees on a dataset spanning 200 samples, each containing a single strain for 46 selected species. The dataset is simulated from 1335 different strains with a mean group size of 6.89±3.97. With benchpro, I computed the aforementioned benchmarks on the StrainPhlAn 4 output.

Sensitivity of StrainPhlAn 4

StrainPhlAn 4 has a low sensitivity of 40.5% of strains (n=525) and 16.5% of samples (n=1510) (Fig. 3.15). This means the remaining samples are discarded due to insufficient coverage for SNP calling and consensus reconstruction, and not included in the output alignments or trees. The species with the lowest number of retained samples is *s__Bifidobacterium adolescentis* with a strain sensitivity of 20.9% and

and sample sensitivity of 11.2% (9 out of 43 strains and 22 out of 200 samples retained). On the other end is *s__Coprococcus eutactus_A* with a strain and sample sensitivity of 90% (18 out of 20) and 30.5% (61 out of 200), respectively. The mean sensitivity across species is 0.5±0.21 for strains and 0.18±0.04 for samples. With sensitivity put into context of per-species mean sample coverage, I find that most variation is explained by insufficient coverage. Interestingly, *s__Lactoplantibacillus plantarum* is detected by MetaPhlAn 4 in all of the 200 samples, but only 23 samples are contained in the phylogenetic tree for StrainPhlAn 4. Of all missing samples, eight have a vertical coverage of more than 5x, with one sample having a coverage of 50x. This specific strain *s__Lactoplantibacillus plantarum GUT_GENOME142473* is present in two samples at with 50x and 8.28x, is missed by StrainPhlAn 4, but present in the taxonomic profile of MetaPhlAn 4. MetaPhlAn 4's predicted relative abundances are less than the gold-standard abundances with 2.4% instead of 4.3% (Sample 157) and 15% vs. 24% (Sample 42). Unfortunately, the log files did not contain any information on why these samples were not included in the strain-level phylogenies. Further species missing in StrainPhlAn 4 output despite vertical coverages greater than 5x are *s__Lachnospira eligens_A* with 5 missing samples, *s__Clostridium_F botulinum* with 4 missing samples, and *s__Lacticaseibacillus paracasei* and *s__Lacticaseibacillus rhamnosus* with one each. Although StrainPhLAn 4 quite consistently detects strains if their vcov is larger than 5x, in some cases strains are missed despite sufficient abundance. This indicates that the marker genes do not receive enough hits to be included in the MSA. As all selected source strains in the dataset are isolate genomes, MAG-related errors in the dataset can be ruled out.

Monophyly score

Next, benchpro will assess how well samples that carry the same strain are resolved with respect to monophyly in the generated phylogenetic trees (Fig. 3.16). The monophyly score is only calculated for strains that occur in at least 2 samples. Across all species, StrainPhlAn4 resolves 76.4% of strains (N_{SP4}=525) and 72.5% of samples (N_{SP4}=1510) with a perfect monophyly score of 1 with respect to species and strains that were included in their trees. StrainPhlAn 4 resolves 7 out of 42 species with perfect monophyly scores, namely *s__Agathobacter rectalis*, *s__Agathobaculum butyriciproducens*, *s__Akkermansia muciniphila*, *s__Bifidobacterium adolescentis*, *s__Clostridium_Q fessum*, *s__Ruminococcus_B gnavus*, and *s__Ruminococcus_E bromii_B*. The lowest mean monophyly score is recorded for *s__Longicatena caecimuris* with 0.692±0.257 (49 samples and 12 strains), followed by *s__Roseburia inulinivorans* with 0.718±0.276 (46 samples and 13 strains), and *s__RUG115 sp900066395* with 0.744±0.295. Species with a higher mean monophyly scores tend to have lower sample sensitivity (Pearson correlation

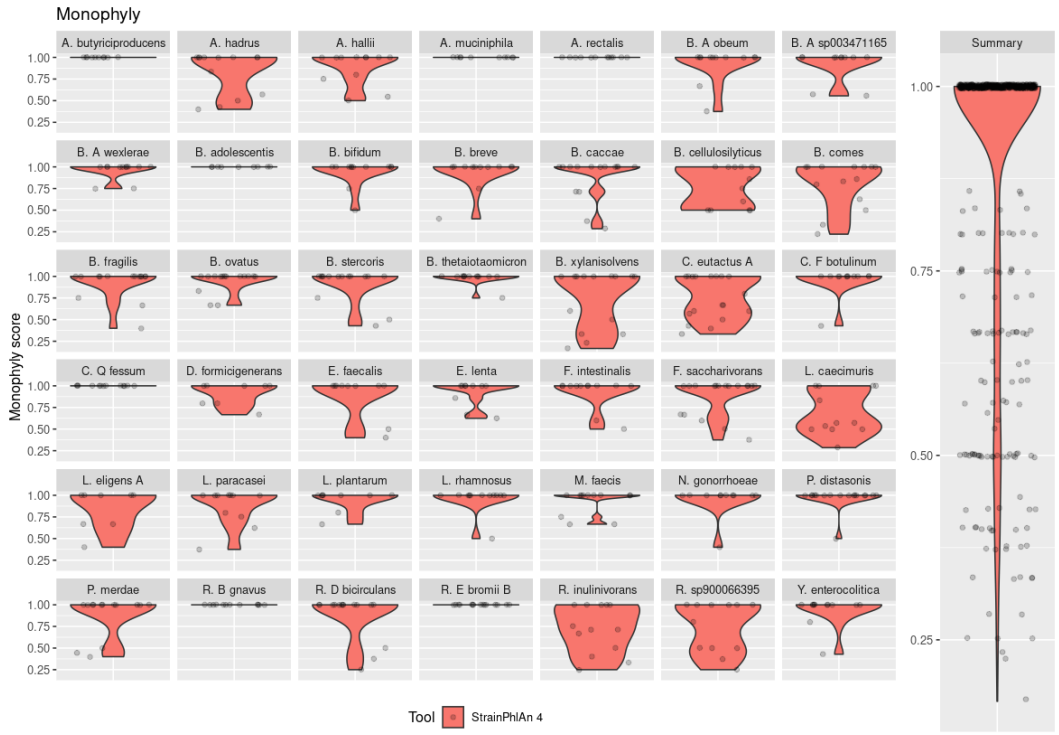

Figure 3.16: The monophyly score of StrainPhlAn 4 stratified across bacteria in panels. The monophyly score measures how pure clusters of samples carrying the same strain in a tree are (see Section 3.2.2). The right-most column is a boxplot summarizing monophyly score across all species. Refer to Table A.2 for species abbreviations.

R=-0.6, p=0.000026, 3.17), which is another demonstration of the trade-off sensitivity vs. precision.

Strain Cluster Error within Phylogenetic Trees

Benchpro compares the phylogenetic distances of pairwise samples carrying the same strain, to those carrying different strains (Fig. 3.18). Ideally, samples carrying the same strain clearly separate from other samples by having high within-strain cluster similarity and low between-strain cluster similarity within the phylogenetic trees. To measure this, I calculate the Max Cluster Error (MCE) as described in section 3.4. The three species *s__Coprococcus eutactus_A*, *s__Longicatena caecimuris*, and *s__Roseburia inulinivorans* have 11, 7, and 7 strains with a positive MaxClusterError have low mean monophyly values of 0.754, 0.692, and 0.718. Even if two samples carry different strains with (for a tool) indistinguishably high similarity, within-strain and between-strain distances should both all be close to zero. The presence of a positive MCE is indicative of errors in the MSA that is either due to read errors or wrong alignments. Especially in cases where strains in the dataset are highly similar (>99.9% ANI), this likely results in positive MCE values as the strains are too similar for the tool to reliably resolve (see Fig. 5.14 for plot displaying ability to

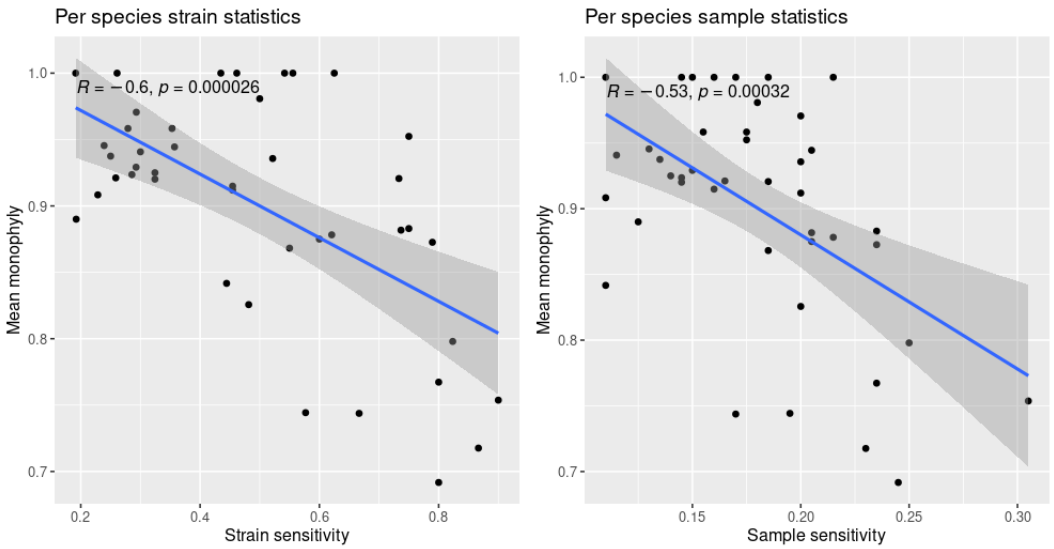

Figure 3.17: Correlation between sensitivity and monophyly scores per species. Each point is a species and the monophyly score measures how pure samples of the same strain cluster together. Sensitivity on the x-axis is how many strains and samples from the original dataset are still contained in the resulting phylogenetic trees. The correlation is computed with pearson and the regression line was computed according to a linear model.

resolve strains monophyletically based on their cophenetic distance). As the dataset consists solely of reads simulated from isolate genomes, quality issues with MAGs can be ruled out. However, also isolate genomes have been shown to exhibit signals of contamination, potentially confounding the strain-level results [184]. This will be further expanded on in Chapter 5.3.5.

Increased error rate leads to decreased Monophyly

Complementary, I investigated errors within the MSA, calculated per group of consensus sequences in the MSA belonging to the same strain (Fig. 3.19). Errors can occur for two reasons. First, mis-alignments of reads, either from within the species or from other species, can lead to the detection of false SNPs and second, sequencing errors are incorporated as true biological variation. As error rate is calculated as errors over alignment length, the error rate goes up with decreasing alignment length. Across all strains, StrainPhlAn has a mean error rate of 0.0016±0.0037 and a mean error count of 5.63±10.36 (Fig. 3.19 B). *s__Bacteroides ovatus* has the highest mean error rate with 0.0144±0.0126 and mean error of 43.53±35.85. The mean monophyly for this species is 0.944±0.121 and not as low as the error rate might suggest. *s__Bacteroides caccae* comes second with a mean error rate of 0.0063±0.006, followed by *s__Bacteroides xylanisolvens* with 0.0038±0.0054. Both have non perfect monophyly with a mean of 0.873±0.242, and 0.744±0.338, respectively. To examine this relationship, I analysed error rate with respect to monophyly, but found no

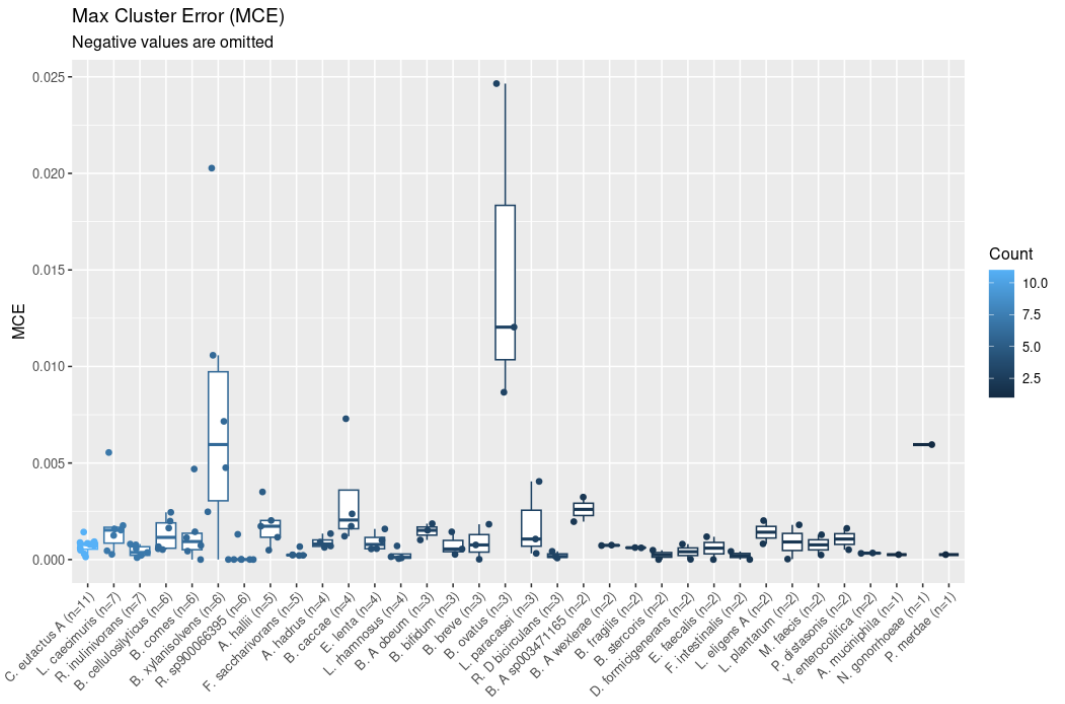

Figure 3.18: Max Cluster Error (MCE) on the y-axis per strain and stratified by species. MCE is computed as explained in section 3.2.2 and quantifies how well all samples with a certain strain cluster together in the phylogenetic tree with respect to all other samples. The color encodes the number of strains with a positive MCE.

significant correlation (Fig. 3.19 C). This can be for two reasons: a) if there is no closely related strain next to a strain with a high error-rate, then monophyly (as a distance agnostic measure) will not be affected, and b) if the error rate is smaller than the genetic signal to distinguish strains due to the selection of less conserved marker genes, monophyly will also not be affected. To further investigate this, I correlated monophyly per strain with the minimum gold-standard distance (distance in Roary phylogenetic tree) to any strain present in the StrainPhlAn 4 phylogenetic tree. It shows that monophyly is mostly driven by how closely related the nearest neighbor in the tree is, however, monophyly also varies significantly between species ($p=2 \cdot 10^{-16}$ for distance to nearest neighbar and $p=2 \cdot 10^{-6}$ for species, ANOVA). One example is the strain *GUT_GENOME142064* of the species *s__Clostridium_F botulinum*, which has a monophyly of only 0.43, however there are no errors within the MSA and the closest strain in the phylogenetic tree has a similarity of 0.9975. Otherwise, StrainPhlAn 4 resolves strains with perfect monophyly if their closest neighbor has similarity of 0.999 or lower. Noteworthy, however, is that there are 452 (12%) within-strain sample pairs that have a similarity lower than 0.999 (cophenetic distance of StrainPhlAn 4 phylogenetic tree).

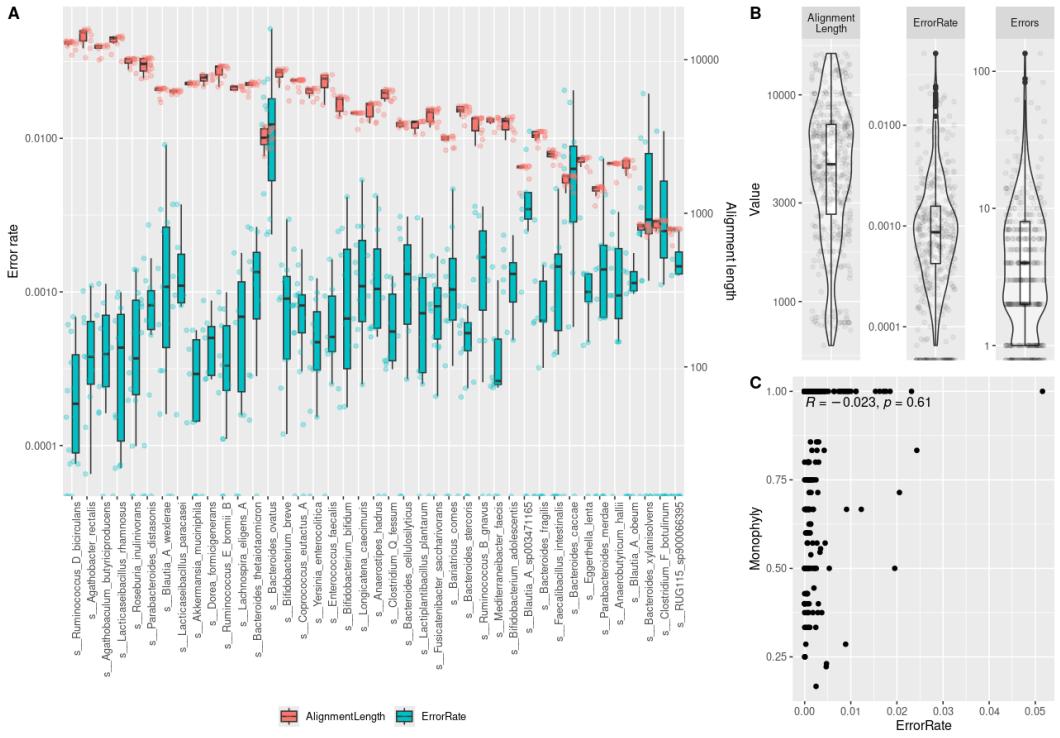

Figure 3.19: **A**, Error rate and alignment length of MSAs per species. Error rate is determined as number of multi-allelic positions per group of sequences representing the same strain considering positions with at minimum 2 informative bases (A,C,G,T) and alignment length is the number of bases in the alignment with at least 2 informative bases (see Section 3.2.2 for details). Each data point is a single strain. **B**, summary over alignment length, error rate and error count. **C**, error rate in context of monophyly per strain. There is no significant correlation.

StrainPhlAn 4 cophenetic distances are no proxy for ANI

In the last benchmark, I compare the predicted phylogenetic trees with the gold standard trees produced with Roary, with respect to tree topology and cophenetic distances. For comparison, I compute the same distances for randomly generated trees. Comparing StrainPhlAn 4 on RFnorm (normalized Robinson-Fould) and wRFnorm (normalized weighted Robinson-Fould), I find that accounting for cophenetic distances (weighted) increases the distance to the gold-standard (Fig. 3.20). This is a cause of StrainPhlAn 4's skewed cophenetic distances, which are not equivalent to ANI values. StrainPhlAn 4's cophenetic distances can exceed 1 within species, which is about 20x times higher as expected for ANI based distance values (0.05 with 95% ANI).

3.4 Discussion

This chapter has shown that benchpro is able to accurately profile the performance of taxonomic profilers and strain-level tools. For benchmarking taxonomic profiling,

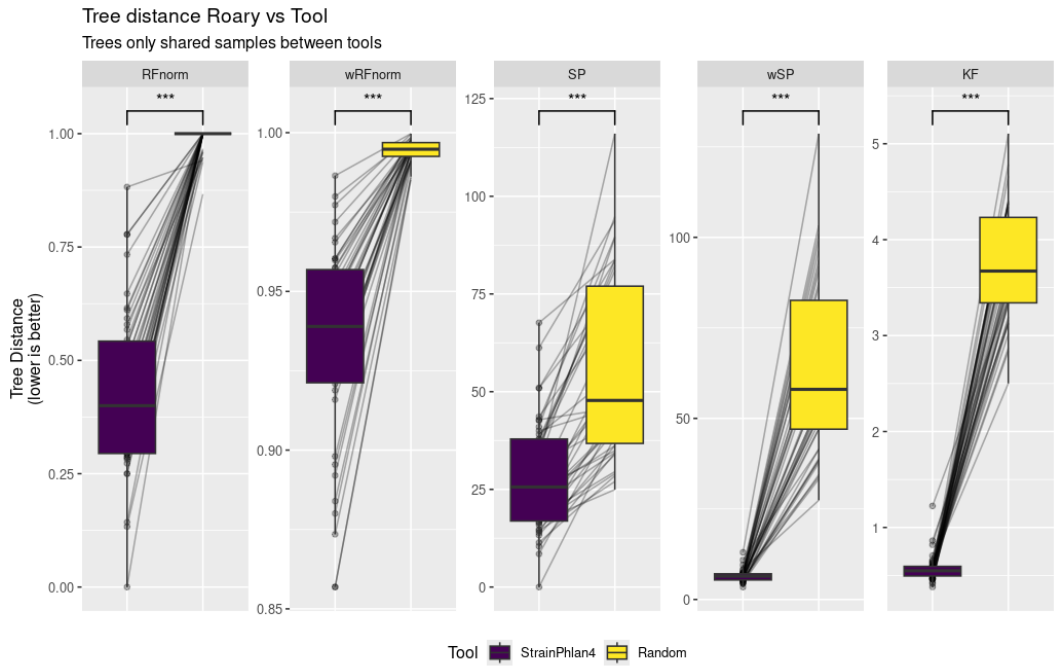

Figure 3.20: Distance between the phylogenetic trees of StrainPhlAn 4 and randomly generated trees to the gold standard tree constructed with Roary for each species (n=42, four species not detected by MetaPhlAn 4+StrainPhlAn 4 due to lack of coverage in their database, see Section 3.2.5). Significance (***: $p \leq 0.001$) was calculated with a paired t-test, with species between protal and StrainPhlAn 4 as pairs. The utilized metrics are normalized Robinson-Fould distance (RFnorm), normalized weighted Robinson-Fould (wRFnorm), Steele and Penny distance (SP), weighted Steele and Penny distance (wSP), and Kuhnert-Felstenstein distance (KF) (see Chapter 3.2.2 for details).

benchpro adjusts for errors caused by differences in phylogeny and taxonomy that do not reflect the actual tool performance, but rather the uncertainty when mapping between different phylogenies and taxonomy versions. This helps users to make an informed decision about which tool to use for a project and get the best results out of a dataset. Benchmarking the taxonomic profiling performance of mOTUs3, MetaPhlAn 4, and Kraken2+Bracken with benchpro provided a more in-depth comparison than previous independent benchmarks [242, 316, 177]. For an even more comprehensive assessment, it would be necessary to create different custom databases for Kraken2 and KrakenUniq using the GTDB marker genes as well as the whole genome of all species-representative genomes. This is necessary to further disentangle the impact of a) marker genes vs. whole-genome databases and b) NCBI Taxonomy vs. GTDB Taxonomy on Kraken's profiling performance. A benchmark in the publication for MetaPhlAn 4 compared Kraken2 using databases from RefSeq [193] and GTDB [206], but it is unclear how the taxonomic assignment for GTDB was performed [26]. I decided against including KrakenUniq in this benchmark due to its high memory consumption, as the standard database for download is 377 GB

in size [3]. KrakenUniq should still be included in the benchmarks as it offers a higher precision over Kraken2 and Kraken, although it should be noted, that a recent benchmark has found only little difference in performance [219]. To measure the profiling performance in the presence of unknown taxa, a leave-one-out approach simulates the absence of whole taxonomic clades in databases [290]. However, this is not generally applicable due to the fixed pre-built databases for mOTUs3 and MetaPhlAn 4. I further set a focus on evaluating the correct classification of taxa over assessing the correctness of predicted abundances. As the correct classification of taxa is a requirement for correct abundances, I prioritized classification over abundance benchmarks, and hence did not calculate the abundance metrics on the adjusted species predictions. Further, detailed abundance benchmarks are already covered in a different benchmark paper [316]. Adjusting the species-level classification was inspired by MetaPhlAn 4's benchmark in their own publication [26]. MetaPhlAn 4's species clustering into species genome bins (SGBs) is based on whole genome distances with clusters spanning 5% genetic diversity [211]. In a separate benchmark, 'SGB evaluation', Blanco-Míguez et al. evaluated the correctness of taxonomic classification by counting a TP if any genome in a predicted SGB matches a genome in the sample. By doing so, their benchmark results improved significantly. As I was not able not recreate this benchmark for the comparison with protal, I implemented the adjustment of FP and FN taxa based on their phylogenetic distance in the GTDB phylogeny to provide a fair benchmark (see 5.3.3). I argue that the most important quality for a taxonomic profiler is to have a 1:1 ratio of species present to species detected. The adjustment within benchpro fulfils this requirement, as each FP can only be matched with one FN, and the taxonomic label of the FN must not be previously present in the tool database. FP predictions that are caused by high-abundant TP taxa still remain, which is problematic as they can elicit a spurious co-occurrence signal [68]. On the other hand, for applications requiring the exact taxonomic label, such as detecting a pathogen, predicting a close phylogenetic neighbour is still incorrect. Due to time constraints, I concentrated on StrainPhlAn 4 for the strain-level benchmarks, thereby excluding other tools from the analysis. In addition, I initially planned on including metaSNV on the mOTUs3 output, however, the benchmark failed due to the large size of temporary files produced (>5TB).

Additionally, the focus was on lightweight strain-level tools for comparison with Protal, as tools like inStrain and MIDAS involve more complex workflows. Another limitation is that the simulated dataset does not resemble any naturally occurring community, as every sample contains a strain of 46 species. This, for example, prevents one from assessing whether FP predictions are included in the MSA and the phylogenetic tree. For StrainPhlAn 4, the default parameters were used, however a recent publication suggests that StrainPhlAn 4 still performs well with lowered coverage requirements [182].

[3] https://benlangmead.github.io/aws-indexes/k2, accessed 23rd June 2024

3.5 Author contributions

I conceptualized, developed, and implemented benchpro by myself, under the guidance of Falk Hildebrand. Using monophyly as ANI-independent measure of performance on longitudinal data was suggested by Falk Hildebrand.

Chapter 4

Alignment-free taxonomic profiling and SNP detection with varkit

4.1 Introduction

Alignment-free methods for taxonomic binning are popular due to their fast performance and flexibility of use when compared to their alignment-based counterparts. Especially Kraken [308], Kraken2 [307], and KrakenUniq [32] are widely used due to their simple, yet effective approach of assigning each read a taxonomic identity through k-mer matches instead of costly alignment against the reference. In addition, the option of creating a database from custom sequences opens up a variety of use-cases such as filtering host contamination [98], targeted screening for selected strains, and incorporating taxa that are not yet represented by public databases. In standard metagenomic profiling, assuming a similar database coverage, alignment-based profilers such as mOTUs [232] and MetaPhlAn [26] tend to perform better at the species level. They achieve a similar sensitivity to tools from the Kraken family, but with a lower false positive rate [26, 316, 219]. This is even true when tools of the Kraken family are used in conjunction with Bracken [159] to filter and calculate abundance profiles from the taxonomic bins [177]. Yet, the ∼5-fold speed advantage of Kraken2+Bracken over MetaPhlAn 4 and mOTUs3 still matters when analysing datasets of hundreds of samples and being able to turn five days of analysis into one.

When the taxonomic resolution is increased from species-level and above to subspecies- and strain-level, Kraken is only able to detect strains that are contained in its database, but is unable to reconstruct strain-level diversity across samples. This limitation is not shared by MetaPhlAn and mOTUs, as their output can be further processed with StrainPhlAn [26] and metaSNV [55], respectively, to reconstruct the phylogeny of known species present in multiple samples. Strain-level resolved analyses are required, for example, for identifying strain-transmission events between hosts [98], tracking pathogen outbreaks and their evolution, and analysing phenotypes that show correlations with taxa below species-level. Tools most commonly

quantify strain-level genomic variation by comparing SNPs between samples with respect to a shared reference [8]. This requires alignment against a shared reference to call SNPs, reconstruct per sample consensus sequences and build an MSA to quantify the diversity; as done by mOTUs+metaSNV and MetaPhlAn+StrainPhlAn. The exception among alignment-free tools is GT-Pro, a tool for metagenomic genotyping which uses an alignment-step during the initial database building, to extract and filter for species-specific SNPs from a species' pangenome (requiring >10 genomes per species) to then store k-mers covering those SNPs [250]. This allows GT-Pro to rapidly genotype samples without alignment and reports counts for reads covering each allele of each SNP in the database. While allowing for strain-resolved analysis, this approach, too, relies on SNPs in references and will only detect known SNPs, as opposed to alignment-based approaches that are able to quantify genomic diversity with *de novo* SNPs. Further, the database building process is computationally expensive in regard to both time and memory and the default database only covers bacteria commonly found in the human gut. Thus, for most strain-level analyses from short-reads, alignment is still a necessary and costly step to accurately quantify within-species genomic diversity across samples.

Here I present the variant k-mer identification toolkit (**varkit**), to combine the speed and simplicity of k-mer based methods with the accuracy and strain-resolution of alignment-based methods. Varkit stores k-mers of GTDB marker genes of all 62,291 species-representative genomes—both isolate genomes and MAGs—in its database (r207), to strike the balance between memory efficiency and representation of taxonomic diversity across all environments. Further, varkit employs patterns of exact matching spaced k-mers between read and reference as a novel method to capture and analyse *de novo* intra-specific genomic variation.

4.1.1 Contribution of this thesis

With varkit, I explore a novel method for SNP calling using k-mers and thus opening up k-mer based approaches to strain-level analyses. Thus, varkit bridges the gap between fast k-mer based analyses restricted to species-level and slower alignment-based tools like MetaPhlAn or mOTUs. Varkit avoids alignment and direct comparison with the reference and instead relies on indirect comparison by using exact spaced k-mer match patterns to identify SNPs. This is facilitated by two means: 1) by implementing a novel data structure that is both fast and memory efficient and stores k-mers along with their exact position in the reference and 2) by a novel approach for SNP calling using patterns of matching and non matching spaced k-mers between read and reference. Different shapes for spaced k-mers have previously been evaluated in the context of match sensitivity for read classification, but have not been explored for SNP calling. I explored how different k-mer sizes and shapes influence the sensitivity of SNP calling in conjunction with this novel species classifier.

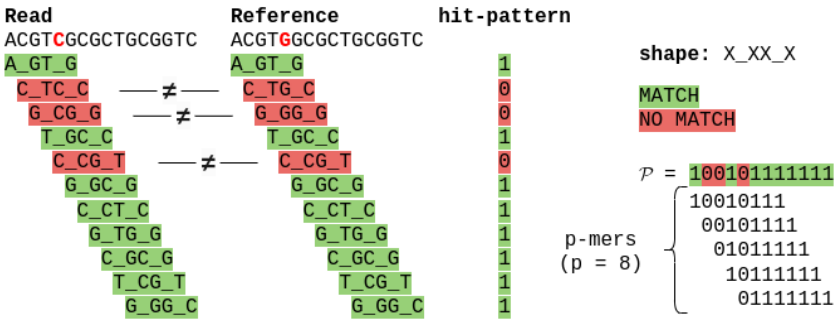

Figure 4.1: Given a query and a k-mer shape (here: 'X_XX_X') a hit-pattern is defined as a sequence of successful (1) or unsuccessful (0) k-mer lookups with respect to a reference. Position 5 (starting position being 0) in the query is considered mutated with respect to the reference (C→G) so all k-mers overlapping and with an 'X' at this position are unsuccessful lookups. A k-mer overlaps all positions where the k-mer shape has an 'X'. Gap positions denoted as '_' are ignored.

4.2 Methods

4.2.1 SNP calling with k-mer look up patterns

The core idea for varkit's approach to SNP calling is leveraging the sequence of matching and non-matching k-mers between the query and a reference. This sequence, from now on referred to as hit-pattern, is used to infer the exact positions where read and reference disagree. A hit-pattern P is a sequence defined over the alphabet $0, 1$ where 1 denotes a match between two k-mers, and 0 a mismatch, respectively. In practice, when extracting and storing k-mers from multiple reference sequences, one will find non-unique k-mers that are present across multiple references.

To explain the workings of the method, I assume that all k-mers are unique within a sequence so there are no ambiguous (uncertain which reference they are from) or random hits between a query and a reference. P is thus a sequence of 1s and 0s, depending on whether the k-mer at the corresponding position has a match (1) in the reference, or not (0). Figure 4.1 shows how k-mer lookups for a query lead to the hit-pattern. The idea is to take this hit-pattern and infer the SNP positions without explicit alignment of the read against a reference. The length of the hit-pattern depends on the length of the query and the overlap between query and reference. To break down the problem from a variable size into a fixed size, I extract all possible fixed size substrings of the hit-pattern. I refer to these substrings of length p as p-mers and infer the potential SNP positions for each p-mer individually.

4.2.2 Building the varkit reference database of hit-patterns

P-mer is defined as a substring of length p of a pattern $\mathcal{P}$. Since each p-mer is the result of p k-mer look ups of a sequence of length $p + w - 1$, with w being the size of the k-mer shape, one can reconstruct all reads (with a given k-mer shape) that

potentially give rise to a certain p-mer. To call variants given a hit-pattern $\mathcal{P}$ I build a dictionary mapping p-mers onto SNP locations relative to the p-mer. The pattern database is built by iterating each possible sub-pattern. The number of p-mers is 2^p. For instance, with p = 16, there are $2^{16} = 65,536$ possible p-mers. For each p-mer, the shape is aligned with the match positions in the pattern to determine positions that are not covered by any k-mer (Figure 4.2 1.). These positions are potential mutations. Next, all permutations of the potentially mutated positions are computed to get a set of candidate reads that could have generated the pattern (Figure 4.2 2.). For all candidate reads the hit/miss pattern is then computed and candidate reads that do not contain the original pattern are discarded (Figure 4.2 3.). The intersection of SNP positions of the remaining candidate reads are the SNP positions associated with the original pattern (Figure 4.2 D). This approach does not produce false positives in a model setting, as intersecting the SNP position as shown in Fig 4.2 D ensures that only unambiguous SNPs are included in the database. In training the database, the computationally expensive part is permuting all possibly mutated positions to get a set of candidate reads from a given pattern as shown in Fig. 4.2 C. The number of candidate reads scales exponentially with the number of potential mutations and for this reason a threshold limit l is applied when necessary, to reduce the time to compute the pattern database, skipping all patterns with more or equal to l potential mutations.

When analyzing real metagenomic samples, false positives SNPs may still occur. This is because the database stores each k-mer with the lowest common ancestor (LCA) of all taxa it occurs in. As a consequence, a genus level k-mer surrounded by species-level k-mers (which are considered unique k-mers as each species is only represented by one strain) is not necessarily present in the detected species if at least two other species under the same genus contain the k-mer. This case cannot be detected as information on gene id and position is lost for non-unique k-mers. Non-unique k-mers are stored with the lowest common ancestor (LCA) of all taxa it occurs in and gene id and gene position is dropped.

However, this approach cannot consider reads with insertions and deletions. The impact of this restriction is likely small, since the reference for varkit are universal marker genes, that typically have mostly SNP variation. Further, multiple strain-resolved tools omit indels when quantifying intra-specific variation, being similar to the varkit approach in this regard [250, 215].

4.2.3 Finding k-mer shapes for SNP calling

K-mer based algorithms often employ spaced k-mers to increase the detection sensitivity with long k-mers [34], and is most prominently implemented in Kraken2 [307]. Long k-mers are necessary to retain a certain amount of specificity, as k-mers which are too short cannot identify taxa uniquely on species level or below. For this reason,

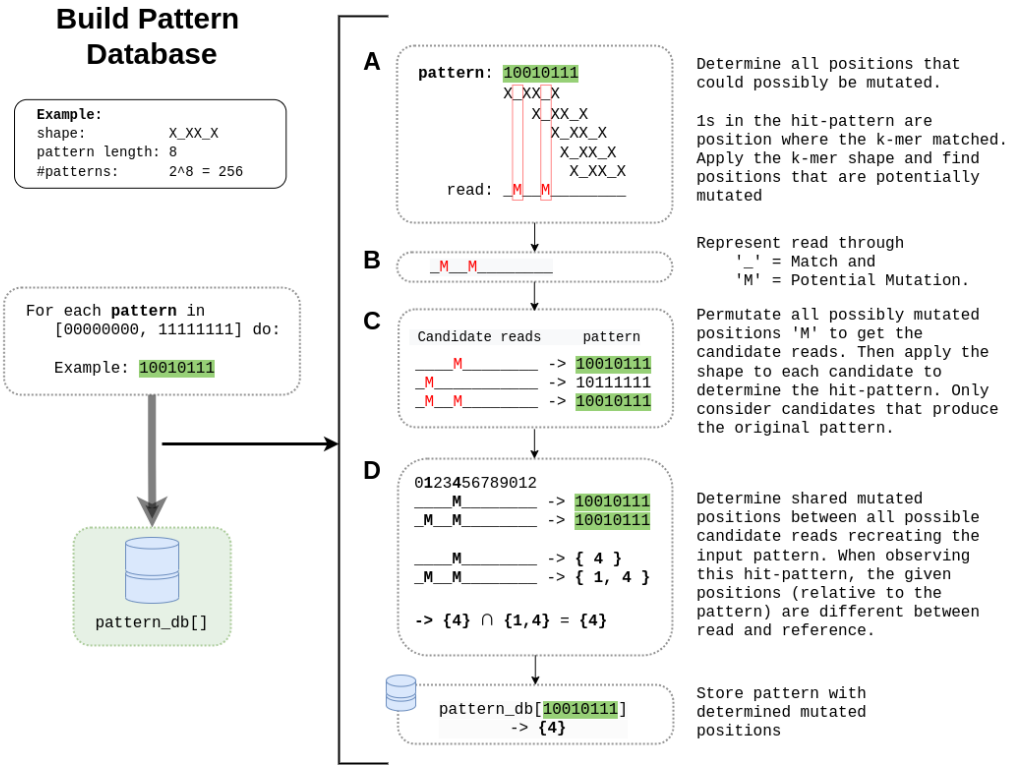

Figure 4.2: The workflow for building a pattern database with a given k-mer shape and pattern length (here: `X_XX_X` and 8).

Kraken uses 31-mers are they have been shown to uniquely identify Species. In contrast to alignment, Minimap2 [145], for example, is more sensitive by using 15-mers to find candidate references for alignment. This does not come at the cost of taxonomic resolution, as their data structure stores all reference locations for a given k-mer (not just the LCA like varkit). The trade-off lies in computational overhead, as more k-mer hits require a more intensive post-processing. This is important, as the goal of an aligner is to align as many query sequences as possible, while taxonomic binning has the more general goal to accurately determine the presence of taxa.

Spaced k-mers improve sensitivity for long k-mers without loss of taxonomic resolution for larger k [34]. As varkit's approach is more similar to Kraken and does not resolve all reference locations for a k-mer I choose to employ a spaced k-mers approach to retain uniqueness with improved sensitivity. As I will discuss in Chapter 4.3, the k-mer shape determines not only the sensitivity, but also how many SNPs can be detected between query and reference and is dependent on the ANI between query and reference. Hence, the choice of a fitting k-mer shape is crucial for the performance of varkit. In order to find the right k-mer shape for SNP calling, I employed an approach similar to evolutionary algorithms exploring the k-mer space with a fixed k and a fixed number of gaps but flexible gap positions, trying to maximize SNP calling sensitivity [14].

Finding a sensitive k-mer shape

Evolutionary algorithms are used to solve optimization problems where the large solution space prevents testing every single solution due to the complexity of the problem. This approach was fitting to find a near-optimal k-mer shape for varkit and was thus implemented.

A fitness function determines the quality of a solution, in our case, how sensitive the k-mer shape is at calling SNPs given a test set of random reads with varying number of reads. In our case, simulating reads only requires distinguishing between mutation and no mutation, while the number of mutations over the total read length determines the similarity to the reference (expressed as ANI). For each iteration, the current shape is assessed regarding SNP calling sensitivity across multiple generated 'reads' with random SNP positions at varying ANI levels between 90% and 99%. It has to be noted that a read of length 100 with 99% ANI does not always contain 99 SNPs. In this case, during read generation, for each position the probability is 1% for it to be a SNP. The assumption with evolutionary algorithms is that closely related solutions perform similarly well given the fitness function and that altering a well performing existing solution has a higher chance of giving us an increase in fitness as opposed to a random solution [14]. The step function defines how I alter an existing solution and is described in the following paragraph.

In the case of varkit, further constraints to the solution require all k-mer shapes to be symmetric and of odd length. Symmetry is important as each k-mer is extracted as the lexicographic minimum between itself and its reverse complement[1] and only uneven lengths were chosen to ensure no k-mer is its own reverse complement. Since k and the number of gaps are fixed, the step function employed here moves the gaps in the k-mer shape, while still obeying the symmetry constraint and the fixed k and number of gaps. If a shape is revisited, the shape is completely randomized in the next step. The step function should i

Given a starting shape, the algorithm assesses the fitness of the shape by building its pattern database with limiting l to reduce runtime complexity (see Section 4.2.2). In our case, the solution space is spanned by all k-mers that satisfy the above restriction. A k-mer shape is a sequence of 'X' and '_' with the regulation that a shape must start or end with an 'X'. 'X' are also referred to as 'take positions' and '_' are referred to as 'gaps'. To construct a shape, consider a string of take positions 'X' with length k. To insert gap positions '_', a fixed number of positions between two 'X' with repetition and without order is picked (since there can be two gaps between two take positions as in 'X__X' and the order in which I select the gaps does not matter). As the shape is symmetric I only need to construct one half of the k-mer shape. The number of possible k-mer shapes at fixed k and g, can be described with

[1] Since varkit uses a canonical representation of nucleotides internally, each k-mer is an integer and the lexicographic minimum for k-mers in string representation is the numerical minimum between the canonical representations of itself and its reverse complement.

Equation

$$NumPermutations(k, g) = \binom{\frac{k-1}{2} + \frac{g}{2} - 1}{\frac{g}{2}} \tag{4.1}$$

For $k = 27$ and $g = 12$, this gives 18564 different k-mer shapes.

4.2.4 Assessing varkit's SNP calling sensitivity

SNP calling sensitivity was assessed with respect to an early version of varkit's database containing custom marker genes from the UHGG database [4] (see Section 4.2.5 for details). The marker genes of 15 random genomes (that are part of the database) were selected. SNPs were inserted to simulate ANIs between 94% and 99%. Reads without errors were simulated with wgsim [2] from the SNP injected marker genes with length 150bp at vertical coverages 1,2,5, and 10. The individual samples were processed using varkit, and predicted SNPs were compare to the known SNPs to calculate SNP calling sensitivity.

4.2.5 K-mer data structure

To efficiently handle and look up k-mers, k-mer based algorithms employ hash tables, as they provide constant time lookup of exact matching k-mers. With k-mers as keys and taxonomic identifiers as values, k-mer based profiling methods often employ a pre-built hash map, storing k-mers of the reference sequences, that is used to identify the query.

I extended this approach: instead of only storing taxonomic identifiers, varkit also stores a gene identifier and gene position for each k-mer to increase sensitivity and precision, both for taxonomic profiling and SNP calling. Due to the lack of a publicly available hash table libraries with a small memory footprint fitting this specific use-case, I implemented a hash map tailored to the needs of varkit.

Varkit needs 46 bits to store the k-mer (k=23), 16 bits for the taxonomic identifier ($2^{16} = 65,536$ different values), 7 bits for the gene id ($2^7 = 128$ different ids, given 120 marker genes), and 12 bits for the gene position ($2^{12} = 4096$). These values were determined to fit the specifics of GTDB r207 marker genes. In total, this makes 46+16+7+12 = 81 bit per cell. In a naïve implementation, key and value would each take 64 bits. This would require 128 bits per cell. A more refined implementation where key and value can take byte defined values and are not restricted to 64-bit each cell would still require 92 bit.

varkit takes this a step further: the hash map reduces the space by storing the key and value in a single 64-bit type and implicitly storing part of the key in the hash map location (Fig. 4.3). With 81 bit for the whole entry - a cell that would otherwise need 92 bit to store - $81 - 64 = 17$ bit need to be stored implicitly.

[2] https://github.com/lh3/wgsim, unpublished

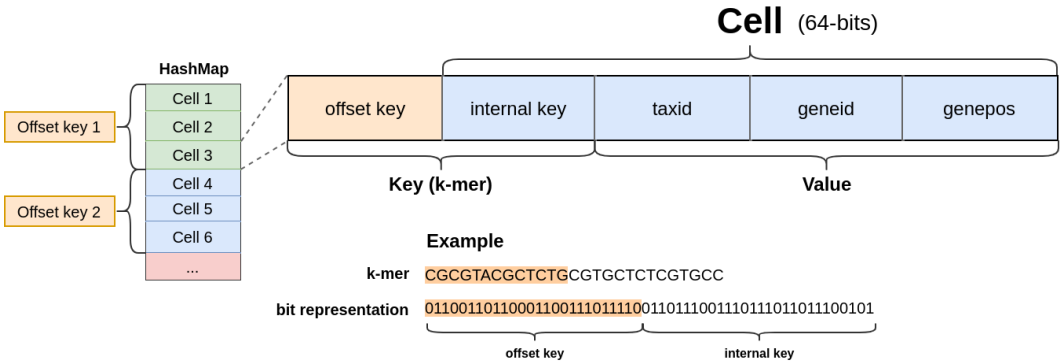

Figure 4.3: The memory layout of varkit's hash map requires 64-bits per cell. The offset key (highlighted in orange in k-mer example) is a part of the key that is implicitly stored in the position such that all k-mers starting with the same prefix with predefined length lay in one contiguous block. Here, the keys in Cells 1-3 all start with the prefix "Offset key 1" and the keys in Cells 4-6 start with "Offset key 2". For each k-mer (offset key + internal key) varkit stores the taxonomic id (taxid), gene id (geneid) and gene position (genepos).

The implicit part is realized with an additional array storing the offset for each possible (implicitly stored) offset key. With 17 bit in the offset key and 64-bits per cell to point to a location in the main data array, the offset table needs 256 MB ($= 2^{25} \cdot 8$ byte $= 33,554,432 \cdot 8 = 268,435,456$ byte) of additional space, while saving 33% of space per cell that would otherwise need 92 bit to store the full k-mer. With growing size, the constant size of the additional offset table becomes negligible and the space savings converge to 33% less space required when compared to the 96 bit implementation without offset (Fig. 4.4).

4.2.6 Building the database

Varkit's database is built from 120 bacterial marker genes of species representatives of GTDB version r207 [208]. Within GTDB, these marker genes are used to construct the phylogeny and are part of the database. The utilized k-mer shape is 'X_XXXXX_XXX__XXXXX__XXX_XXXXX_X', and the SNP calling database was built with a pattern size of 26 and full training depth (see Section 4.2.2). The space for a single data cell in the hash table is shared between k-mer, taxonomic identifer, gene identifier, and gene position, that take up 46 bits, 17 bits, 9 bits, and 14 bits, respectively. 26 Bits of the cell is stored in the offset to reduce overall memory footprint (see Section 4.2.5).

The database is built in two iterations over the reference sequences. The first iteration is for determining the number of entries under each offset key. In the second iteration the values are stored. For all k-mers with the same offset (which are stored in a consecutive region in the hash table), varkit uses the hash function to determine

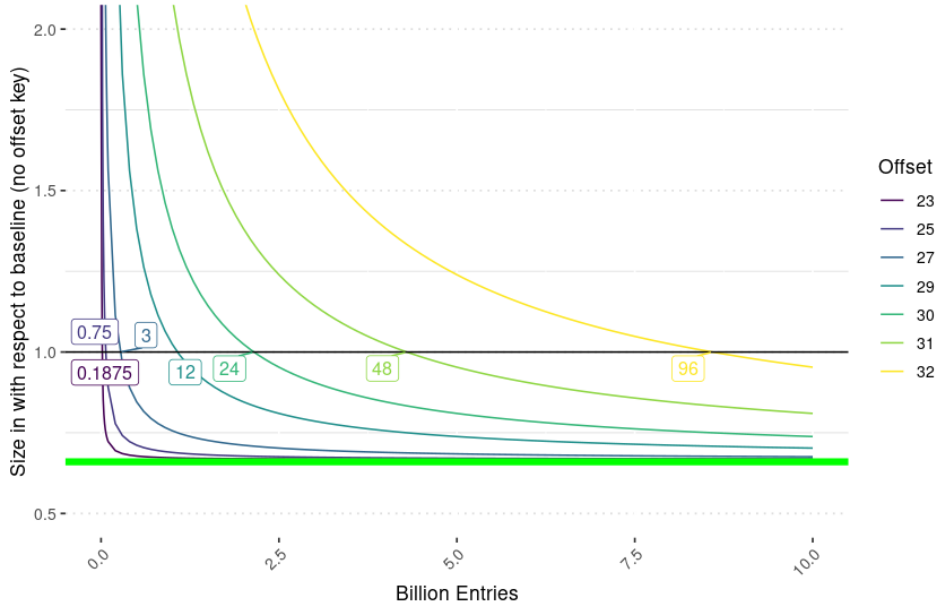

Figure 4.4: Space savings of varkit's hash map with 64 bits per cell + offset in extra data structure compared to a hash table with 96 bits without offset as baseline. With offset 32, the effective cell size is the same between both data structures. The labels denote the GB size at the break-even point for each offset, i.e. what the size of the map is when it is equally space efficient as the implementation without offset.

the position within an offset. Varkit uses the same hash function as Kraken2 [3].

A k-mer is stored with its taxonomic id, gene id, and position within the gene. If a k-mer is not unique, the gene id and gene position is dropped, and the taxonomic identifier is set to the LCA of all references it occurs in. A collision happens if two different k-mers with the same offset have the same hash. Collisions are resolved with Robin Hood hashing [42]. Briefly summarized, Robin Hood hashing is a way to resolve collisions without external data structures, where entries in the table are moved to minimize the distance to their original hash. If a k-mer hashes to a cell that is already occupied, the distance of both entries to their original hash is compared. If the distance of the to be inserted k-mer is larger, Robin Hood hashing replaces the entry with the k-mer, and searches for a new position for the previous value by moving to the following cell and repeating the last step. To ensure good performance, the hash table needs to be larger than the amount of entries. This is referred to as load factor - varkit uses a default load factor of 0.75, meaning that only 75% of the table is filled with entries.

This is the standard database used for the benchmarks in section 4.3.1. For plot 4.10, an earlier version of varkit's database was used. Briefly, this database was based on marker genes from species representative genomes of the UHGG database (unified human gastrointestinal genomes) [4]. Marker genes were selected *de novo* by first

[3] https://github.com/DerrickWood/kraken2/blob/4cbdc5fac92d0a19d76ce68d6633d7ff794a1587/src/kv_store.h#L23

predicting genes of all genomes (n=204,938) (prodigal with '-meta') and subsequent clustering of the sequences based on 95% sequence identity and minimum alignment length with mmseqs2 [268] ('mmseqs linclust -id 0.95 -min-aln-len 100'). For each species cluster (n=4,644), marker genes were originally selected based on uniqueness in their genus (number of genomes in species with gene g divided by all genomes in genus) and coreness (number of genomes of species s with gene g divided by all genomes in the species) and the top 300 marker genes were selected for a species. This approach was dropped early, as the integration of GTDB marker genes appeared to be a more promising approach regarding quality of marker genes and taxonomic diversity.

4.2.7 Taxonomic classification and SNP detection

To profile a metagenomic dataset, for each read all k-mers are extracted and looked up in the database (see Fig. 4.5 A). Similar to Kraken [308], k-mers are counted up in a tree structure (see Fig. 4.5 B). The tree has the same structure as the taxonomic tree, containing all taxa that have been hit by a k-mer, while keeping intermediate nodes with zero hits. Each node holds a counter for the number of k-mers it was hit by. Varkit then classifies a read by traversing the subtree from the highest node (least specific), and traversing closer to the leaves by taking the path with the most hits in the subtree (see Fig. 4.5 C). When traversing the tree, varkit considers the two variables 'min_confidence' and 'min_hits'. Varkit traverses down the branch with the highest count if the hits in the subtree divided by all hits is $> min\_confidence$ and the number of hits is $> min\_hits$. If the condition is not fulfilled, a read (-pair) is classified as the current node in the taxonomic tree.

The k-mer hit-pattern across the whole read is processed according to section 4.2.2 (see Fig. 4.5 D). K-mer hits to different leaves (species) as well as k-mer hits on higher taxonomic levels must be disentangled, to get SNPs with respect to the classification. A k-mer hit is included as '1' in the pattern, if it matches with the read classification or it is on its the root-to-leaf path. Based on this pattern, sub-patterns are extracted and used to get SNP positions which are written out subsequently.
To get from read classifications to abundance profiles, varkit collects all read classifications across taxa and marker genes. In the current version of varkit, only reads classified on species-level are further processed. For each species that receives hits, varkit estimates the mean ANI, based on predicted ANI values for each read using hit k-mers vs total k-mers.
Given a certain amount of reads mapping to a species, we expect a certain 'spread' across the genes if it is a genuine signal. We can quantify this as Expected Gene Presence and Expected Gene Presence Ratio which are calculated for species t as

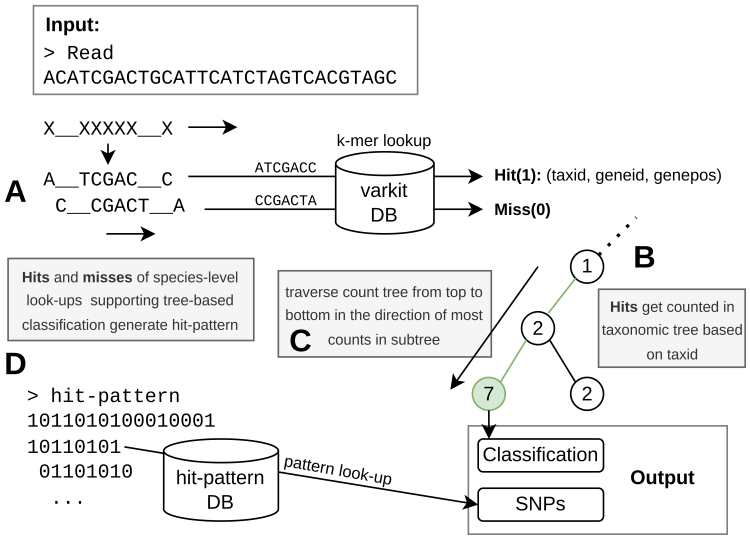

Figure 4.5: Basic workflow of varkit's classification and SNP detection algorithm. A, extract all k-mers from read based on the k-mer shape. Look up each k-mer in pre-built database, which contains k-mers from reference sequences. If the database has an exact matching hit, report taxid, geneid, and genepos, otherwise report a miss. B, hits are counted in the taxonomic (sub)tree and increment the count for that node by one. C, to classify the read, all hits are counted in the tree. The tree is then traversed from top to bottom (bottom is species-level) and the path is determined by the subtree with the higher total counts. D, hits and misses from the k-mer database lookup forms hit-pattern. If the classification result is on species-level, sub-patterns are looked up in a separate database to receive SNP positions between read and reference. Varkit reports the taxonomic classification as well as SNP positions for each read.

follows.

$$\textbf{ExpectedGenePresence(t)} = (1 - (\frac{genes(t) - 1}{genes(t)})^{reads(t)}) \cdot genes(t) \quad (4.2)$$

$$\textbf{ExpectedGenePresenceRatio(t)} = \frac{ExpectedGenePresence(t)}{PresentGenes(t)} \quad (4.3)$$

In varkit, for a species to be predicted present, the expected gene presence ratio must be higher than a threshold e, mapped read count must be higher than a threshold r, and the median ANI must be higher than threshold a. By default the values are $e = 0.6$, $r = 100$, and $a = 0.95$; used like this in the profiling benchmark. The abundance is calculated as the median abundance across all genes.

4.2.8 Implementation details

Varkit is implemented in C++ using the std++20. Internally, k-mers are represented as integers with each base occupying exactly 2 bits. The mapping for this 2-bit representation is A=00, C=01, G=10, T=11. Any other ambiguous bases are converted to A=00 by default. Extracting unspaced k-mers is a lot faster than extracting

spaced k-mers as for unspaced k-mers, simple bit-shift operations can be used.

For unspaced k-mer extraction, all k-mers smaller or equal than 32 nucleotides fit into a 64-bit integer. Starting with a k-mer in 2-bit representation, we need to bit-shift to the left by two (the bits of one nucleotide), apply logical 'and' with a mask of one-bits of the size $k \cdot 2$ to cut the overhanging two bits we created by shifting left, and appending the new nucleotide by applying logical 'or' with the two bit encoding of the new nucleotide. The runtime of this algorithm is not dependent on k, the number of take positions in the k-mer, but instead has a constant runtime as once the initial k-mer is computed, each new k-mer just requires the fixed number of operations to add the next nucleotide (see Equ. A.1) .

To extract spaced k-mers, varkit uses an iterative algorithm to speed up extraction similar to an algorithm proposed by Petrucci et al [213]. Briefly summarized, each new k-mer, given a certain k-mer shape, can be constructed from a number of previously extracted k-mer by using bitwise operations. The number of previous k-mers needed depends on the size of the largest gap in the k-mer shape.

4.3 Results

4.3.1 SNP calling sensitivity

SNP calling usually requires alignment of query and reference, however, with the introduced algorithm, I can leverage k-mer hit patterns to determine SNP positions. Alignment is costly, and being able to directly go from k-mer hits to SNP positions eliminates a computationally expensive step. As introduced in Section 4.2.3, I hypothesize that SNP calling sensitivity with k-mer hit patterns is dependent on the spaced k-mer shape and the pattern length p, similar to sensitivity in match finding. With selected fixed ks of 23, 25, 27, total spaces of 2, 4, 6, 8, 10, 12, and a fixed $p = 16$ I explored the k-mer space. Given the space limitations of the underlying data structure as described in 4.2.5, I chose the k-mer size 27 for the following, 12 spaces for sensitivity and set $p = 16$ and $l = 16$ to limit the runtime complexity of the fitness function. p is the pattern size and l limits the 'training depth' as explained in section 4.2.2. The fitness function measures the mean SNP calling sensitivity for random sequences with simulated ANI values between 95%-99%. The best shape refers to the shape with the highest SNP calling sensitivity with a given set of constraints that are the size k (Number of 'X') and the number of spaces s (Number of '_')..

In 9,623 iterations I assessed the fitness of different k-mer shapes (Fig. 4.6). This range is appropriate, as this is the expected ANI between reads and reference marker genes in varkit's database. The shape 'XXXX_XX_X_XXX_X__XXXXX__X_XXX_X_XX_XXXX' at iteration 4184 performs best among all tested shapes across all itera-

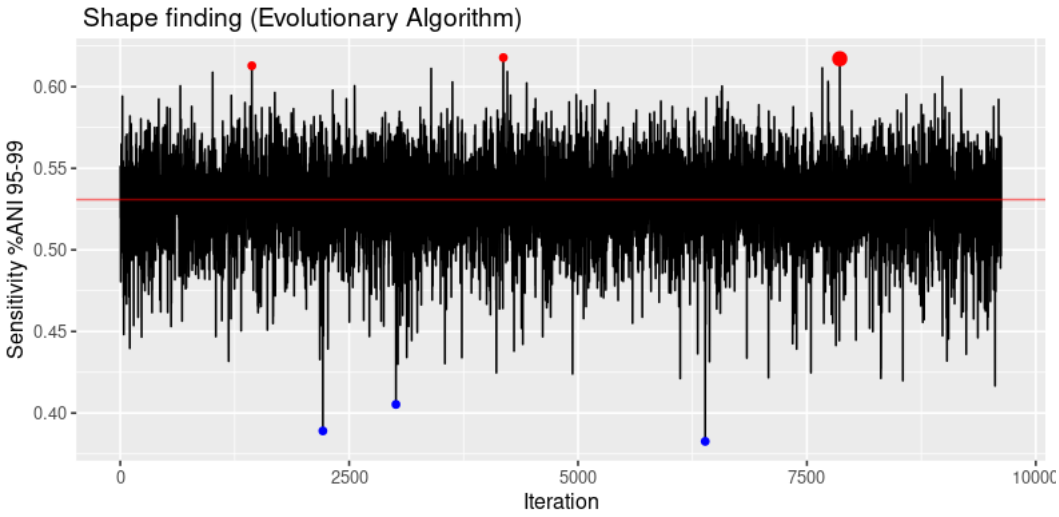

Figure 4.6: In 9623 iterations, each iteration tests the fitness of a new k-mer shape and then moves to the next (see Section 4.2.3). Fitness is defined as mean SNP sensitivity for ANIs between 95% and 99% (y-axis). The red dots are the three best performing shapes based on their fitness. From left to right these are 'XXXX_XX_X_XXX_X__XXXXX__X_XXX_X_XX_XXXX', 'XXXX_XX_XXXX__XX_X_X_X_XX__XXXX_XX_XXXX', and 'XXXX_XX_X_XXX_XX_X_X_X_XX_XXX_X_XX_XXXX'. The blue dots mark the lowest scoring k-mer shapes and from left to right these are 'XXXXXXXX_X_X_X_X_X_X_X_X_X_X_XXXXXXXX', 'XXXXXXXXX_____XXXXXXXXX_____XXXXXXXXX', and 'XXXX_XXXXX_X_X_X_X_X_X_X_X_X_XXXXX_XXXX'.

tions and has a mean SNP sensitivity of 0.6178524. This shape is followed by 'XXXX_XX_XXXX__XX_X_X_X_XX__XXXX_XX_XXXX' and 'XXXX_XX_X_XXX_XX_X_X_X_XX_XXX_X_XX_XXXX' with 0.6128544 and 0.6087724 respectively. The worst shapes are 'XXXXXXXX_X_X_X_X_X_X_X_X_X_X_XXXXXXXX', 'XXXXXXXXX_____XXXXXXXXX_____XXXXXXXXX', and 'XXXX_XXXXX_X_X_X_X_X_X_X_X_X_XXXXX_XXXX' with a mean sensitivity of 0.3826254, 0.3890564, and 0.4052756. On this data I observed that, 1) shapes with the same length and a fixed number of spaces perform very differently (difference of 0.23 between best and worst in fitness) and 2), navigating the shape space with our current step function does not lead to a gradual improvement of the solution, but is rather a random traversal (see Section 4.2.3).

I further computed this for k=23 and s=8 (Fig. 4.7). Here, the best shape across all iterations regarding mean ANI for values between 90% and 99% is 'X_XXXXX_XXX__XXXXX__XXX_XXXXX_X' (mean ANI of 0.4904544) and the worst is 'XXXX_X_X_X_XXXXXXXXX_X_X_X_XXXX' (mean ANI of 0.6849034). For mean ANI value between 95% and 99%, the best shape across all iterations for this k and s is 'XXXX_XXX_XX_XX_X_XX_XX_XXX_XXXX'

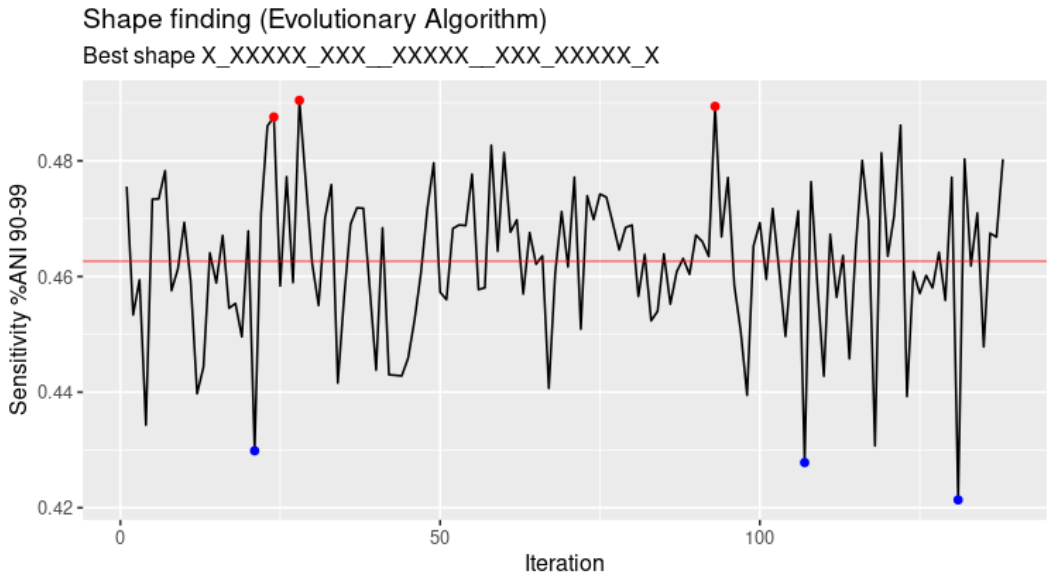

Figure 4.7: In 138 iterations, each iteration tests the fitness of a new k-mer shape and then moves to the next (see Section 4.2.3). Fitness is defined as mean SNP sensitivity for ANIs between 95% and 99% (y-axis). The red dots are the three best performing shapes based on their fitness. From left to right these are 'XXXXX__XX_XXXX_X_XXXX_XX__XXXXX' (mean ANI of 0.4875677), 'X_XXXXX_XXX__XXXXX__XXX_XXXXX_X' (mean ANI of 0.4904544), and 'XXXX_XXX_XX_XX_X_XX_XX_XXX_XXXX' (mean ANI of 0.4894113). The blue dots mark the lowest scoring k-mer shapes and from left to right these are 'XXX____XXXXXXXXXXXXXXXXX____XXX' (mean ANI of 0.4298529),'XXXXX_XXXXX___XXX___XXXXX_XXXXX' (mean ANI of 0.4278185), and 'XXXX_X_X_X_XXXXXXXXX_X_X_X_XXXX' (mean ANI of 0.4213629).

(mean ANI of 0.7685434, Fig. A.6).

Properties of k-mer shapes correlating with SNP detection sensitivity

Next, I focused on which properties in gap placements improve fitness. The first hypothesis is that a higher number of different space distances, which is the pairwise distance between any two spaces in a k-mer shape, improves the performance, as this allows for k-mer hits in the presence of two SNPs with various distances. Additionally, since our test data contains more synonymous mutations to model realistic patterns more accurately, mutations at distances that are multiples of three, occur more frequently. Hence, I also hypothesize that the number of gaps that are a multiple of three apart, is also indicative of the SNP calling sensitivity. For all 9,623 k-mer shapes I tested a) the number of different pairwise gap distances, b) the number of pairwise gap distances that are a multiple of 3 and c) the sum of both values (Fig. 4.8).

With linear modeling the R^2 for a) is 0.13, for b) is 0.02, and for c) is 0.19.

This confirms that both pairwise gap distances and the number of three apart gaps indicate to a degree whether the shape will be performant. While further analyses could be done to analyse gap distances and spacing patterns within spaced k-mers with regard to the proposed SNP calling algorithm, it is likely that potentially better solutions will only result in negligible improvements of <1% sensitivity. In the end, I used the shape 'X_XXXXX_XXX__XXXXX__XXX_XXXXX_X' with k=23 and s=8 (best shape mean ANI 90%-99%) to build the database for varkit used in Chapter 4.3.1. I chose k=23 as a good compromise between higher SNP calling sensitivity, while not losing too much precision for species profiling.

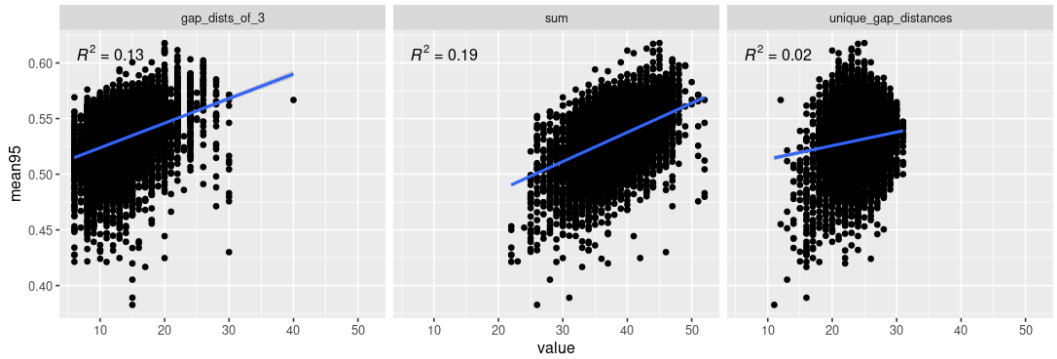

Figure 4.8: Depiction of how three properties of the k-mer shape affect SNP calling sensitivity at ANI 95%. Panels show on the x-axis: number of gaps with distance three (gap_dists_of_3), number of unique pairwise gap distances in a shape (unique_gap_distances), and the sum of both values (sum). The y-axis shows the mean SNP sensitivity. Data is from shape finding with k=27 and s=12 (Fig. 4.6).

Impact of pattern-database training depth on SNP detection

The sensitivity of the pattern-based SNP calling depends on the k-mer shape and the sub-pattern length. Figure 4.9 shows SNP calling sensitivity with respect to different k-mer shapes, varying in k and number of spaces. As mentioned in 4.2.2, the pattern size p and 'training depth' l can be limited to save building time of the database when needed. In this section I will unwrap how p and l affect the sensitivity of a shape. For this, I trained a shape database for four very different k-mer shapes. The first shape is a 23-mer without gaps, the second shape a 23-mer with 44 gaps ('_'), the third shape a 23-mer with 16 gaps, and the fourth shape a 9-mer with 20 gaps.

For a more direct comparison, I tested k-mer shapes trained at different pattern sizes and included performance assessments for every stage of training. Training two different k-mer shapes 'XXXX_XX_XXX_X__X_XXXXX_X__X_XXX_XX_XXXX' (k=27, s=12) and 'X_XXXXX_XXX__XXXXX__XXX_XXXXX_X' (k=23, s=8) with pattern sizes 16 and 32 shows how pattern size and k-mer length affect the SNP calling sensitivity (Fig. 4.9). For both k-mer shapes and low-training depth,

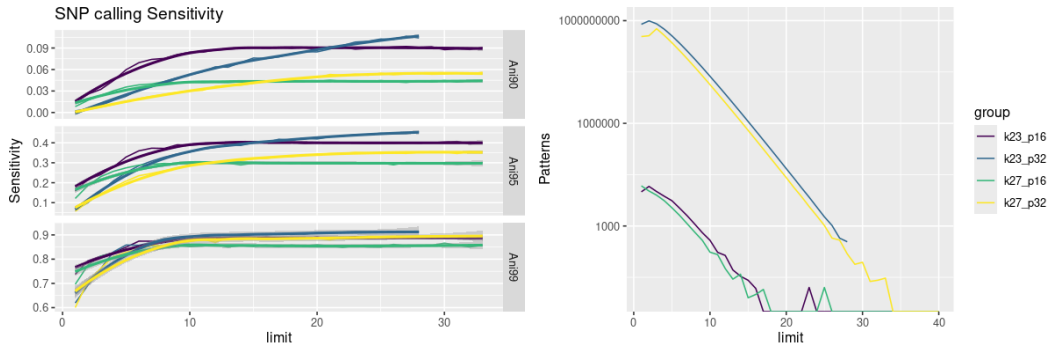

Figure 4.9: Comparison for training the pattern database with two different shapes 'XXXX_XX_XXX_X__X_XXXXX_X__X_XXX_XX_XXXX' (k=27, s=12) and 'X_XXXXX_XXX__XXXXX__XXX_XXXXX_X' (k=23, s=8) for two different pattern sizes 16 and 32. The left plot shows SNP-calling sensitivity for ANIs 90%, 95% and 99% with respect to the training depth 'limit' shown on the x-axis. The right plot shows how many patterns (y-axis) were added to the database with increasing training depth (x-axis).

p=16 has a higher sensitivity than p=32. However, there is a training depth where the longer pattern size surpasses the short pattern size in sensitivity. This point comes later with lower ANI and thus suggests that shallow training depths increases detection of SNPs for high similarity and deeper training adds shapes that improve the SNP calling for lower ANI regions. This effect is more pronounced with the shorter pattern and is likely due to the fact that the difference $p-(k+s)$ is smaller.

Assessing SNP detection sensitivity with synthetic metagenomes

Next, I assessed SNP calling sensitivity with varkit given simulated reads from marker genes with injected SNPs (Fig. 4.10 A). For varkit, SNP calling sensitivity mostly depends on the ANI to the reference genome and quickly declines with growing distance to the reference. At 99%, over 90% of all SNPs are recovered, but at 95% ANI this rate drops to ∼50%. This is in line with what already emerged during the building of the pattern database in Figure 4.9. Figure 4.10 B reveals a species-specific signal. Across all 15 genomes that were used to simulate reads from, both genomes from *g__Collinsella* stand out with a mean sensitivity of 33.078±13.456 and 35.121±14.41, respectively. Vice-versa, the mean FP-rate for SNPs is high with 7.503±2.69 and 10.006±2.648, respectively. The mean sensitivity and FP-rate across all genomes is 58.095±20.861 and 1.824±3.086. This is caused by the low rate of species-level k-mers in the database. For both genomes within *g__Collinsella*, 66% and 69% of all k-mers are above species-level and as mentioned in section 4.2.7 the SNP detection algorithm cannot disentangle, whether a genus hit is actually present in the detected species or not. In conclusion, the SNP detection performance depends on the coverage, ANI to the reference, and the number of species-level k-mer. In the next section, I will further showcase how varkit performs in taxonomic profiling

benchmark.

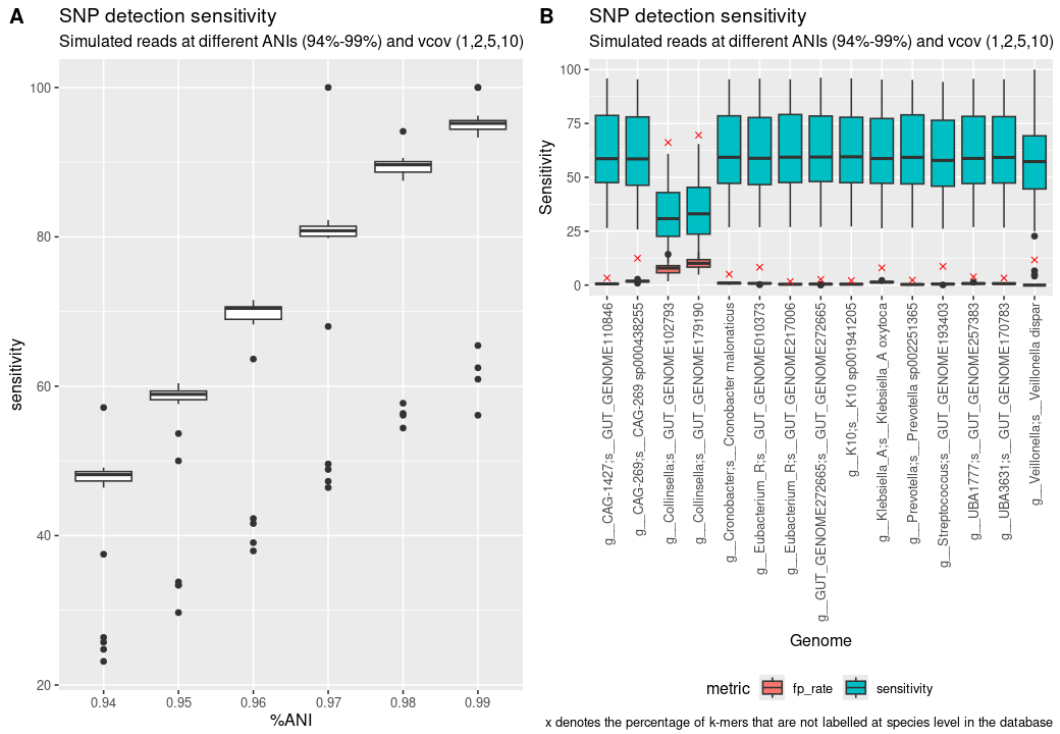

Figure 4.10: SNP calling sensitivity at different ANI rates to reference, stratified by genomes. Random species-representative marker genes were selected and SNPs were introduced to match certain ANI values (94, 95, 96, 97, 98, 99% and vertical coverages 1,2,5,10). Reads were then simulated on the new references. Sensitivity is the number of SNPs detected divided by all inserted SNPs. GUT_GENOME227824 belongs to the species *s__Collinsella sp900761165*. The x-axis denotes the percentage of species-level k-mers for this species. see Section 4.2.4 for more details.

4.3.2 Taxonomic profiling with varkit

For species-level benchmarking, I compared varkit's performance to MetaPhlAn 4 and mOTUs3 on GTDB, with adjusted scores using benchpro to analyse the data. Varkit was tested against MetaPhlAn 4 and mOTUs3, all against GTDB r207 profiles on the CAMI HumanToy dataset (Airways, Gastrointestinal, Oral, Skin, and Urogenital).

Across all datasets for species (adjusted), varkit has the second highest mean F1 with 0.915±0.059 (Fig. 4.11 A). Varkit performs best on the Gastrointestinal samples, with both a high mean sensitivity and precision (0.977±0.027, 0.972±0.039, respectively). Varkit's lowest F1 score is for the Oral dataset(mean F1 of 0.824±0.031) as a result of the low mean precision (0.737±0.052).

Further, the oral datasets contains reads from multiple strains of *s__Streptococcus suis*, which, across all oral datasets, cause many FP predictions in the surrounding closely related species (Supp. Fig. A.7). In fact, ∼63% of varkit's

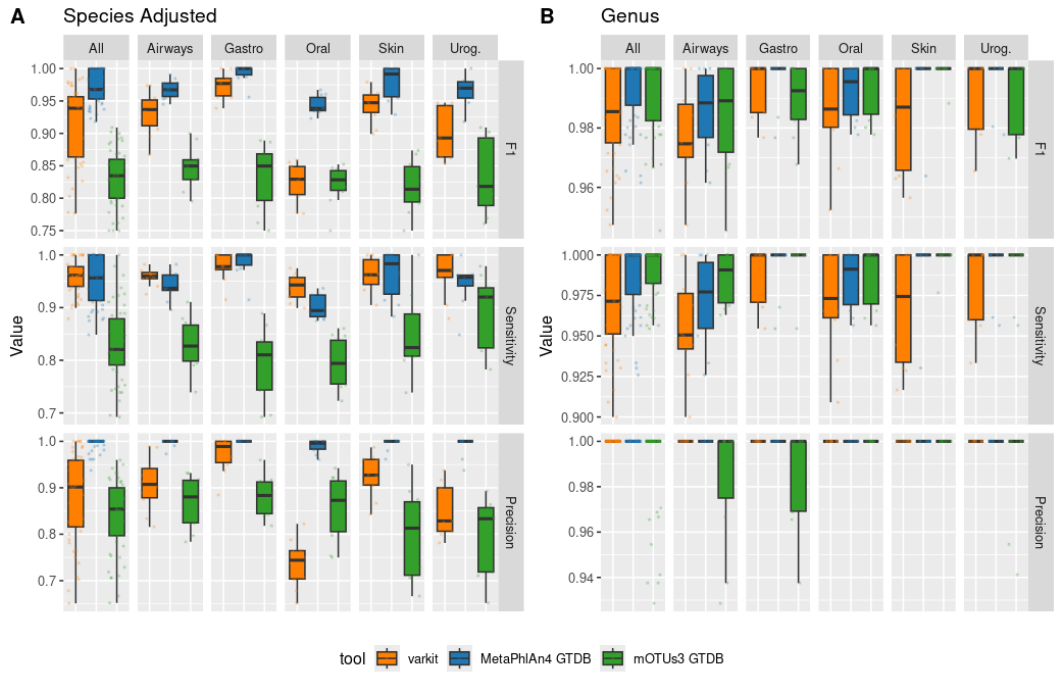

Figure 4.11: Profiling performance across samples, environments, and tools measured as F1-score, precision, and sensitivity. The boxplots represent different tools and each data point is a sample. From left to right, the column panels are F1-Score, precision, and sensitivity and the row panels stratify between different environments with the top row being the summary across all environments.

FPs in the Oral datasets are species of the genus *g__Streptococcus*. Species of the genus *g__Neisseria* are responsible for 34 FPs (15%) in the Oral datasets, 16 FPs (28%) in Airway, and 11 FPs in Skin, suggesting a similar issue as *g__Streptococcus*. Considering all datasets, other genera with frequent FPs are *g__Haemophilus* (37), *g__Campylobacter_D* (15), *g__Lactobacillus_D* (15), and *g__Escherichia* (13). Varkit performs particularly well on the Gastrointestinal dataset with only 10 FP and 11 FN species. On genus level and across all datasets, varkit and MetaPhLAn 4 GTDB have perfect precision while mOTUs3 GTDB has 10 FPs across all datasets (Fig. 4.11 B). However, overall varkit performs worse than the other two tested tools, with a mean F1 of 0.985±0.015 and a mean sensitivity of 0.972±0.03.

As mentioned, Kraken2+Bracken was omitted from this benchmark, as it had already been described and assessed in Chapter 3.3. In comparison, varkit has a mean F1-score of 0.915±0.059 across all datasets, and Kraken2+Bracken with NCBI and GTDB have a mean F1-score of 0.451±0.15 and 0.427±0.157, respectively. Further, as varkit uses a marker gene based approach while Kraken2+Bracken use whole genomes as reference, I focused on the comparison to MetaPhlAn 4 and mOTUs3.

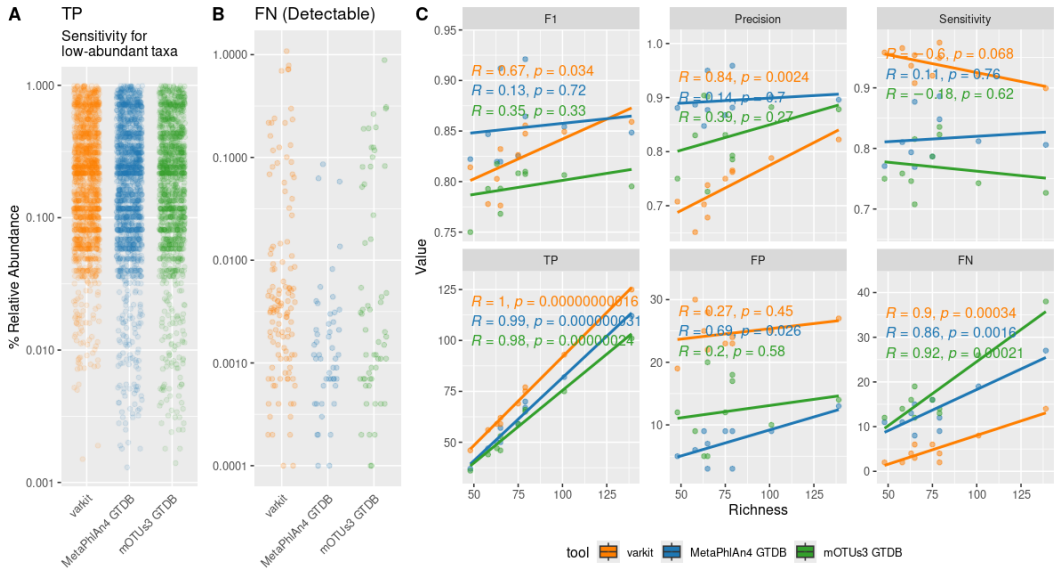

Figure 4.12: A, relative abundance for TPs for different tools coloured by dataset. Points on the lower end show the sensitivity of the tool to detect low abundant taxa. B, relative abundance for FNs for different tools, only showing taxa that are covered by the tool database. Points are coloured by dataset and higher end points show that a tool failed to detect higher abundant taxa.

Detection performance at difference taxa abundances and sample richness levels

When further looking into the detection ability for low abundant species (<0.01% relative abundance), I found that varkit only detects 19 TPs, while MetaPhlan 4 GTDB detects 45, mOTUs3 GTDB detects 46 (Fig. 4.12 A). Varkit also has 16 FN species above 0.1% relative abundance (MetaPhlan 4 GTDB has 0 FN, mOTUs3 GTDB has 11 FN) (Fig. 4.12 B). For varkit, 8 out of the 16 FNs, and for mOTUs3 GTDB 7 out of 11 were species of the genus *g__Streptococcus*, which were previously identified as a cause for FPs in varkit.

Next, I looked at how sample richness affects profiling performance for the Oral dataset, which metagenomes had a richness between ∼50 and ∼135 different species. For varkit, an increase in F1 and precision with sample richness is significant (p=0.034 and 0.0024, respectively with Pearson Correlation). The main reason for this is *s__Streptococcus suis*, which, when present in a sample, causes FP predictions in the whole clade (Supp. Fig. A.7). For varkit, there is a higher increase for TPs than FPs with increasing richness, which overall leads to a higher precision. A possible conclusion is therefore that richness affects sensitivity more than precision, as a higher richness causes more taxa to be below the abundance detection threshold. To further expand on the abundance detection threshold, I filtered reads at different abundance thresholds (Supp. Fig. A.8). Varkit's F1 score improves gradually with an abundance threshold and peaks at a threshold of 0.05% with a mean F1 of

0.921±0.053 (increase from 0.915±0.059 unfiltered).

FP predictions with respect to their phylogenetic context

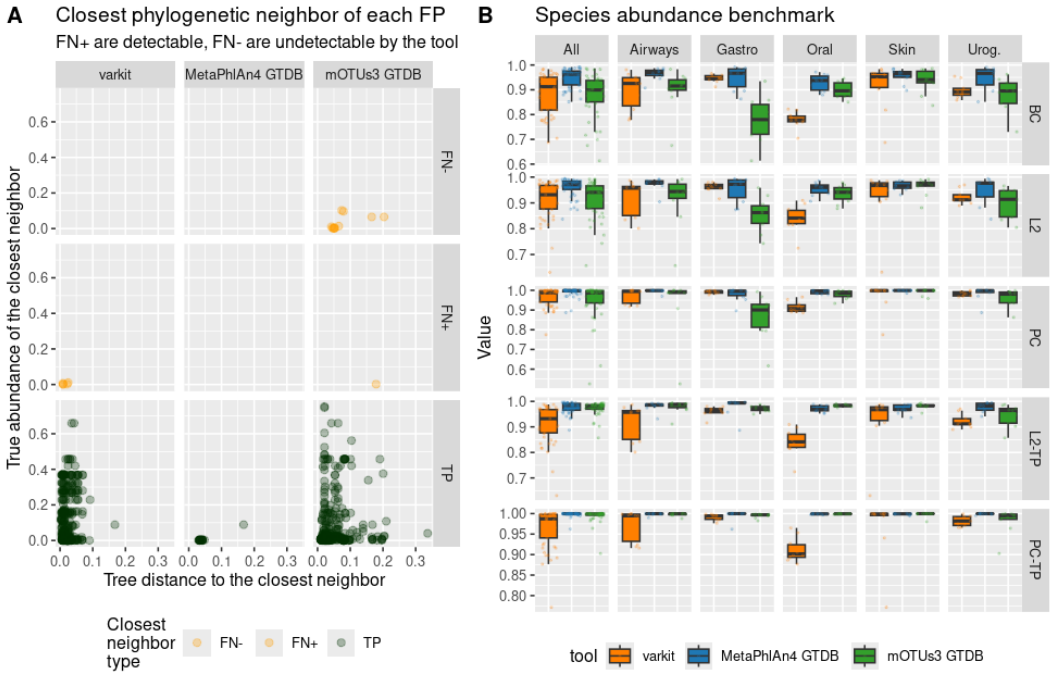

Figure 4.13: A, false positive species in context of their closest TP or FN neighbour in the phylogenetic tree. FNs are split into detectable (FN+) and undetectable (FN-) based on whether the tool has this taxon in their database. B, Abundance prediction benchmark on species-level for all datasets. Metrics are Bray-Curtis Similarity (BC), 1-L2 error (L2), Pearson Correlation (PC), 1-L2 error only on TP taxa (L2-TP), and Pearson Correlation only on TP taxa (PC-TP).

I already established in Chapter 3.3.2 that FPs are often a byproduct of closely related higher abundant TP taxa (Fig. 4.13 A). I found that all FP predictions (except for one) are within 0.1 tree distance of a TP, and 58% of those TPs have a relative abundance above 1%. The mean tree distance from TPs is 0.022±0.017 with a mean abundance of 10.2±13.7%. Five FPs are closest to a FN+ (FN taxa that are present in the tool database) in the tree, three of which are *s__Haemophilus influenzae*, that are falsely detected as *s__Haemophilus influenzae_E*, *s__Haemophilus influenzae_F*, or *s__Haemophilus aegyptius*.
Varkit's FPs are mostly caused by unclear species boundaries for certain closely related species, a source of error that also seems to affect mOTUs3 GTDB. I also observed a larger spread in phylogenetic distance of FP taxa and their closest TP neighbour (0.0 - 0.1 for varkit and 0.0 - 0.2 for mOTUs3 GTDB) between varkit and mOTUs3. While varkit natively supports the GTDB phylogeny, mOTUs3 has a custom species clustering. Hence, phylogenetic distances between species can differ between the two phylogenies, depending on the phylogenetic approach.

As for abundance prediction, varkit scores last in the dataset summary across metrics (Fig. 4.13 B). By comparing between metrics applied on all taxa (TP, FP, FN) and metrics only applied to TPs, I find that varkit does not benefit from only considering TPs. Especially on the oral dataset, varkit has a lower score across all metrics. This shows that in some cases, varkit's abundance predictions are incorrect for TP taxa. Out of 15 TP predictions with an absolute difference between predicted and true abundance > 0.1, 7 of those TP taxa are *s__Streptococcus suis* and 4 are *s__Neisseria meningitidis*. Hence, the same species that are responsible for FP predictions also cause wrong abundance predictions and have closely related neighbouring species.

4.3.3 Runtime

In the following speed and memory benchmark, Kraken2+Bracken is included, as it is often used as a benchmark reference for fast tools. Further, a thorough benchmark on its taxonomic profiling performance was already conducted in chapter 3. On a benchmark using the 10 samples from CAMI Airways (see Section 3.2.7), varkit was the second fastest with a runtime of 12min (Fig. 4.14). Kraken2+Bracken came in first with 8min 30 sec followed by mOTUs3 taking 1h 59min and MetaPhlAn 4 taking 1h 28min. Part of varkit's speed is because it is the only tool that is able to run multiple samples in the same run and hence does not have to reload the database for each sample. Reloading the database still takes some time, even if it is cached from the previous run. StrainPhlAn 4 takes the longest with ∼12h 30min. Varkit was approximately 7.3 times faster than MetaPhlAn 4 and approximately 10 times faster than mOTUs3. The memory consumption was highest for varkit with 85 GB followed by Kraken2+Bracken with 77 GB. MetaPhlAn 4 took 19 GB, and mOTUs3 and StrainPhlAn 4 ran with only 6 GB of memory.

4.4 Discussion

In this chapter, I presented varkit, a k-mer based taxonomic profiler and strain-resolved tool. Varkit uses the GTDB marker genes (r207) as reference and presents a novel method to determine SNP positions from k-mer lookup patterns, without alignment to the reference. The species-level benchmarks revealed that varkit's performance is competitive with the other tested tools. Especially compared to Kraken2+Bracken [307, 159], varkit performs well and is shown to be closer to the performance of the two alignment-based tools, MetaPhlAn 4 and mOTUs3 in benchmarks. Other benchmarks have already demonstrated that many k-mer based tools such as Kraken, Kraken2 or CLARK [200] offer great speed and sensitivity, but struggle with precision [242, 177] . The main reason for this is that k-mer based

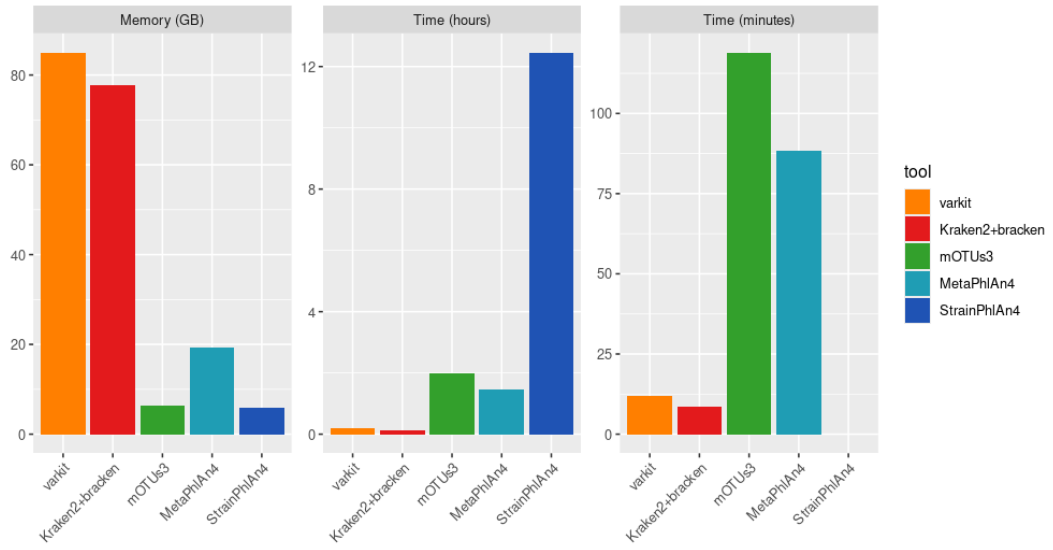

Figure 4.14: Runtime and memory analysis of varkit with respect to other taxonomic profilers and strain-resolved tools. The benchmark was done on a single node with no interfering input and output using 16 cores. The tools were run on 10 samples from CAMI Airways with 2x 5GB uncompressed paired-end reads (see Section 3.2.7 for more details).

tools like Kraken [308] and Kraken2 often use databases built from whole genomes. Genomes in public databases can exhibit contamination and are further often subject to HGT [46]. As a result, many FP hits emerge from using the whole genome information without curating the database in some way. KrakenUniq [32], follows a different approach to estimate the coverage of unique k-mers across the genome with a HyperLogLog data-structure to minimize false positive predictions, but has been shown to perform inferior to mOTUs2 and MetaPhlAn 4 in a recent benchmark [219]. Varkit instead stores marker genes with positional information, which allows for better quality control (see Section 4.2.7). By storing the positional information, varkit's approach is closer to the seeding phase of aligners and alignment-based tools and exhibits much higher precision (see also Chapter 3). Further, by using universal marker genes which are rarely subject to HGT [305], varkit eliminates a potential source for FP predictions. However, a downside is that due to its specialized approach, varkit does not allow for custom databases, which is one of the major advantages of other k-mer based tools.

When comparing varkit to Kraken2 in terms of memory consumption, it's evident that both have databases of the same size. However, varkit stores short regions of species representatives, whereas Kraken2 retains complete information from 203,948 sequences [4]. Kraken2 achieves this by sub-sampling k-mers (minimizer approach) and storing a lossful representation of k-mers; a strategy not viable for varkit, as the pattern-based SNP detection requires hit or miss information for every single

[4]Kraken Standard database from 12/01/2024

k-mer lookup. A fundamental issue with varkit lies in the selection of the parameter k. The results for SNP calling sensitivity for different k-mer shapes suggest that a shorter k-mer size is advantageous. While this is true for SNP calling, it is not true for taxonomic classification [307]. Varkit's hash table stores the LCA for each non-unique k-mer, and the proportion of unique k-mers for each species decreases with smaller k. This decreases the species-specific signal of k-mers and increases the false positive rate. When optimizing for one application, the other will suffer - the current results show that while the profiling performance is comparable to mOTUs3, the SNP detection sensitivity suffers for species with a low count of unique k-mers. Without substantial changes to the data structure and approach, this trade-off leads to unsolvable problems. A solution would be to store each k-mer with all its locations, retaining the exact information for non-unique k-mers. However, this would inevitably increase the database size, as it would require storing every k-mer. Moreover, it would shift varkit closer to an alignment-based approach, necessitating significant algorithmic changes to accommodate this shift.

Other k-mer based strain-level tools are either reference-based, or compare between whole genomes as opposed to short-read sequencing data. FastANI [111], for example, approximates ANI for whole genomes by breaking down each sequence into equally sized large segments, for which the similarity is approximated through overlapping k-mer sets using MashMap [122]. This speeds up ANI estimation for whole genomes significantly, but is an unfeasible approach for short-reads. Skani has a similar approach, but prevents ANI biases arising from comparing sequences with a great size difference [222, 254]. Kmer2SNP [149] is a reference-free SNP caller, which analyses the k-mer frequency distribution to identify heterozygous SNPs and is thus only applicable to diploid organisms. FastGT [202] identifies SNPs from short-reads, but only works with a pre-built k-mer database. Similarly, GT-Pro [250] is a meta-genotyper for sequencing data from the human gut and utilizes a pre-built database of k-mers, selected based on co-occurrence patterns across genomes and linkage disequilibrium. Varkit thus presents a novel method for alignment-free detection of *de novo* SNPs. While varkit's conceptual approach is intriguing, its lack of practicality for the reasons mentioned necessitates significant modifications for it to be effectively usable in practice.

4.5 Author contributions

I conceptualized, developed, and implemented varkit by myself, under the guidance of Falk Hildebrand. The idea to utilize k-mer lookups to infer the position of SNPs came from Falk Hildebrand. I then conceptualized and implemented the idea.

Chapter 5

Alignment-based taxonomic profiling and strain-resolved metagenomics using protal

5.1 Introduction

As established in the introductions of the previous Chapters 3 and 4, the amount of publicly available metagenomic data is rapidly increasing. Facilitated by *de novo* assembly pipelines and the subsequent characterization of unknown species, current reference databases such as GTDB have immensely expanded their taxonomic coverage over the past years. This demands fast and precise taxonomic profilers and strain-resolved tools, to (re-)analyse existing as well as new metagenomic data, with respect to the expanded taxonomy now becoming available.

In the previous Chapter I introduced varkit as an attempt to tackle these issues with a novel k-mer based approach that allows for strain-level resolution based on k-mer hits in a database. I showed that varkit's taxonomic profiling is largely on par with existing tools, while benefiting from covering the whole GTDB taxonomic space. While the strain-level performance was fast, the achieved resolution was inadequate for the applications I envisaged. Additionally, varkit's high memory requirements (80-90Gb) reduce accessibility and increase the runtime for small datasets. Leveraging the accumulated knowledge and ideas that were not fit for varkit, I implemented an alignment-based approach, integrating novel and known concepts to achieve speed, precision, and taxonomic coverage for taxonomic profiling and strain-resolved analysis.

Here I present protal, a tool for taxonomic profiling and strain-resolved analysis using short-read metagenomic data. Like varkit, protal uses 120 bacterial marker genes from GTDB as a reference to offer broad taxonomic coverage and seamless integration with taxonomic tools such as GTDB-tk [43]. I implemented a custom alignment algorithm around a novel data-structure, the flex-map. This allows for

fast and accurate read alignment with a database densely populated with high similarity sequences, such as conserved marker genes. Further, protal's flex-map stores and reports information on k-mers that are unique to their species cluster, but are absent from other species. This is a unique advantage, as protal is simultaneously an alignment tool, but also stores additional information about taxonomy in its data structure. In combination with machine learning assisted taxonomic profiling, protal has higher sensitivity and precision on almost all datasets. On strain-level, protal reuses alignments for SNP calling and a subsequent reference-guided multiple sequence alignment (MSA), allowing for a 70-fold speed improvement over similar strain-resolved tools.

In the methods section, I introduce protal's workflow, covering its input, output, alignment, taxonomic profiling, and strain-level resolution. The alignment section begins with an introduction to the flex-map data structure, followed by detailed steps on building the index, seeding, anchor-finding, and performing the actual alignment between read and reference. Later, I explain the taxonomic profiling process and how alignments, unique k-mers and random forests are employed to create taxonomic profiles. This is followed by a description of how reference-guided alignment is implemented to reconstruct per-species and across samples MSAs and achieve strain-resolution.

The results section leads with a benchmark of protal against other contemporary aligners on a dataset simulated from all marker genomes (see Section5.2.5), and further information about unique k-mers and the trained random forest. Next, protal's profiling performance is benchmarked against MetaPhlAn 4 (GTDB) [26] and mOTUs3 (GTDB) [232] using benchpro (see Chapter 3). As protal natively profiles within the whole GTDB space, I chose to compare protal against MetaPhlAn 4 and mOTUs3 using only their adjusted species-level benchmarks to enable a fair comparison. On strain-level, I compare protal's MSAs and subsequently generated phylogenetic trees to StrainPhlAn 4, again using benchpro. Lastly, a speed and memory benchmark demonstrates the potential of protal's approach.

5.2 Methods

5.2.1 Workflow Approach

Protal takes a set of short, paired-end reads from shotgun metagenomic sequencing as input and computes taxonomic profiles for each sample by aligning all read-pairs to a reference containing GTDB marker genes for 80,789 bacterial species (Fig. 5.1). For strain-level, protal reconstructs MSAs for each species present in multiple samples. A pre-built database is provided to the user.

Protal follows a common approach of aligning reads against marker genes [26, 232]. Read alignment provides enough information to infer metrics such as ANI to

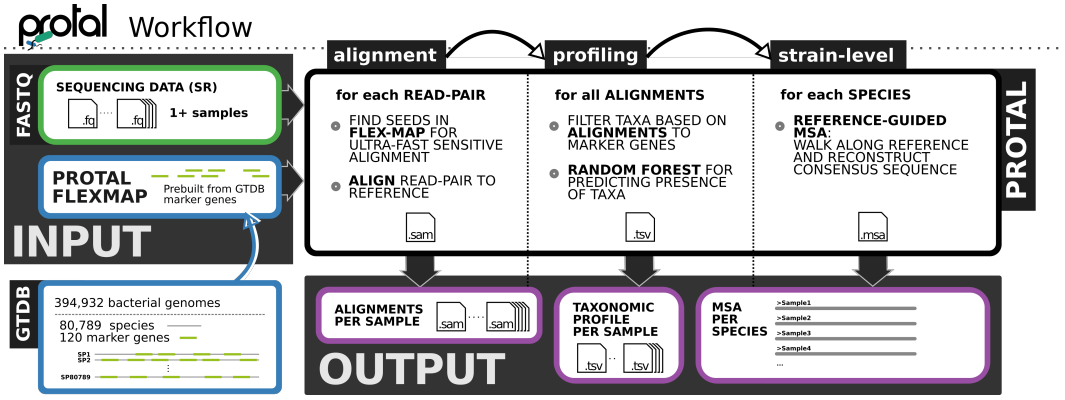

Figure 5.1: Protal takes a set of paired-end short reads from shotgun metagenomic sequencing and outputs per sample taxonomic profiles and strain-resolved MSAs per species present in multiple samples. Internally, protal has three distinct steps - alignment, profiling, and strain-level. In the first step, all reads are aligned against all species-representative marker genomes within GTDB r214.0. During profiling, these alignments are processed and counted for each marker gene of each species. A random forest evaluates evidence for each species to predict presence or absence. To achieve strain-level resolution, alignments from the same species across multiple samples are used to yield a reference-guided alignment.

the reference, MAPQ scores, and alignment length, and further allows inference of SNPs to increase the resolution to strain-level. Marker genes are commonly used to a) focus on highly reliable regions that are single-copy and do not exhibit any HGT to avoid spurious evidence for species-presence and b) to keep the database smaller, which is usually loaded into memory at once.

Protal's design philosophy follows an integrated approach rather than using external tools. This allows for customizing parts of the pipeline so they are tailored towards the use-case of metagenomics. These customisations entail high-similarity read alignment, incorporation of unique k-mers to decrease FP rate, random-forest assisted presence prediction, and reference-guided MSA directly utilizing existing alignments. This allows protal to not only increase sensitivity and precision, but it further decreases the runtime as opposed to relying on external software.

5.2.2 Alignment

Flex-mers and flex-map

Protal's alignment algorithm is based on a seeding approach that uses core-mers and flex-mers to speed up finding candidates for alignment. I define core-mer (here 15-mer) as a substring of a k-mer (here 31-mer) that sits in its middle and is used for exact matching. Further, I define a flex-mer as the flanking region around a core-mer, and it is used for inexact matching (2x8-mer) (see Fig. 5.2). In practice, this means that for each lookup, the core-mer is exactly matched and all positions in the reference are then further filtered by highest similarity in the flanking flex-mers.

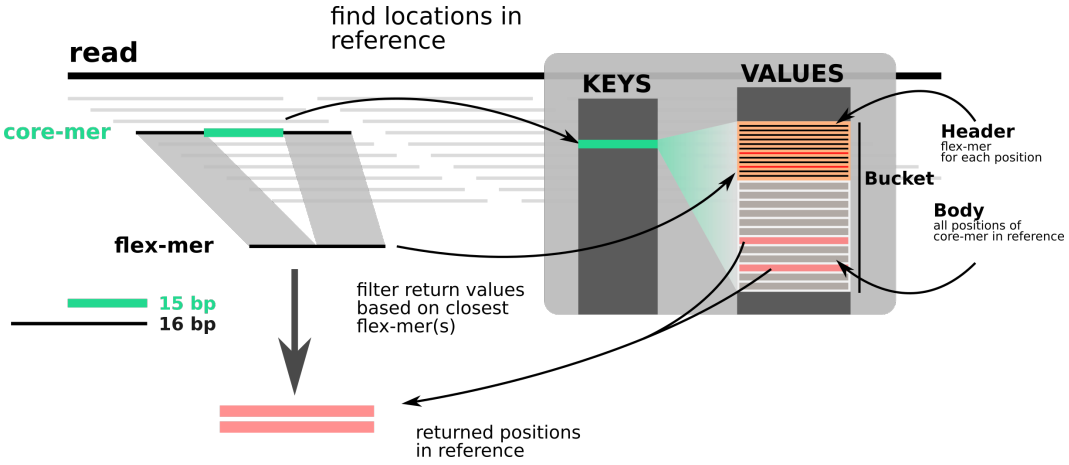

Figure 5.2: Concept of the flex-map. The flex-map serves as data structure for key-value pairs, with keys being core-mers and values being all positions in the reference for a core-mer and additional flex-mer information. For each core-mer stored in KEYS, there is a bucket in VALUES storing the location of all its occurrences in the reference. Those locations are stored in the 'Body' section of the bucket. Additionally, buckets of size > 1 have a 'Header' section storing the flex-mer for each core-mer. The flex-mer is the 2x8 flanking region of each core-mer. Given a k-mer with core-mer and flex-mer from the query, exact matching of the core-mer is used to retrieve the bucket with all its locations in the reference and the flex-mer is used to further filter the location based on flanking region similarity.

For clarification, I use the term core-mer to emphasize the concept, however, here, a core-mer is always of length 15. This length has proven to be sensitive, yet precise in other aligners (e.g. minimap2 [145]) and works well with the flex-map data-structure. Similarly, flex-mers are of length 16 and are the concatenation of the flanking regions around a core-mer within a 31-mer.

Before querying reads, the reference database needs to be indexed. Protal's index is a custom implemented hash table, called flex-map, holding key-value(s) pairs with keys and values stored separately using a 16-bit-array, called KEYS, and a 64-bit array, called VALUES (see Fig. 5.2). Keys, stored in KEYS, are core-mers and stored in 2-bit representation with A, C, G, and T stored as 00, 01, 10, and 11, respectively. Compared to classical hash tables, the hash function for the flex-map is the 2-bit representation of a core-mer and gives us the position for each core-mer in KEYS. For example, the core-mer 'AAAAAAAAAAAAAGT' is represented as the bit string '00000000000000000000000001011', which translates to index 11.

In the following, taxonomic identifier is abbreviated as tid, gene identifier as gid, gene position as gpos, and read position as rpos. Value(s), stored in VALUES, corresponding to a core-mer are 4-tuples (tid, gid, gpos, unique) populating consecutive entries in VALUES. Unique refers to additional information whether a k-mer is unique to its species and is explained later in Section 5.7. Each 4-tuple populates a single 64-bit cell in VALUES and points to an exact location in the reference. Core-

mers that are absent from the reference sequences have an entry in KEYS, but no entries in VALUES. If a core-mer has more than two occurrences in the reference, an additional flanking 8 nucleotides, the flex-mer, is stored for inexact matching to reduce the number of returned locations.

To reduce memory footprint, the KEYS array only has 16-bit cells. Since 16-bits can only index $2^{16} = 65,536$ positions in VALUES, an additional 'global offset' needs to be stored between entries in KEYS. Hence, entries are grouped in blocks of 16. Every 16 entries, an extra 32-bit (2 * 16-bit cells, called control-block) is used to store a new global offset for VALUES; the following 16 entries in KEYS are local offsets in VALUES relative to this block's global offset. Thus, the total size of KEYS is constant and occupies $2^{30} \cdot 2byte + (2^{30}/16) \cdot 4byte = 2.25$ GB of space. $2^{30}/16 \cdot 4byte$ is the space requirement for the control-blocks, given blocks are inserted every 16 keys, and each control-block occupies 4 byte. In contrast to this, using the naive approach and storing offsets in 64-bit cells, the space requirement would be $2^{30} \cdot 8 = 8$ GB. The size of VALUES is variable and depends on the number of stored reference locations. The index is built from marker genes of GTDB species representative genomes (version r214) [206].

Minimizers and Syncmers

In the previous Chapter 4, varkit demonstrated a high memory consumption (Fig. 4.14), making it less accessible for users. While Varkit could not avoid storing all k-mers due to its unique approach for identifying SNP positions, many common methods exist to subsample k-mers in order to reduce memory usage and computational load. One widely used technique is minimizers, which offer a space- and time-efficient way to represent k-mers in DNA sequence analysis. First introduced by Roberts et al. in 2004 to address storage limitations in biological sequence processing, minimizers work by selecting only a subset of k-mers—typically the lexicographically smallest k-mer within a sliding window of size w—thereby reducing redundancy in sequence representation [227]. This strategy enables faster sequence comparison and alignment by avoiding the need to store or process every k-mer.

This type of minimizer is context dependent, as mutations within the sliding window w could alter the lexicographic ordering of the k-mers, and thus changing the selected minimizer. There are various types of minimizers including modulo minimizers, and sync-mers. While standard minimizers select the smallest k-mer in a window, syncmers and their variants are independent of the local sequence context and are therefore less susceptible to sequencing errors. Popular tools that use minimizers include Minimap2 and Kraken2 [145, 307]. Open syncmers, in particular, select k-mers where the smallest s-mer (a substring of length s) occurs at a specific position within the k-mer, leading to better conservation in comparison to context dependent minimizers [61]. Protal uses open-syncmers as a strategy to reduce memory.

Unique k-mers

To increase precision, protal implements an additional post-processing step to add information, whether exact matching 31-mers are unique to their species. For this, all stored k-mers (core-mer + flex-mer) are tested against all 394,932 marker genomes to determine which k-mers are unique to a species and have no occurrences in other species. Short uniques are k-mers with a unique core-mer. Long uniques are 31-mers (core-mer and flex-mer), with no occurrences outside of their species. Super long uniques are similar to long uniques but their closest matching k-mer outside of their species cluster has a minimum Hamming distance of 2. It follows that each long super unique is by definition also a super unique. This information is stored in 3 bits out of the 64 bits per value in VALUES. Unique k-mer information does not influence the alignment process, but is reported together with the best alignment.

Seeding, anchor-finding, and alignment

Briefly summarized, to align reads to reference, protal uses matching k-mers, called seeds, between read and reference to find candidates for alignment. As alignment is computationally expensive, the aim is to only submit a few selected candidates for alignment. The following describes the candidate selection process and alignment in detail.

Seeding is used to find candidate references to align the reads to, in order to reduce the number of computationally expensive alignments. I define a hit as a 4-tuple (tid, gid, gpos, unique) pointing to a single location in the reference database; it is stored in a 64-bit value. A value-block corresponds to a stretch of hits that have identical core-mers. I define a seed as a 5-tuple of (tid, gid, gpos, rpos, unique) that links a core-mer in the query to the location(s) in the reference where the same core-mer can be found. For each read, all 31-mers that are also syncmers are extracted (Fig. 5.3 1). The core-mer of every selected 31-mer is used as a key for the flex-map (Fig. 5.3 2). For each core-mer, the bucket in VALUES is retrieved. The size of a bucket refers to the number of seeds for this core-mer. All buckets are then sorted by size in ascending order (Fig. 5.3 3). Starting with the smallest bucket, seeds are retrieved and stored in a seed list based on their closest matching flex-mer (Fig. 5.3 4). A strategy is implemented to reduce the number of retrieved seeds, and hence the computational complexity for sorting and processing retrieved seeds: protal stops if seeds from at least four buckets have been retrieved and the seed list size exceeds the threshold of max_seed_size (default value is 128). All retrieved seeds are sorted by tid, gid, and rpos (Fig. 5.3 5). Seeds are then grouped into anchors based on tid and gid (Fig. 5.3 6). Seeds in the same anchor must further have the same pairwise distance in both read and reference, hence have no indels in between. In the next step, for each anchor all seeds are extended with respect to the reference, until a mismatch is found or another seed is hit (Fig. 5.3 7). Anchors are

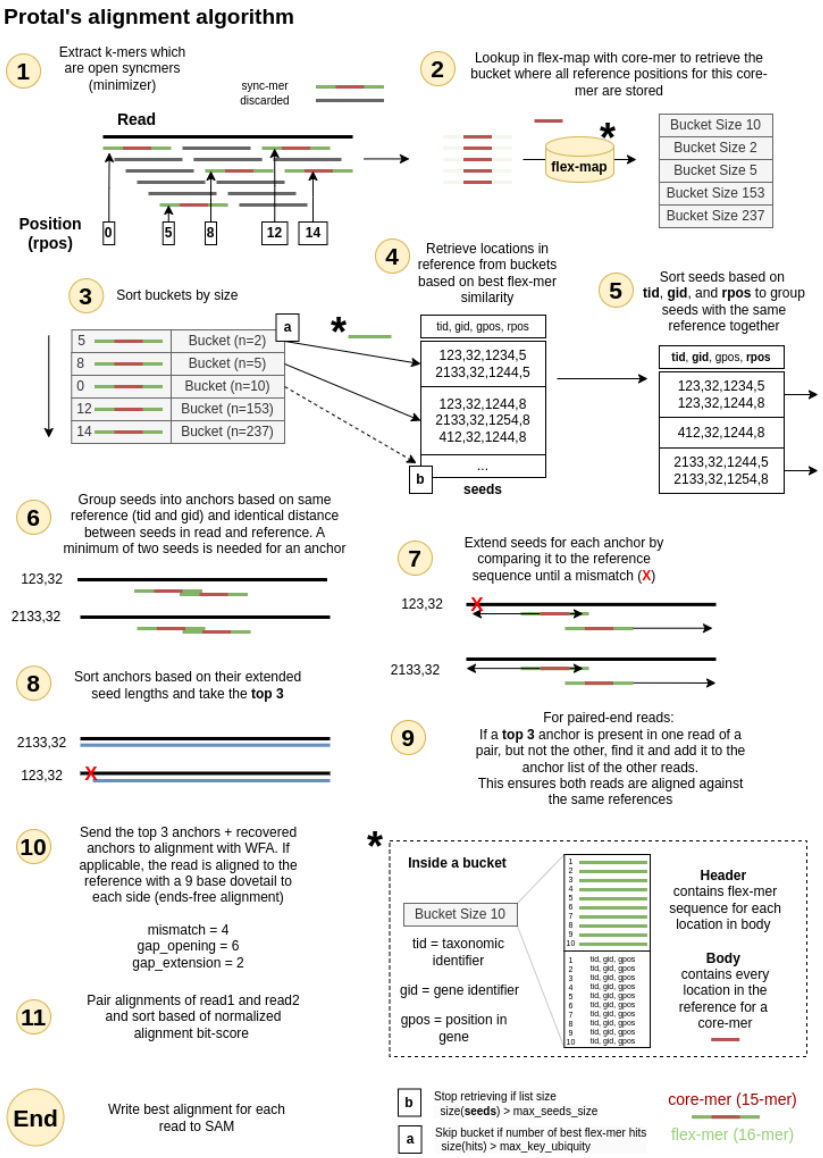

Figure 5.3: Detailed description of protal's alignment workflow. See Section 5.2.2 for more details. Information about unique k-mers is not mentioned here, as they do not influence the alignment process.

then sorted in descending order by their number of exact matching positions to the reference (Fig. 5.3 8). If the reads are paired end and not all top 3 anchors are shared between both reads (with respect to tid and gid), additional anchors are recovered (Fig. 5.3 9). If there is no matching anchor, further seeds are retrieved from the buckets to match the target anchor. Finally, the top 3 anchors and recovered anchors for each read-pair are submitted to alignment with WFA2 against the respective reference sequence [167] (Fig. 5.3 10). The penalty scores used for alignment are mismatch=4, gap_opening=6, and gap_extend=2. Subsequently, alignments are paired and sorted based on the normalized alignment bit-score (Fig. 5.3 11). With the presented scoring system, the perfect alignment has a score of 0. When the score

is divided by 4 (mismatch penalty), I get an ANI approximation. The best scoring alignment pair is reported.

During seeding, anchor grouping, and alignment, information about unique k-mers is passed through and retained to be stored with the alignment. For each reported alignment, the number of long unique and long super unique k-mers are stored in the optional columns in the sam file under ZU:i:<value> and ZT:i:<value> for long uniques and long super uniques, respectively. Further, for each alignment protal computes a MAPQ score. MAPQ scores are the phred-scaled probability (-10 * log10(p)) that a read(-pair) is incorrectly mapped. If a read maps to multiple locations in the reference at a similar identity, the MAPQ score is lower; if it maps to only one location at high identity it is higher. MAPQ is computed according to the following formula, which follows the implementation of calculating MAPQ in minimap2 but I omit the chaining factor [145]. S_1 and S_2 hereby represent the score of the best and the second best alignment, respectively.

$$MAPQ = 40 \cdot (1 - \frac{S_2}{S_1}) \cdot log_{10}(S_1) \tag{5.1}$$

5.2.3 Taxonomic Profiling using Random Forests

Protal utilizes random forests [31] to make sense of a wide array of per-species metrics to predict its presence or absence. Random forests are a machine learning algorithm composed of multiple individual models to make predictions. The training data was obtained from protal's output on all benchmark datasets CAMI Human-Toy, Marine, and Mouse. Further, the CAMI dataset Rhizosphere was used to train the random forest, but was later removed from the species-profiling benchmark as there were inconsistencies across all tools. Further, the dataset MSSS200R was used for training, and is introduced in Section 5.2.5). Protal outputs a dataset of metrics for all species that have at least one read assigned (see Table 3.1). Using the gold-standard profiles, the dataset was annotated with information which taxa are present and served as an input for building a random forest with the 'randomForest' R package [152] version 4.7-1.1. I used an 80/20 training/test split in the dataset and trained the forest with 'randomForest' using the parameters 'ntree = 256, maxnodes = 128'. Protal calculates the same metrics from alignments when profiling reads, and uses the random forest to predict species presence.

Expected gene presence is an input for the random forest (see Table 5.1). Given a certain number of reads, I expect to have a certain number of genes discovered. This is modeled with the following equation. $genes(t)$ is the number of total genes

Table 5.1: The following metrics are computed by protal for each species from all read alignments and are used as input for the random forest.

Metric Name	Metric Description
lu_gene_rate3	Observed number of genes with at least 3 long uniques divided by the number of genes with at least 3 long uniques in database
lu_gene_rate2	Observed number of genes with at least 2 long uniques divided by the number of genes with at least 2 long uniques in database
lu_gene_rate	Observed number of genes with any long uniques divided by the number of genes with any long uniques in reference
lu_genes	Observed number of genes with at least one long unique reported
present_genes	Number of genes hit by at least one read
lu	Total number of observed long unique k-mers (reported in optional part of alignment sam-file)
lsu_gene_rate	Observed number of genes with any long super uniques divided by the number of genes with any long super uniques in reference
lsu_genes	Observed number of genes with at least one long super unique reported
lsu	Total number of observed long super unique k-mers (reported in optional part of alignment sam-file)
lsu_gene_rate2	Observed number of genes with at least 2 long super uniques divided by the number of genes with at least 2 long super uniques in reference
expected_gene_presence_ratio	Observed gene presence rate relative to expected gene presence
expected_gene_presence	Formula for how many genes are expected to be hit based on the number of reads (see Equ. 5.2)
lu_per_read	Number of long uniques divided y number of reads
lsu_gene_rate3	Observed number of genes with at least 3 long super uniques divided by the number of genes with at least 3 long super uniques in reference
mean_ani	Mean ANI across all reads and genes (computed from Alignment CIGAR [147])
variance1	Variance calculated on predicted vertical abundances for all genes
stddev	StdDev calculated on predicted vertical abundances for all genes (only for present genes)
AF0	Number of alleles that did not pass quality filter (sum of qualities >60 and minimum 2 observations)
lsu_per_read	Number of long super uniques divided y number of reads
total_genome	Total number of k-mers for reference in index
RAF0	AF0 divided by all Variant positions
hittable	Genes that can be hit (Need to have at least 1 unique k-mer in protal database)
su_rate_ref	Fraction of short uniques in reference with respect to all k-mers
uniqueness	Number of reads with MAPQ >20 divided by number of reads
lu_genome	Number of long uniques in reference genome
lu_rate_ref	Fraction of long uniques in reference with respect to all k-mers
lsu_rate_ref	Fraction of long super uniques in reference with respect to all k-mers
lsu_genome	Number of long super uniques in reference genome
su_genome	Number of short uniques in reference genome
mean_mapq	Mean MAPQ across all reads and genes (computed from Alignment CIGAR [147])
variance2	Variance calculated on predicted vertical abundances for all genes (only for present genes)

in the reference for taxon t and $reads(t)$ is the number of reads mapped to t.

$$\textbf{ExpectedGenePresence(t)} = (1 - (\frac{genes(t) - 1}{genes(t)})^{reads(t)}) \cdot genes(t) \quad (5.2)$$

$$\textbf{ExpectedGenePresenceRatio(t)} = \frac{PresentGenes(t)}{ExpectedGenePresence(t)} \quad (5.3)$$

For example, if a taxon has 120 marker genes with 350 reads mapped in total, I would expect reads to hit $(1 - \frac{120-1}{120}^{350}) \cdot 120 = 110$ genes. If I then find only 50 genes are hit, the expected gene presence ratio is $\frac{50}{110} = 0.45$.

Abundance calculation For all present species, the abundance is computed as the median of per gene vertical coverage. The vertical coverage for each gene is calculated according to the Lander-Waterman equation from all its mapped reads and is the sum of length over all mapped reads divided by the length of the gene [134].

5.2.4 Strain-level

In a dataset with multiple metagenomes, strain-level is achieved by comparing alignments across metagenomes for all species that are present in at least two metagenomes. A common approach is to call SNPs based off the alignments and then reconstruct a per species consensus sequence for each metagenome. A multiple sequence alignment of all consensus sequences can then be used to create a maximum-likelihood phylogenetic tree. This approach is computationally expensive as MSA construction typically has an algorithmic complexity of $\mathcal{O}(n \cdot \log n)$ [118]. Further, computing a consensus and passing it on to an external tool comes with an additional I/O (Input/Output) overhead. Protal massively speeds this up by calling SNPs internally and reconstructing the consensus simultaneously along with the MSA. This is called reference-guided MSA and is possible, as reads aligning to the same species across metagenomes share the same reference and computing an MSA *de novo* on the consensus sequences is not necessary. ViralMSA [181] and VIRULIGN [153] implement reference-guided MSAs to cope with the large number of viral sequences in molecular epidemiology, but use genomes instead of reads as input. As opposed to traditional MSAs, the computational complexity for reference-guided MSAs scales linearly ($\mathcal{O}(n)$) with the number of input sequences (samples). The following is a detailed description of how protal computes reference-guided MSAs.

Reference-guided MSA

During an initial step, all alignments are grouped per species and marker gene. All alignments with a MAPQ lower than 4 are discarded. Individual marker genes need to be present in at least three samples to be included in the MSA. From alignments

against the reference marker gene, protal extracts all (unfiltered) variants in a vector and further keeps a vector containing vertical base coverage for every position. Every position in the MSA needs to have a minimum coverage of 2 to be included in the MSA. For multi-allelic positions, the consensus call is the variant with the higher sum of quality-scores. For example, with variants denoted as $(base, quality)$, if a position has the bases $('A', 20), ('A', 35), ('A', 40), ('T', 25), ('T', 40)$, the consensus is 'A' with a quality sum of 95 as opposed to 'T' with a quality sum of 65. As protal only tracks variant positions with quality, reference bases have a default quality of 30. Finally, the consensus variant must have a minimum coverage of three, and a minimum sum of quality scores of 60 to pass quality control. In our example, variant 'A' passes with a coverage $3 \geq 3$ and a sum of quality scores $95 \geq 60$. When building the MSA, protal traverses the marker gene reference and for each position traverses all samples twice. In the first traversal, protal determines whether any variant is an insertion passing quality control, and if so, keeps track of the maximum insertion in any of the samples for this column. In the second traversal, protal considers the following rules for each sample to decide which base to incorporate at this position. First, the sample needs to have at least a vertical coverage of two, otherwise a '-' (gap) is incorporated. If the vertical coverage is ≥ 2, and the position has a variant that passes quality control, the variant is incorporated. If the passing variant is an insertion, the insertion is added. If the passing variant is a deletion, '-' for the next reference positions according to the length of the deletion is incorporated. If there is a variant that did not pass quality control (that includes the reference base), an 'N' is incorporated. If there is no variant and the vertical coverage is ≥ 2, the reference base is incorporated. An additional number of '-' is added according to the maximum insertion size for this column. At the end of building the species MSA, protal filters all samples based on a minimum 5,000 bases vertical coverage. At the moment all thresholds are hard-coded and cannot be changed by the user.

5.2.5 Additional Datasets and Benchmarks

Firstly, to evaluate the internal read alignment, I simulated reads from all marker genomes (n=394,932), including non representative ones, using art_illumina 2.5.8 [102] and the parameters *'-nf 0 -p -i reference.fna -o output -l 150 -ss HS25 -f 5.0 -m 200 -s 10'* resulting in 2x944,933,468 reads. Each species has up to 120 marker genes, but not all marker genes are present in all species. As some marker genes are shorter than 200 nucleotides, I added 128 N's at the beginning and the end of each marker gene for reads to be able to span the sequence. I aligned all resulting reads using bowtie2 2.5.3 [136], bwa-mem2 2.2.1 [146], strobealign 0.13.0 [234], and minimap2 2.28-r1209 [145]. The parameters for the aligners were *' –very-sensitive -p 16'* for bowtie2, *'-t 16'* for bwa-mem2 (bwa-mem2.avx to enforce usage of the avx extension), *'-ax sr -t 16 –secondary=no'* for minimap2 to change the settings to sam output,

short-reads, and to only output the primary alignment, *'-t 16 -U'* for strobealign to suppress output of unaligned sequence, and *'–max_ score_ ani 0.9 -u 256 -s 128 -t 16'* for protal to stop alignments with lower prospective ANI than 0.9. The databases for all tools were built with default settings from species representative marker genomes for bacteria in GTDB r214 (80,789). Each non-representative marker genome belongs to a species cluster. I assess all alignments based on whether the reads were aligned to the representative marker genome of their respective species cluster. I count TP, FP, and FN alignments to compute TP-rate and FP-rate for all MAPQ thresholds. I did not compute TN or FN values as this would require to know whether it is possible to align the read to the true reference. Speed and memory benchmarks for the alignment tools were conducted with an AMD EPYC 7713P CPU and 512GB RAM with no other jobs running on this node during the benchmarks. I used a subset of the simulated marker gene data, which resulted in an uncompressed set of paired-end reads with 7.9GB each. After each run, I further used pigz 2.5 with *'-p 16'* to compress the resulting sam file. Each tool was run twice to ensure all tools had their databases pre-cached by the second run.

I further introduce a new dataset MSSS200R (**m**ultiple **s**pecies **s**ingle **s**train), which was added to assess the profiling performance of rare species. From the UHGG database [4], a collection of common genomes in the human gut, 783 isolate genomes were selected from 363 species that have only one genome representative in GTDB r214. Similar to the strain-level dataset described in Section 3.2.5, this dataset is more a technical benchmark and not an attempt to create a truthful representation of a common microbiota. All genomes utilized are listed in Table A.6.

I simulated 20 samples with a single genome across 200 species. Both genomes and species were randomly selected from the pool of available genomes. The vertical coverage across all genomes was fitted to match a binomial distribution, with a minimum vertical coverage of 0.05, and a maximum vertical coverage of 70. Per sample mean vertical coverages and number of species are listed in Table A.5. I used art_illumina to simulate paired-end reads of length 2x150bp with the parameters *'-p -l 150 -ss HS25 -m 200 -s 10'*. This dataset is only used for species-level profiling in Section 5.3.3. Strain-level benchmarking was conducted using the dataset introduced in Chapter 3.2.5. Further, species and strain-level benchmarks were conducted as explained in Chapter 3.2.

5.3 Results

5.3.1 Overview

In this section I will provide a brief overview over the presented results (Fig. 5.3). As protal employs its own algorithm for aligning reads to the marker-gene reference, the first benchmark is an evaluation of the alignment performance compared to other

aligners with respect to correctness, speed and memory (Fig. 5.3 A). This part further presents results regarding unique k-mers, a way to improve accuracy in the downstream taxonomic classification (Fig. 5.3 B). Part of this is also the evaluation of the random forest as an automated process to classify species as present or absent, based on data such as unique k-mers found for each species (Fig. 5.3 C).

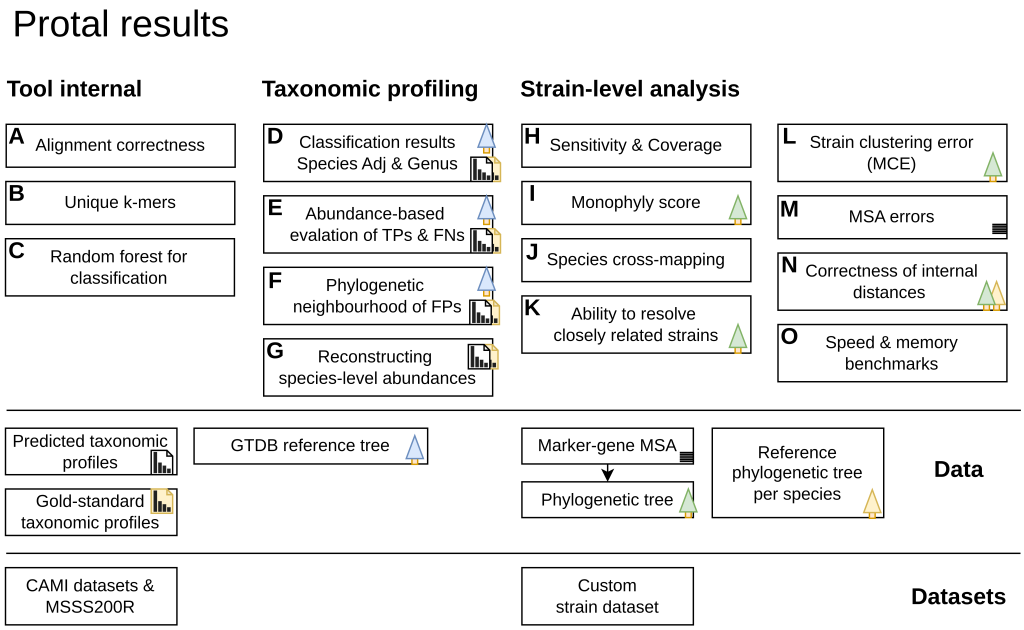

Figure 5.4: Overview over protal's results section and the presented analyses. The results section broadly divides into tool internal benchmarks, such as evaluating the correctness of alignments, taxonomic profiling benchmarks, and strain-level analysis benchmarks.

The subsequent analyses are structured similarly to the benchpro results chapter and are divided into sections on taxonomic profiling and strain-resolved analysis. For the taxonomic profiling benchmarks, only GTDB-based profiles are considered (r207 for MetaPhlAn 4 and mOTUs3; r214 for protal), excluding the NCBI taxonomy-based CAMI profiles, which were already addressed in Chapter 3.3.2. It is important to note that protal does not currently support output in the NCBI taxonomy, as converting between NCBI and GTDB taxonomies is non-trivial. Additionally, a novel dataset—MSSS200R—is introduced, composed exclusively of simulated reads from species represented by a single genome in GTDB r214. This dataset tests database coverage and each tool's ability to detect novel taxa. Kraken2+Bracken is excluded from this benchmark section, having already been evaluated in Chapter 3.3 where it demonstrated suboptimal performance.

The evaluation begins with taxonomic classification performance of protal, MetaPhlAn 4, and mOTUs3 across all CAMI datasets and MSSS200R, at both the species level (adjusted; see Section 3.2.2) and genus level, to assess protal's ability to accurately predict taxon presence and absence (Fig. 5.3 D). Following this,

FN and TP are visualized with respect to their relative abundance (Fig. 5.3 E). Next, we assess whether protal's performance is influenced by species richness. The analysis then explores protal's phylogenetic neighbourhood of FPs with respect to FN-s, FN+s, and TPs (Fig. 5.3 F). This is followed by benchmarking the ability to reconstruct species-level abundance values across all datasets (Fig. 5.3 G).

The strain-level analysis begins by evaluating protal's sensitivity at both the sample and strain levels, with particular attention to the vertical coverage of strains (Fig. 5.3 H). To identify potential species-specific challenges, these results are further stratified by species. Given protal's higher sensitivity compared to StrainPhlAn 4, results are presented for both the full protal trees (including all samples) and for trees filtered to only those tips shared with StrainPhlAn 4. This facilitates a side-by-side comparison of the tools' performance.

The next analysis focuses on species-specific monophyly scores for protal and StrainPhlAn 4 to assess their ability to accurately resolve strain-level identities (Fig. 5.3 I). This is followed by an in-depth case study of two species—one exhibiting well-resolved monophyly, the other poorly resolved—highlighting issues such as cross-mapping of reads between closely related species (Fig. 5.3 J).

Subsequently, monophyly is analysed in relation to the phylogenetic distances between strains in the dataset, illustrating each tool's ability to differentiate closely related strains in the resulting trees (Fig. 5.3 K). To quantify errors in cases of non-monophyly, the MCE metric is applied (Fig. 5.3 L). The relationship between alignment quality and phylogenetic accuracy is then explored by linking MSA errors to alignment lengths and monophyly scores (Fig. 5.3 M).

Following this, the inferred phylogenetic trees are compared to reference trees generated using roary on the source genomes. Several tree distance metrics are employed to evaluate how well protal and StrainPhlAn 4 recover the internal topology of the true strain relationships (Fig. 5.3 N). Finally, the runtime and memory consumption of protal are benchmarked against StrainPhlAn 4 and other tools at the strain resolution level (Fig. 5.3 O).

5.3.2 Alignment evaluation

At the core of protal is its custom sequence aligner using the flex-map data-structure for fast and sensitive read alignment in a metagenomic setting. To validate the benefits of the flex-map data-structure, I tested protal's internal aligner against four state-of-the-art alignment tools bowtie2 [136], bwa-mem2 [169], minimap2 [145], and strobealign [234]. As protal's aligner is optimized for an application in metagenomic profiling, I created a benchmark that targets this specific use-case. Like protal, the databases for the other aligners are built from all species representative marker genomes in GTDB r214. I simulated reads from all marker genomes, not just species-

representatives, and tested whether they align to the representative marker genome of their respective species cluster (for more details see Section 5.2.5). TP-rate and 1-FP-rate are compared with respect to increasingly restrictive MAPQ thresholds (Fig. 5.5 A). The MAPQ value is commonly used for quality filtering and represents a confidence score of the alignment (see Section 5.2.2). Protal shows a good balance between TPs and FPs at MAPQ == 4 with a TP-rate of 0.680 and a 1-FP-rate of 0.974. At this 1-FP-rate, it is only surpassed by bwa-mem2 with a higher TP-rate of 0.691 at a MAPQ of 48. As different tools implement different methods to calculate the MAPQ score, identical MAPQ scores yield different degrees of filtering. While bowtie2 yields lower FP-rate values for MAPQ between 12-37, the low TP-rate of < 0.6 does not justify the lower FP count.

In terms of speed, protal is the second fastest aligner on this dataset with a mean runtime of 5.88min±0.51, just after strobealign with 5.18min±0.95 (Fig. 5.5 B). Both aligners are at least twice as fast as the other aligners (14.07min±1.62 for bwa-mem2, 19.51min±1.46 for bowtie2, and 22.39min±0.88 for minimap2). With ∼35GB, protal is also in the middle regarding memory consumption, using 7GB less than the fastest, strobealign, and 26GB less than the third fastest, bwa-mem2. Bowtie2 and minimap2 have the lowest memory requirement with ∼15GB and ∼20GB, respectively. An analysis of the runtime of individual parts in protal shows that the alignment step takes longest by far (Fig. 5.5 C). This shows that protal's flex-map for seeding is not the bottleneck of the tool. Protal's flex-map is also responsible for keeping the number of seeds and anchors small by first seeding for 15-mers, but only reporting based on best hits to 31-mers (Fig. 5.5). This reduces the downstream runtime while maintaining a high sensitivity (Fig. 5.5 D).

5.3.3 Unique-k-mers and Random forest

Protal utilizes a random forest for predicting the presence or absence of taxa. When training the random forest on a random 80% split of the dataset, the sensitivity was 0.9886 and the precision was 0.9898. On the 20% test dataset, the sensitivity was 0.9857 and the precision was 0.9880. Evaluating the importance of variables in the random forest shows that information about unique k-mers is most important for determining presence or absence of taxa (Fig. 5.6, see Section 5.2.3 for details). Unique k-mers are split into three types: short unique k-mers, long unique k-mers and long super unique k-mers. Long unique k-mers are exact matching 31-mers in the database, that only occur in genomes within a species, but not outside (see Section 5.2.2). Variable lu_gene_rate3 quantifies the number of genes present with at least three long unique k-mers with respect to genes with at least three long unique k-mers in the database. Similarly, lu_gene_rate2 and lu_gene_rate measure with a two and one uniques per gene thresholds. The most important variable not describing unique k-mers is present genes - the number of genes for

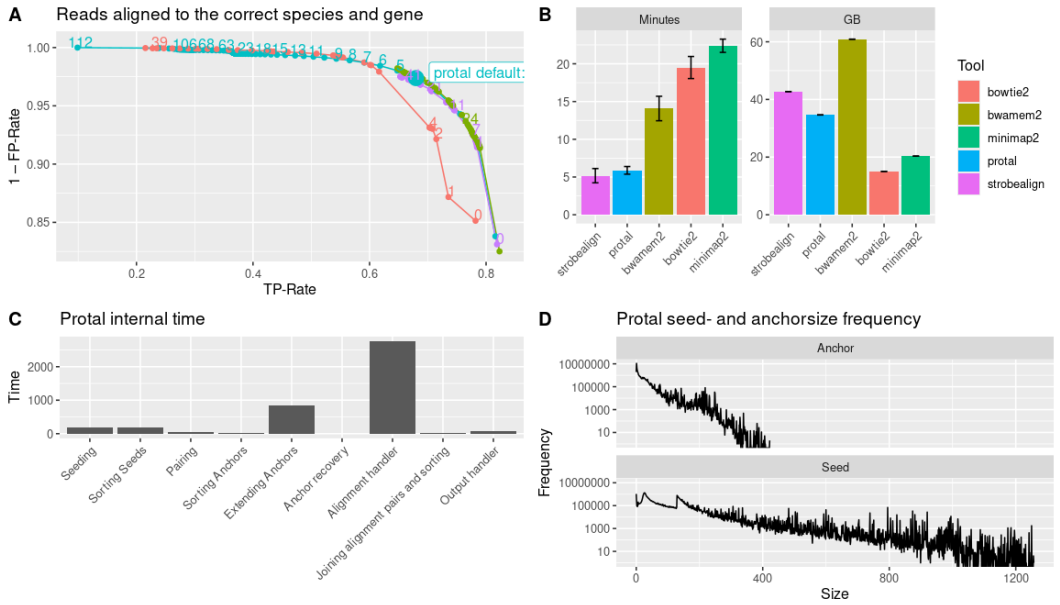

Figure 5.5: A, benchmark on 1,889,866,936 reads simulated from all bacterial marker genomes in GTDB r214 (n=80,789), comparing different aligners. TP and FP are evaluated based on whether a read was mapped to its correct species cluster representative. All reads that are neither TPs nor FPs are FNs. TP-rate and FP-rate are calculated relative to the total number of reads at all MAPQ filtering thresholds. B, runtime in minutes and memory in GB for all aligners on a dataset of 2x7.9GB uncompressed paired-end reads. Output is uncompressed for all tools. C, protal's internal time benchmarks on the dataset in B. Alignment takes by far the longest out of all stages of the alignment process. Stages refer to the following sections in Fig. 5.3: Seeding (1,2,3,4), Sorting Seeds (5), Pairing (6), Sorting Anchors (8), Anchor recovery (9), Alignment handler (10), Joining alignment pairs and sorting (11), Output handler (End). D, seed- and anchor-size frequencies in protal. Smaller seed sizes and anchor sizes with higher frequency result in a better runtime.

a species with at least one read alignment. Surprisingly, uniqueness has a very low impact on the classification result of the random forest. Uniqueness measures the percent of reads mapped to a species with a MAPQ >20 and should in theory be indicative of unambiguous high-quality read alignments. In the context of the distribution of unique k-mers within protal (see Fig. 5.7 A), it is observed that many species possess a sufficient number of unique k-mers to enable identification by the random forest. Roughly 95% (76749 out of 80789) of all species have more than 6,250 unique k-mers. However, some genera, for example *g__Collinsella*, harbor many species with a low number of unique k-mers (mean number of uniques in *g__Collinsella* 1812±4419, see Fig. 5.7 B). Within *g__Collinsella*, 399 species have less than 1000 unique k-mers, indicating a high marker genome similarity.

Note, that unique refers to 'not outside of the species', but it does not guarantee that other members other than the representative genome also carry those k-mers.

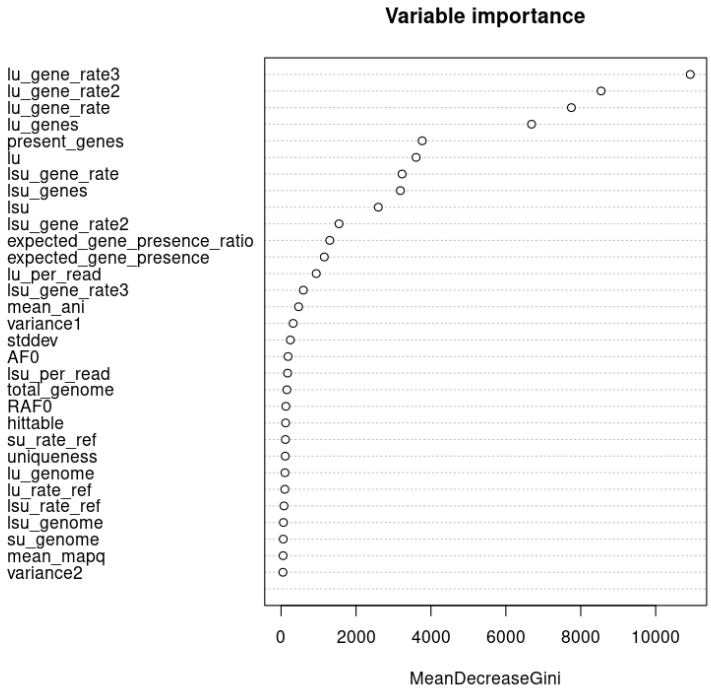

Figure 5.6: Variable importance reported from constructing the random forest. MeanDecreaseGini quantifies the importance of a variable within the random forest. Variable explanations can be found in Table 5.1.

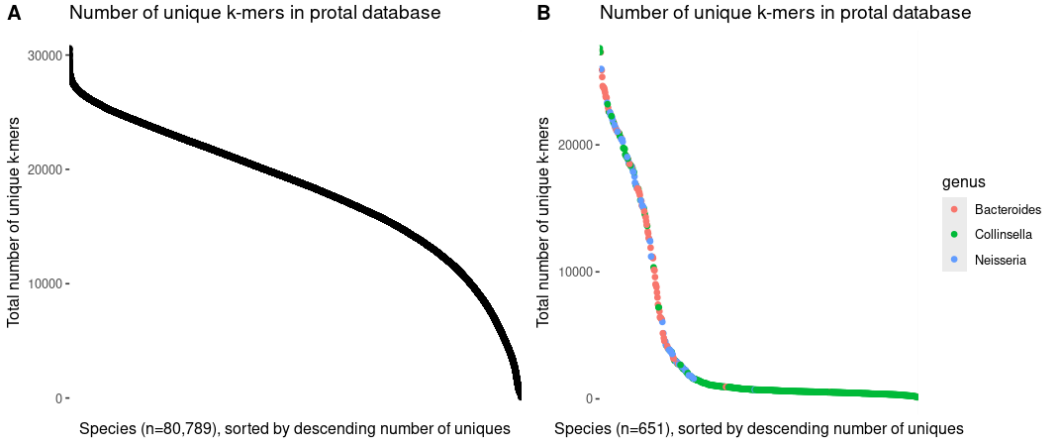

Figure 5.7: The number of unique k-mers (short uniques and long uniques) within the protal database stratified by species. A, unique k-mers across all species. B, unique k-mers only for species of the genera *'g__Bacteroides'*, *'g__Collinsella'*, *'g__Neisseria'*. The distribution of unique k-mers is not uniform across taxa, and the majority of *Collinsella* species have only few (mean±sd = 1812±4419) unique k-mers. Out of 502 *Collinsella* species in GTDB r214, 6 have less than 200 unique k-mers in protal and 86 have less than 200. This indicates a high similarity to other species or high diversity within the species.

5.3.4 Species-level profiling

To benchmark protal's performance for taxonomic profiling, I used benchpro and the CAMI datasets introduced in Chapter 3. With the dataset MSSS200R, I added a benchmark to show the advantages of covering the whole native GTDB space by focusing on GTDB species comprised of only one genome (see Section 5.2.5). As with varkit, adjustment of protal's benchmark scores is not possible as protal natively covers all GTDB taxa. The following only shows benchmarks of tools with respect to the GTDB taxonomy, as the NCBI results have already been discussed in Chapter 3.3. The species-level comparison focuses on adjusted benchmark scores to account for differences in taxonomic databases between tools (see 3.2.2). I excluded Kraken2+Bracken from this benchmark for its inferior performance in previous benchmarks and as results were already discussed in depth in Chapter 3.3.2.

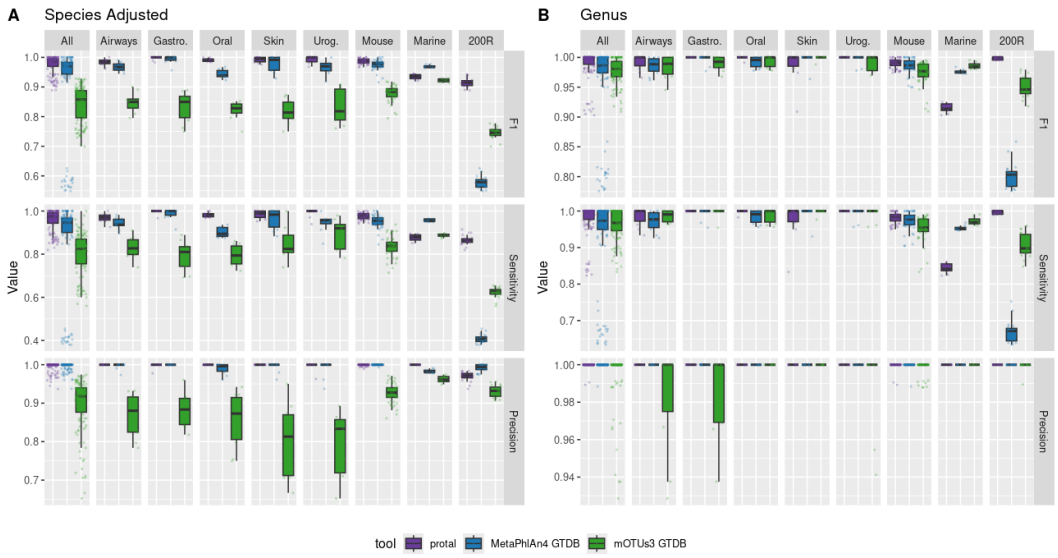

Figure 5.8: Profiling performance across samples, environments, and tools. Performance is measured with the metrics F1-score, precision, and sensitivity (row panels). The boxplots represent different tools and each data point is a sample. The column panels stratify datasets. '200R' is the dataset MSSS200R. A, benchmarks on species-level adjusted. B, benchmarks on genus-level.

Summarized, protal shows superior profiling performance across all datasets, with a total mean F1 of 0.974±0.031 (Fig. 5.8 A). With a focus on precision over sensitivity, protal has a higher mean precision of 0.994±0.013 over a mean sensitivity of 0.955±0.05. For F1 and sensitivity, this puts protal ahead of MetaPhlAn 4, which has a mean F1 of 0.91±0.146 and a mean sensitivity of 0.865±0.201. For precision, MetaPhlAn 4 and protal are almost tied, with MetaPhlAn 4 having a mean precision of 0.995±0.011.

For individual datasets, MetaPhlAn 4 surpasses protal in precision in MSSS200R and Gastrointestinal with a mean precision of 0.991±0.01 over 0.969±0.012 for

MSSS200R, and 1±0 over 0.998±0.007 for Gastrointestinal for MetaPhlAn 4 and protal, respectively. Protal's sensitivity is outperformed by mOTUs3 and MetaPhlAn 4 only for the Marine dataset with mean sensitivity of 0.956±0.006 for protal, 0.956±0.006 for MetaPhlAn 4, and 0.887±0.007 for mOTUs3. This is because protal currently only profiles bacteria, and the Marine datasets contain 442 archaeal species making up for ∼6.9% of all species. For all other datasets, protal has the higher sensitivity.

It is notable that protal has a perfect mean precision of 1 for the four datasets Airways, Oral, Skin, and Marine, and MetaPhlAn 4 has a perfect precision for Gastrointestinal. The biggest disparity in performance is seen for MSSS200R, where protal has a high mean sensitivity of 0.866±0.023 and both mOTUs3 and MetaPhlAn 4 have low mean sensitivity of 0.622±0.025 and 0.408±0.02, respectively. This is due to a lack of species coverage in the respective databases as for MSSS200R, MetaPhlAn 4 and mOTUs have 2,296 FNs from 226 species and 1,340 FNs from 141 species respectively that are not covered by their respective databases. The genus-level results mostly align with the species-level findings in terms of order of tool performance (Fig. 5.8 B). Both protal and MetaPhlAn 4 have perfect precision across all datasets except for Mouse. For MSSS200R, protal increases the sensitivity from species to genus-level from 0.866±0.023 at s to 0.997±0.004 while mOTUs3 has a mean sensitivity of 0.905±0.032, and MetaPhlAn 4 has a mean sensitivity of 0.67±0.03. While the dataset MSSS200R does not simulate a specific environment, it still showcases that there are common species in the human gut that MetaPhlAn 4 and mOTUs3 have no taxonomic coverage for.

Protal is sensitive for low-abundant taxa

Next, I assess the detection thresholds for TPs and FNs for all tools (Fig. 5.9 A). In my benchmarks, protal is the most sensitive profiler and in some cases can detect TP species down to 0.0002% relative abundance. The lowest TP abundance for mOTUs3 is 0.0006% and 0.0007% for MetaPhlAn 4. Although protal is the most sensitive, it also misses some high abundance taxa (FNs) that other tools detect. For each tool, considering only taxa contained in their database, protal misses 25 species with a relative abundance of >1% while MetaPhlAn 4 and mOTUs3 only miss 3 and 8 species, respectively. With one exception, these species are all from the genus *g__Collinsella*. Within GTDB, *g__Collinsella* has multiple closely related species that are either absent from MetaPhLAn 4 and mOTUs3, or are represented by fewer species in their respective databases. MetaPhlAn 4 has 16 species for the genus *g__Collinsella* while GTDB r214 has 502 distinct species for this genus. Is is important to note that all genomes in the dataset were taxonomically placed into GTDB r207 and GTDB r214 using GTDB-tk. The species-level classification is based on whole-genome ANI and AF. For MSSS200R_15, for example, MetaPhlAn 4 misses

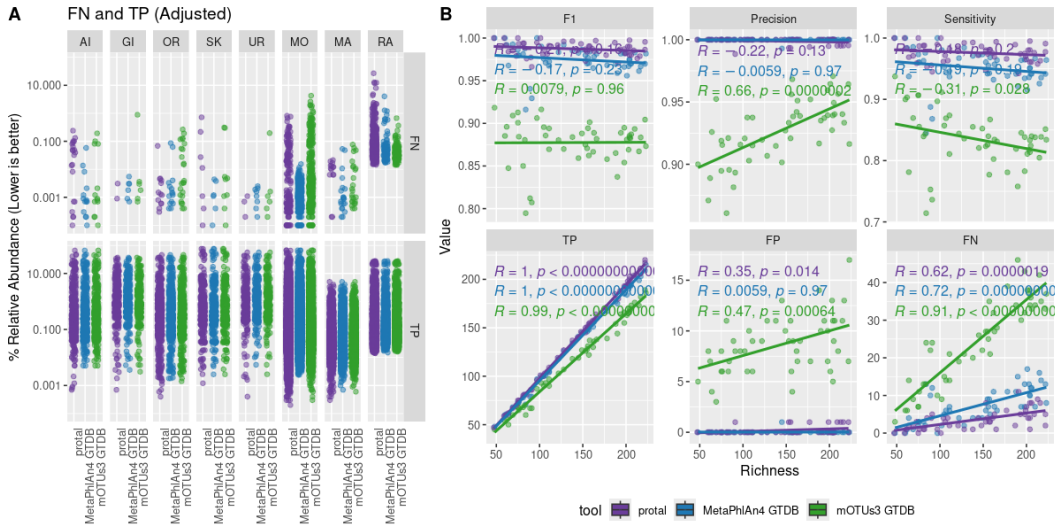

Figure 5.9: A, relative abundance for TP and FN+ (Taxa covered by the respective tool databases) for different tools coloured by tool. Horizontal panels stratify datasets (UR=Urogenital, GI=Gastrointestinal, MA=Marine, MO=Mouse, OR=Oral, SK=Skin, AI=Airways, RA=MSSS200R). Points on the lower end show the sensitivity of the tool to detect low abundant taxa. B, Richness with respect to F1-score, precision, and sensitivity. Each dot represents one sample in the Mouse dataset and all statistics are after adjustment on species level. Richness on the x-axis is calculated as TP+FN and the y-axis shows the value for the respective statistic in each panel.

69 FN *g__Collinsella* species not in its database, with total relative abundance of 20.5%. However, MetaPhlAn 4 predicts one FP *g__Collinsella* species with a predicted abundance of 13.5%, indicating that this single FP species *s__Collinsella sp003458415* covers for many of the closely related missing FN species. This is even more apparent when comparing pairwise distances between *g__Collinsella* species for the GTDB 207 species tree containing only MetaPhlAn 4 species and the GTDB r214 tree (Suppl Fig. A.10). The distribution of pairwise distances in GTDB r214 has a peak between 0.0 and 0.05 while the majority of pairwise distances for MetaPhLAn 4 is between 0.25 and 0.3 . As a consequence, protal fails to detect some high abundant taxa in parts of the tree where species are closely related, while MetaPhlAn 4 fails at representing the species diversity in those regions.

To investigate how the tool performance changes with respect to sample richness, I plotted F1, precision, sensitivity, TP, FP, and FN with respect to all samples within the Mouse dataset (Fig. 5.9 B). F1 score and sensitivity decline for all tools with increasing richness while precision increases for MetaPhlAn 4 and mOTUs3. Albeit not significant, protal is the only tool with a negative correlation between precision and richness, which is caused by protal having no FPs for the low richness samples. Protal also had the lowest increase in FNs (R=0.62, p=10^{-6}) with increasing richness when compared to MetaPhlAn 4 (R=0.72, p=10^{-8}) and mOTUs3 (R=0.91,

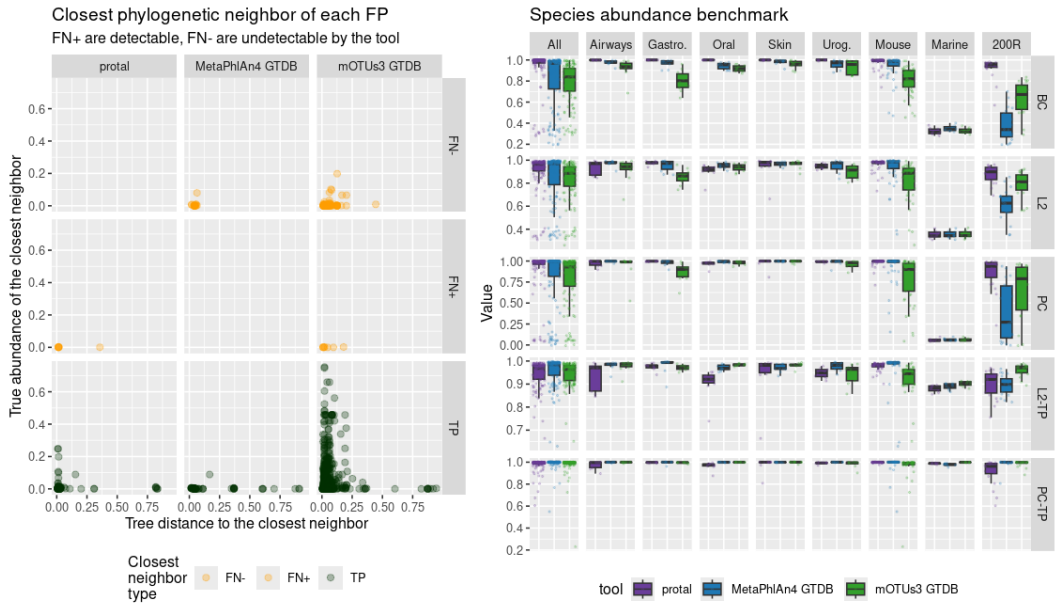

Figure 5.10: A, false positive species in context of their closest TP or FN neighbor in the phylogenetic tree. FNs are split into detectable (FN+) and undetectable (FN-) based on whether the tool has this taxon in its database. B, Abundance prediction benchmark on species-level for all datasets. Metrics are Bray-Curtis Similarity (BC), 1-L2 error (L2), Pearson Correlation (PC), 1-L2 error only on TP taxa (L2-TP), and Pearson Correlation only on TP taxa (PC-TP).

p=10^{-10}), showing that protal remains sensitive for high richness samples.

In the following analysis, I explored the within-sample phylogenetic neighborhood of FPs (Fig. 5.10). Across all datasets, protal has 122 FP species, of which >94% are closest to a TP species with a mean abundance of 0.013±0.04. This means that protal's main source of FPs is misaligned reads from TP species. mOTUs3 exhibits a similar pattern, but more pronounced. mOTUs3 has the highest FP count (n=1075) of which 88% are neighboring a TP with a mean abundance of 0.048±0.104. MetaPhLAn 4 has the lowest of FP count of 81. For protal, the remaining 7 FP that are closest to a FN+ most likely stem from reads that are simulated from genomes having similar distance between species clusters and thus being hard to classify.

Considering FP species with a neighboring TP and FN with a phylogenetic tree distance less than 0.05 and an abundance higher than 1%, protal has 36 FPs, mOTUs3 has 574 FPs, and MetaPhLAn 4 has no FPs with these parameters. I paired FPs with TPs under the assumption that FPs can be a byproduct of neighboring TP species and imperfect alignment. However, some FPs have a high distance to any TP in the tree. For MetaPhlAn 4, the reason is that some FPs and FNs were not merged during adjustment as their distance is above the threshold (see 3.2 for details). For protal and mOTUs3, these FP have a closely related FN neighbor that

is present in the tool database and thus could be detected.

Lastly, I assessed the ability of protal to reconstruct correct bacterial abundances in metagenomes. Similar to the analysis in Chapter 3, I differentiate between metrics that incorporate FPs and FNs (BC, L2, PC) and metrics that consider only TP taxa (L2-TP, PC-TP). At the species level, protal shows the best performance with a mean PC value of 0.897±0.251 (higher is better) and a mean L2 value of 0.102±0.165 (lower is better), surpassing the second-place MetaPhlAn 4, which has values of 0.806±0.338 and 0.152±0.211, respectively (Fig. 5.10 B). Protal also achieves the highest mean BC value of 0.953±0.134, compared to 0.931±0.124 for the second-place mOTUs3. However, for the metrics that only consider TP abundances, protal is slightly behind other tools, with a mean PC-TP value of 0.977±0.059 and a mean L2-TP value of 0.056±0.056, meaning that in estimating purely abundances of taxa, protal is slightly worse than either mOTUs3 or MetaPhlAn 4. This deficit is more than compensated by protal's more accurate species detection.

5.3.5 Strain-level evaluation

In the previous section I showed that protal is able to correctly align reads against a universal marker gene database and further demonstrated that these alignments can be translated to precise and accurate species profiles. In this section, I benchmark how well individual read alignments for all predicted species translate to strain-resolved trees.

Similar to Chapter 3.3.3, I used a simulated dataset of 200 metagenomes, each containing a single strain per 46 different species. With many samples containing reads from the same strain, I benchmarked the correct reconstruction of monophyletic clades per strain, errors in MSAs, within strain pairwise distances between samples, and compared tree topology with gold standard trees for StrainPhlAn 4 [26] and protal. As protal currently employs only little filtering on which samples are incorporated in the MSA, I both assessed protal's native unfiltered strain-level performance, as well as a filtered performance where protal trees are trimmed to the same tips as StrainPhLAn 4.

Sensitivity for detecting low-abundant strains

Protal's unfiltered strain-level results display a much higher sensitivity across species with a mean value of 0.959±0.085 for strains and 0.907±0.174 for samples, compared to StrainPhLAn 4 with 0.497±0.211 and 0.18±0.044, respectively (Fig. 5.11 A). In total, protal retains 90.67% of all samples (8,342 out of 9,200), 94.38% of all strains (1,260 out of 1,335), and detects all 46 species. StrainPhlAn 4 only retains 17.98% of samples (1,510 out of 8,400), 42.17% of strains (525 out of 1,245), and 42 out of 46 species. The difference in total samples (9,200 vs. 8,400) and strains (1,335 vs. 1,245) is caused by StrainPhlAn 4 failing to detect four species and their strains and

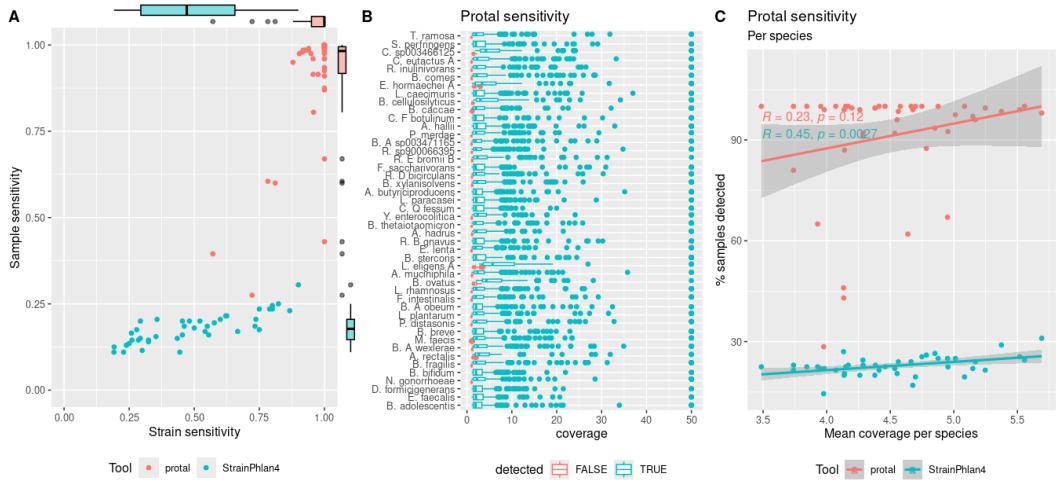

Figure 5.11: A, Per species sensitivity, measured by how many samples and strains are present in the tree. B, protal per sample true vertical coverage stratified by species and colored by presence (blue) or absence (red). C, per species percentage of detected samples (y-axis) and mean true vertical coverage (x-axis).

samples were excluded for the total number. For three species, protal has below 50% sample sensitivity (*s__Lachnospira eligens_A* with 27.5%, *s__Bacteroides ovatus* with 39.5%, and *s__Collinsella sp003466125* with 43%). Although those species have a low mean vertical coverage across samples (3.98, 4.13, and 4.13, respectively), other species like *s__Bifidobacterium bifidum* have a lower mean vertical coverage of 3.485 at a much higher sample sensitivity in protal of 98.5% (Fig. 5.11 B). This indicates that beyond coverage, there is a species-specific component that determines detectability of samples, most likely caused by varying success of read alignment. This is even more interesting considering that protal uses universal marker genes, so the difference lies in the genetic similarity of marker genes in certain areas of the tree, leading to a different number of recovered reads at the same vertical coverage. *s__Lachnospira eligens_A* for example, has four samples with a vertical coverage >3x that are missing from protal's MSA. This is surprising, given that the mean vertical coverage of undetected samples across all species is 1.425±0.429 and the minimum detected coverage is 1 (which is the lower vertical coverage bound set for the simulated dataset). Of 200 samples with a vertical coverage of exactly one, ∼66% of species (133) are detected by protal. Of all undetected samples with a coverage >2x (n=81), appears 31 belong to *s__Bacteroides ovatus*, 28 to *s__Lachnospira eligens_A*, and 15 to *s__Enterobacter hormaechei_A*.

Analysing monophyly scores

Next, I assess the monophyly score for protal and StrainPhlAn 4 on the full trees, and on filtered trees only with shared tips (see Section 3.2.2 for details). Across all samples of all species, protal exhibits a mean monophyly score of 0.732±0.327, which

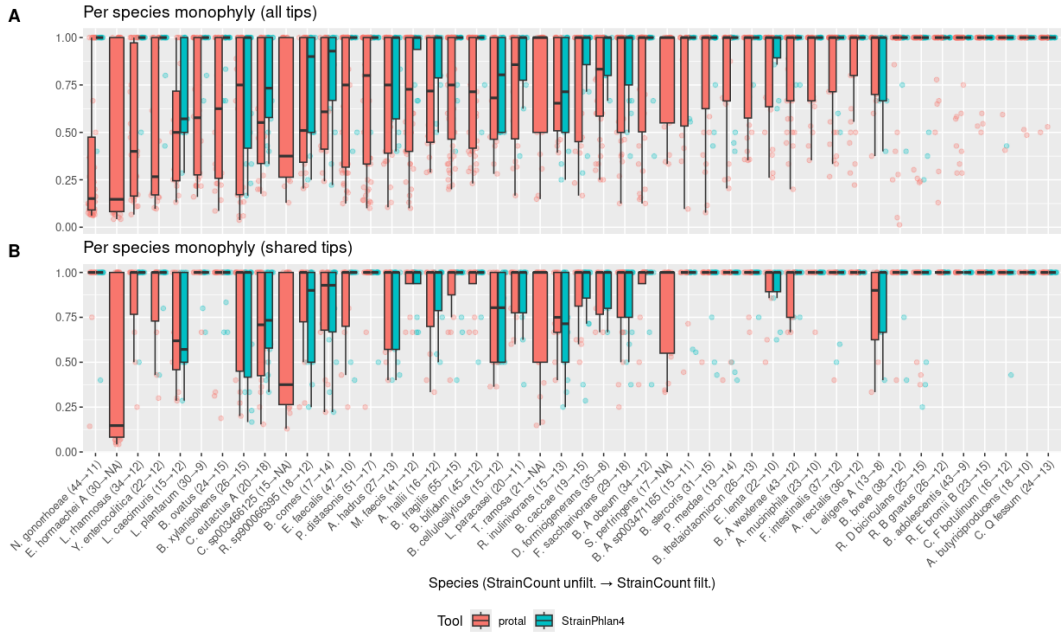

Figure 5.12: Per strain monophyly score of protal and StrainPhlAn 4 stratified by species. Monophyly score measures how pure clusters of samples carrying the same strain in a tree are (see Section 3.2.2 for details). A, considering all tips. B, trees are subset to tips shared between protal and StrainPhlAn 4 per species. Species are ordered by ascending mean monophyly score.

is lower than the mean monophyly score of 0.897±0.202 of StrainPhlAn 4 (Fig. 5.12 A). However, when assessing only tree tips present in both StrainPhlAn 4 and protal, protal's mean monophyly score increases from 0.732±0.327 to 0.887±0.217 (Fig. 5.12 B). For *s__Neisseria gonorrhoeae*, protal's mean monophyly score increases from 0.342±0.36 to 0.899±0.262 while reducing detected strain count from 44 to 11. Only *s__Lachnospira eligens_A* has a decrease in monophyly score after filtering with a difference of 0.08493. Out of all species, 14 species have an increase of less than 0.1 mean monophyly score, 4 species have an increase higher than 0.3 (Supp Fig. A.15).

Cross-mapping causes low monophyly scores

To further analyse why species perform differently with respect to sample sensitivity and monophyly scores, I looked at the marker gene read alignments from Chapter 5.3.1, where reads were simulated from all marker genomes (not only species representatives) within GTDB. I picked *s__Clostridium_Q fessum* and *s__Bacteroides ovatus* as both perform very differently with respect to sensitivity and monophyly. I selected *s__Clostridium_Q fessum* for its high mean monophyly score of 0.959±0.136 and 100% sample sensitivity, despite its low mean coverage of 3.738±6.34. In contrast, I picked *s__Bacteroides ovatus* due to its low mean monophyly score of 0.615±0.363, and low sample sensitivity of 46% despite a high mean coverage of 7.076±7.28. Like in Chapter 5.3.1, alignments are considered correct if

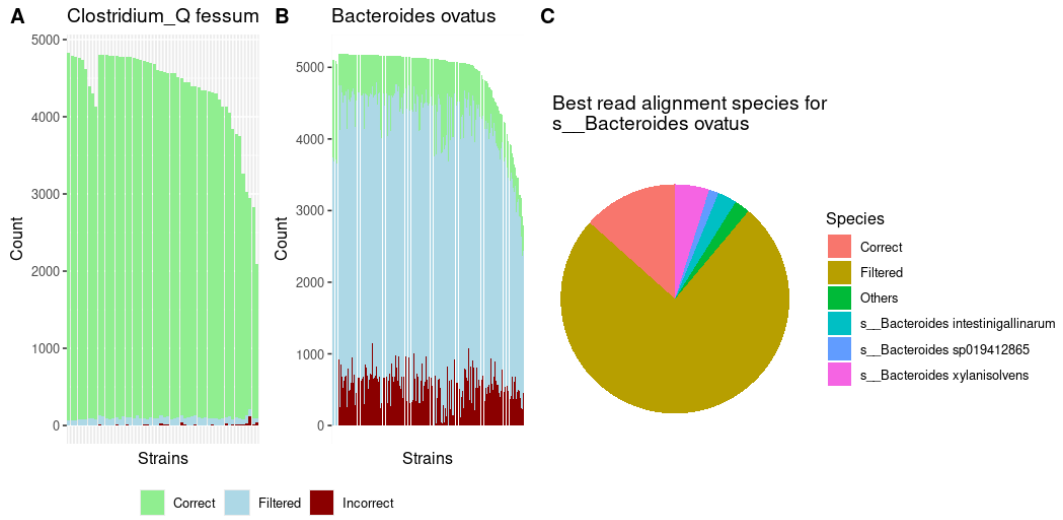

Figure 5.13: Protal's read alignments for all marker genomes under *s__Clostridium_Q fessum* (A) and *s__Bacteroides ovatus* (B) against the database containing all representative marker genomes. Alignments are classified correct or incorrect based on whether they align to their respective species cluster representative. Alignments with a MAPQ value lower than 4 are filtered (protal default). C, Proportions of read alignments of all marker genomes under *s__Bacteroides ovatus* with respect to the species they aligned best to. This figure both quantifies how many reads are retained after filtering, and how many reads align to other species, potentially leading to FPs species detection and chimeric signal in MSAs (in this case for example for *s__Bacteroides xylanisolvens*).

they align against the species representative marker genome of their original species cluster.

Across all alignments for reads simulated from marker genomes of *s__Clostridium_Q fessum*, the mean percentage for correct alignments out of all reported alignments is 97.635±0.848%, 2.082±0.406% of read-pairs were filtered, and a mean of 0.337±0.635% of read-pairs were aligned to a different species (Fig. 5.13 A). For *s__Bacteroides ovatus* only a mean percentage of 13.338±5.012% of reads were correctly aligned, 75.477±2.148% were filtered due to protal's default MAPQ threshold, and 11.507±4.533% aligned onto other species (Fig. 5.13 B). When looking at where the incorrect read alignments from read-pairs of *s__Bacteroides ovatus* went, I found that 43.5% percent of all incorrect alignments (after filtering) aligned with the neighbouring species *s__Bacteroides xylanisolvens*.(Fig. 5.13 C). To ensure that this is not caused by protal's custom alignment, I further aligned these reads with bowtie2. This showed a qualitatively similar result, and with the same MAPQ ($\geq$ 4) filtering threshold the mean percentage of correct classifications is 28.508±9.636, incorrect classifications is 27.429±10.156, and an average 39.236±1.981% of reads were filtered (Fig. A.11). Further, the incorrect alignments were also mostly aligning to *s__Bacteroides xylanisolvens*, similar to protal's alignments.

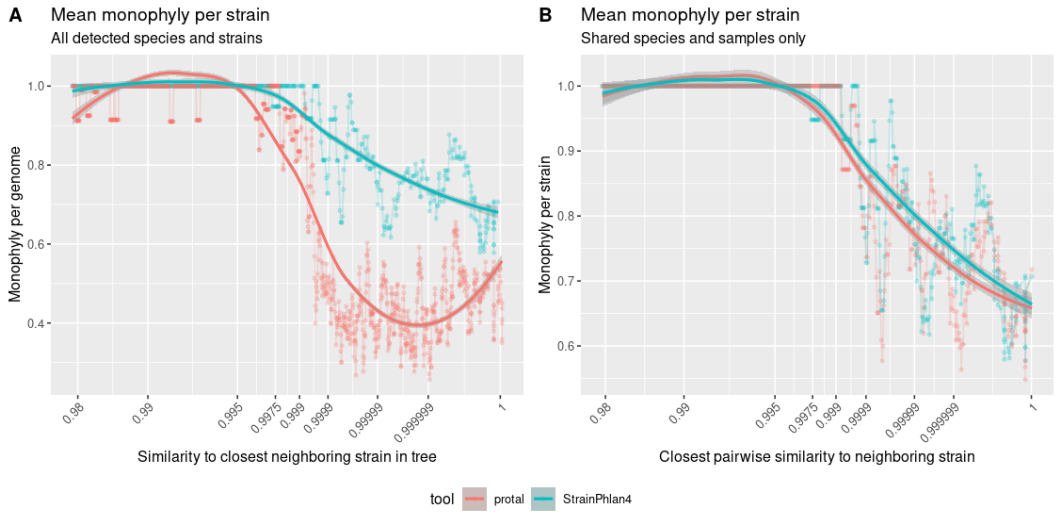

Figure 5.14: Monophyly analysis with respect to closest neighboring strain. Each data-point is the mean monophyly of a sliding window of 10 samples per tool (y-axis) with samples sorted by closest pairwise similarity to neighboring strain (x-axis). Values on the x-axis are rank-transformed and label-placement is according to the closest matching point in the data. Mind that samples from the same strain are simulated with sequencing errors, but otherwise genetically identical. A, for each tree, all samples are included. B, for all trees only samples shared by both protal and StrainPhlAn 4 are considered. The fitted line is a loess regression.

In GTDB r214, 203/250 genomes classified as *s__Bacteroides ovatus* are isolate genomes, 184 out of 221 genomes of *s__Bacteroides xylanisolvens* are isolate genomes. Within the genus *g__Bacteroides* 1948 out of 2561 are isolate genomes. It remains unclear whether the observed misalignments stem from assembly artifacts—such as chimeric contigs in MAGs—or from genuine biological processes like homologous recombination. During the read-alignment evaluation, I only distinguished misalignments by species and gene, not by their source genome. Consequently, further work is needed to compare leakage levels between reads simulated from isolate genome marker genes and those from MAGs.

It has to be noted, that all alignments are correct in the sense that they align to the reference with the highest sequence identity. By filtering with alignments with a MAPQ-threshold, all alignments that are highly ambiguous were removed; the remaining alignments should have a higher confidence. This suggests that some species clusters within *g__Bacteroides*, as provided by GTDB, have substantial genetic overlap in their conserved core marker genes.

An important metric for *de novo* strain-level tools is the ANI resolution at which they still can reliably resolve closely related strains, as this means higher accuracy for tracking strains. For this, the following analysis put monophyly scores in context with the true phylogenetic distance within the gold-standard phylogenetic tree as constructed with roary. Pairwise phylogenetic distances hereby serve as proxy of

%ANI. I compared protal against StrainPhlAn 4 with protal's unfiltered output, as well as all phylogenetic trees subset to shared tips (Fig. 5.14 A, B).

For the unfiltered benchmark, there are eight strains for which protal has a non-perfect monophyly score with their closest neighboring strain less similar than 99.5% in the roary gold-standard tree. Six of those strains belong to *s__Bacteroides xylanisolvens* and the other two to *s__Bacteroides breve*. This is in line with the previous results and is likely due to species cross-mapping of reads. For strains with 99.5% or greater similarity, I see a decline in monophyly score for both protal and StrainPhlAn 4, albeit this decline is more extreme for protal. Protal has a monophyly score of 0.4 (fitted loess regression) for the most closely related strains while StrainPhLAn 4 is better at resolving strains with its monophyly score not going below 0.6. If I compare both tools only for shared species and shared tips, both exhibit a perfect monophyly score for all strains with the closest neighbor having a similarity of 99.75% or lower. For more closely related strains the monophyly score degrades equally below ∼0.7 when distinguishing between the most closely (>99.999999% ANI) related strains.

Strain Cluster Error within Phylogenetic Trees

Similar to Chapter 3.3.3, I next compared the pairwise distances of samples carrying the same strain, vs. samples carrying different strains by calculating the Max Cluster Error (MCE) as explained in Section 3.2.2 (see Fig. 5.15). There are two signals that are apparent. First, *s__Bacteroides xylanisolvens*, *s__Bacteroides ovatus*, *s__Bacteroides breve*, and *s__Blautia_A sp003471165* display large MCEs up to 0.018, but not necessarily many. Second, the remaining species have occurrences of positive MCE ranging between 34 (*s__Neisseria gonorrhoeae*) and 2 (*s__Clostridium_C Q fessum*, *s__Clostridium_F botulinum*, *s__Agathobaculum butyriciproducens*), however, never exceeding a MCE of 0.00098. Considering the previous analysis investigating read alignments, *s__Bacteroides xylanisolvens* was one of the species that was falsely represented by reads that were derived from *s__Bacteroides ovatus* genomes. In the light of alignment cross-mapping, it makes sense that both species have a larger MCE, as could be considered a baseline noise from sequencing errors.

Bacteroides strains exhibit errors in MSA

Looking at protal's MSAs directly there was no clear association between alignment length and error rate (Fig. 5.16 A). Errors are columns in the MSA between consensus sequences of the same strain, that are not monoallelic. It follows that the error rate is the number or errors divided by the consensus length.

Error rate was highest for *s__Bacteroides xylanisolvens* with a mean value of 0.00184±0.00119. As previously shown, the error rate is most likely caused by reads

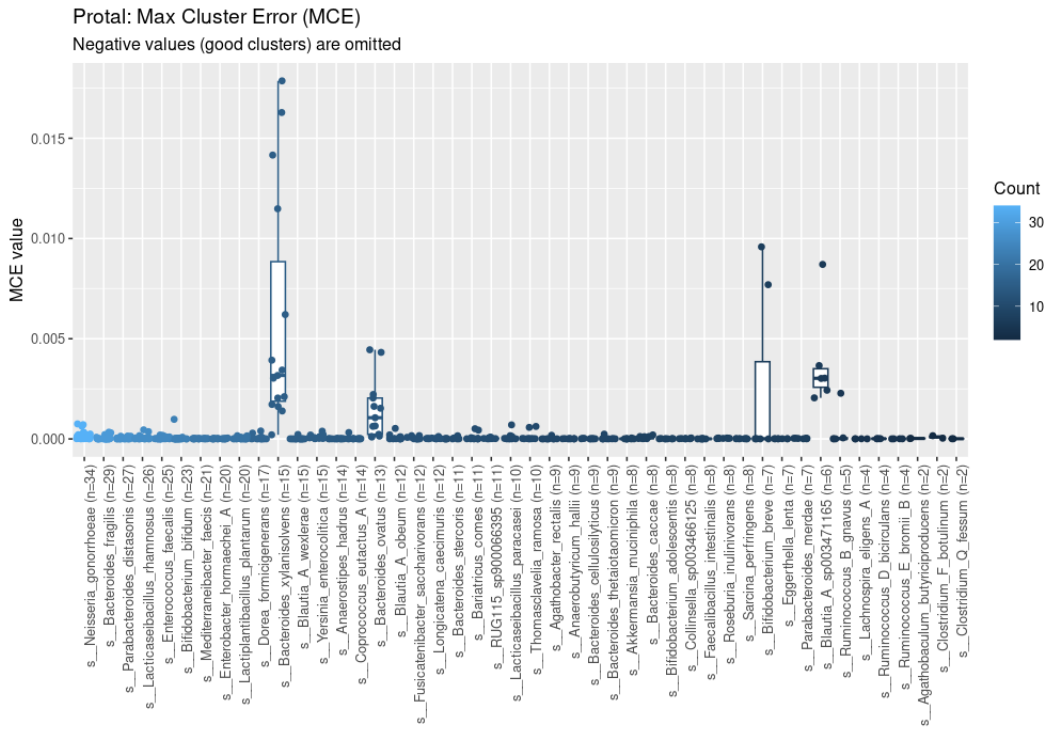

Figure 5.15: Max Cluster Error (MCE) on the y-axis per strain and stratified by species. MCE is computed as explained in Section 3.2.2 and quantifies how well all samples with a certain strain cluster together in the phylogenetic tree with respect to all other samples. The color encodes the number of strains with a positive MCE. Negative MCEs (good clusters) are discarded.

falsely aligned to *s__Bacteroides xylanisolvens*, creating a wrong consensus DNA sequence reported by protal. Of other species, *s__Blautia_A sp003471165* has a mean error rate of 0.00075±0.00046 and *s__Bacteroides ovatus* has a mean error rate of 0.00046±7e-04. As expected, the raw number of errors supports this ranking with *s__Bacteroides xylanisolvens* having a mean error count of 85.22±54.71. *s__Bacteroides ovatus* has the lowest mean alignment length with 12591±5856 nucleotides. The mean alignment length across all species is 66265±36598 and thus substantially higher than StrainPhlAn 4's mean alignment length of 5401.786. While protal does not filter any columns in the MSA to retain as much signal as possible, StrainPhlAn 4 has a stricter filtering approach. StrainPhLAn 4 has a higher mean MSA error rate of 0.00156±0.00369 than protal, which has a mean error rate of 0.000066±0.00036. The reason why this does not translate to a lower monophyly score on the side of StrainPhlAn 4 is that their species-specific marker genes are less conserved, and the between strain-specific signal (positions that are different) is a stronger. Protal needs to maintain a low error rate as a higher error rate would drown out the true signal resolving the inferred strains from metagenomes (Fig. 5.16 B).

Comparing monophyly scores with error rate indicates there is a potential neg-

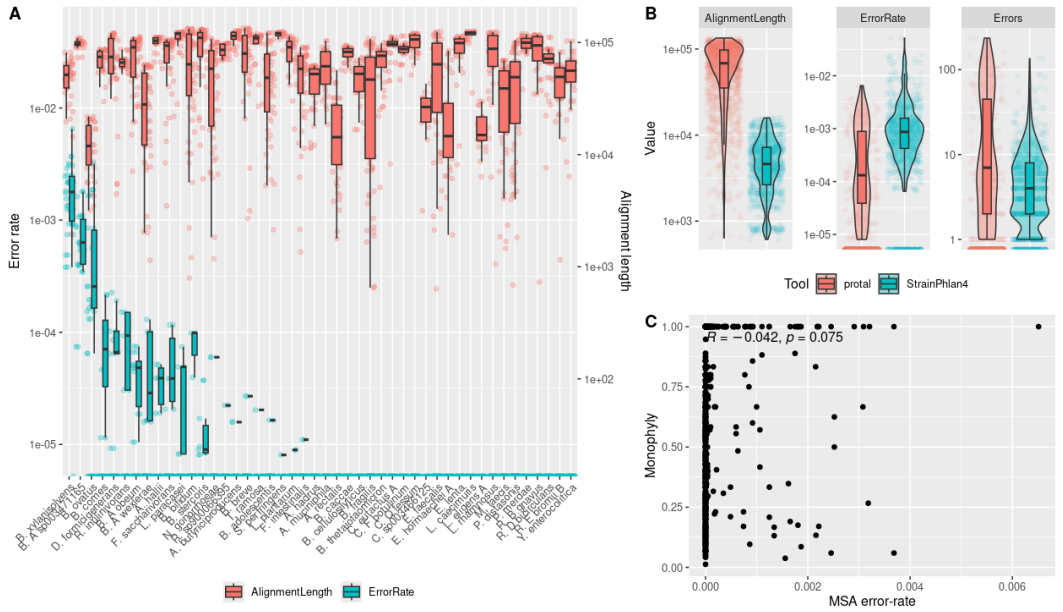

Figure 5.16: A, Error rate and alignment length for each sample for protal's MSAs per species. B, alignment length, error rate, and total errors within MSAs of both protal and StrainPhlAn 4 of strains across all species. C, monophyly of protal's trees per strain in context of MSA error rate. Correlation computed with pearson correlation. See Table A.2 for species abbreviations.

ative correlation (r=-0.042), however this was not significant (p-value = 0.075, Fig. 5.16 C). Interestingly, StrainPhlAn 4, too, shows a high error rate in MSAs and a low monophyly for *s__Bacteroides xylanisolvens* and *s__Bacteroides ovatus*. This suggests that some species have a high rate of 'false' alignments to neighbouring species regardless of the strategy for selecting marker gene.

In the next benchmark, I compared protal's and StrainPhlAn 4s trees with respect to topology against the roary gold standard trees using the five metrics RF normalized, weighted RF normalized, SP, weighted SP, and KF distance (see Chapter 3.2.2). All samples in protal's and StrainPhlAn 4's trees were collapsed per strain by removing them and keeping their LCA to represent the strain. This was necessary to compare the trees to the roary gold-std trees, which are built from the strain genomes. I again benchmarked protal without filtering as well as protal with trees filtered to the samples present in StrainPhlAn 4's trees. For the three metrics wRFnorm, wSP, and KF, which consider branch lengths, protal's filtered trees have a significantly lower distance (p-value < 0.001 for paired t-test) to the gold-standard than StrainPhLAn 4 (Fig. 5.17 A). For the metrics RFnorm and SP, which only assess the topology without incorporating branch lengths, there is no significant difference between protal's and StrainPhlAn 4's performance (mean RFnorm for protal is 0.491±0.195 and 0.43±0.184 for StrainPhlAn 4, mean SP for protal is 30.504±17.575 and 28.197±14.639 for StrainPhlAn 4). Comparing the unfiltered trees, protal is still significantly better for the metrics considering branch lengths with

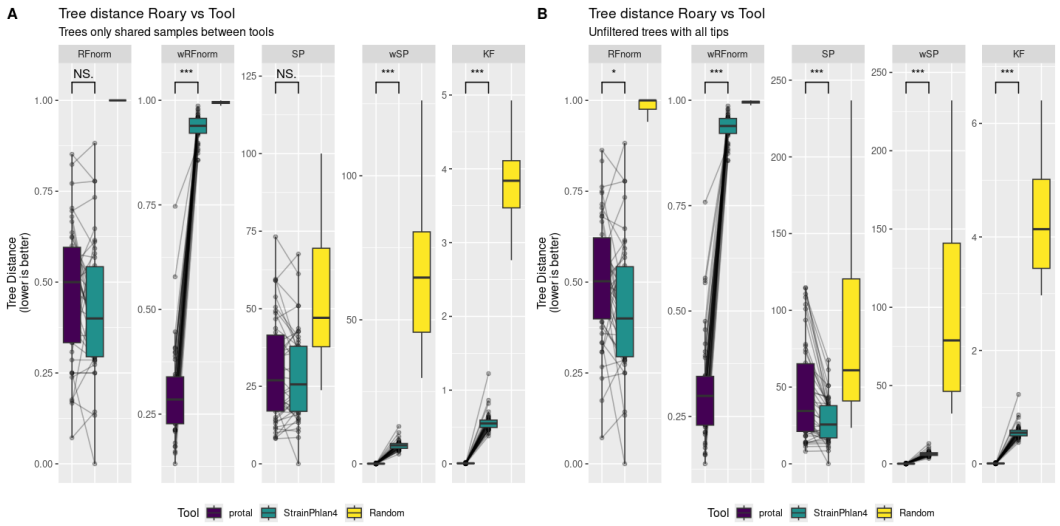

Figure 5.17: A, distance between the trees of protal, StrainPhlAn 4, and randomly generated trees to the gold standard tree. All trees are subset to samples shared between protal and StrainPhlAn 4 and only trees for species that are predicted by both tools are considered (n=42). B, same as A, but protal and StrainPhlAn 4 trees are not subsets to shared samples. Significance was calculated with a paired t-test (*: $p \leq 0.05$, ***: $p \leq 0.001$, n=46 for protal, n=42 for StrainPhlAn 4), with species between protal and StrainPhlAn 4 as pairs. For B, species that are not shared were removed from the test. The utilized metrics are normalized Robinson-Fould distance (RFnorm), normalized weighted Robinson-Fould (wRFnorm), Steele and Penny distance (SP), weighted Steele and Penny distance (wSP), and Kuhnert-Felstenstein distance (KF) (see Chapter 3.2.2 for details).

mean distances of 0.314±0.126 for wRFnorm, 0.131±0.107 for wSP, and 0.01±0.005 for KF (StrainPhlAn 4 with 0.933±0.032 for wRFnorm, 6.492±1.717 for wSP, and 0.572±0.14 for KF). On the metrics RFnorm and SP StrainPhlAn 4 has a significantly lower distance (p-value < 0.05 for RFnorm and p-value < 0.001 for SP for paired t-test) with 0.43±0.184 (protal 0.509±0.172) and 28.197±14.639 (protal 46±32.149), respectively. In the shared tips benchmark, for 14 out of the 42 shared species, protal has a lower RFnorm distance compared to StrainPhlAn 4 (mean lower distance of 0.119±0.1 for protal), 4 species have the same distance, and 24 species have a higher distance to the gold standard compared to StrainPhLAn 4 (mean lower distance of 0.124±0.097 for StrainPhlAn 4). In the all-tips benchmark, protal has a lower RFnorm for 15 species (mean lower distance of 0.109±0.079 in favor or protal), StrainPhlAn for 27 species (mean lower distance of 0.159±0.105 for StrainPhlAn 4). This shows that protal's ability to reconstruct the topology is often on-par or better than StrainPhlAn 4, but has more variance in performance. Further, protal's tree distances are much more similar to the gold-std tree distances, offering a viable proxy for ANI. For StrainPhlAn 4, however, their distances are skewed, especially for higher distances. This is a result of StrainPhLAn 4's sub-sampling of positions in the MSA that are uninformative.

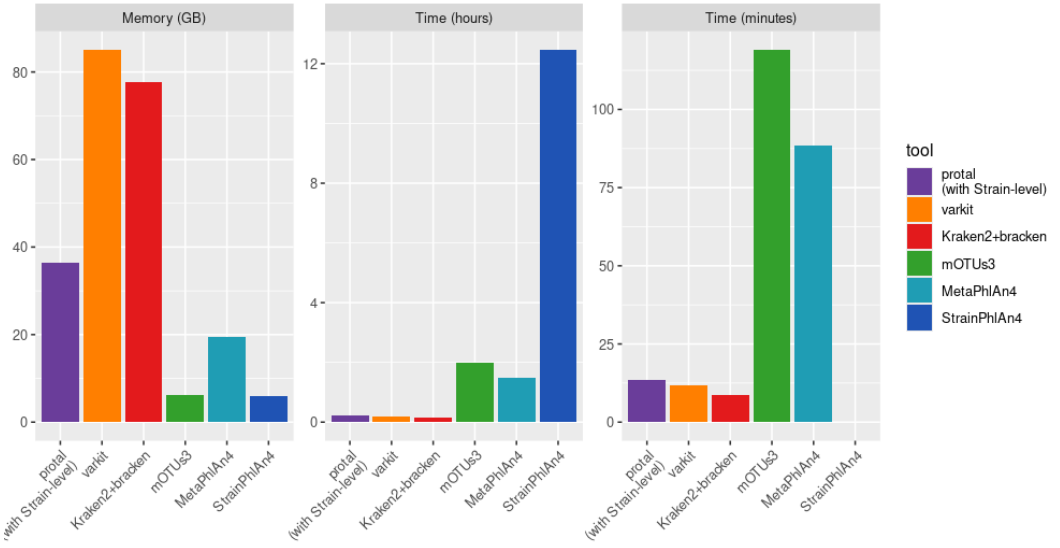

Figure 5.18: Runtime and memory analysis of protal with respect to varkit, other taxonomic profilers, and strain-resolved tools. The benchmark was done on a single node with no interfering input and output using 16 cores. The tools were run on 10 samples from CAMI Airways with 2x 5GB uncompressed paired-end reads (see Section 3.2.7 for more details).

5.3.6 Runtime

Similar to Chapter 4, I chose to include Kraken2+Bracken in the memory benchmark as a reference. On a dataset with 10 uncompressed paired-end reads with 2x5GB size each, protal was the third fastest with a runtime of 13min 30sec (Fig. 5.18). Compared to varkit, protal is 2 minutes slower. However, it has to be considered that protal outputs alignments, taxonomic profiles, and reconstructed MSAs. Kraken2+Bracken came in first with 8min 30sec, MetaPhlAn 4 was third with a runtime of 1h 28min and mOTUs3 was fourth with 1h 59min. StrainPhlAn 4 takes the longest with ∼12h 30min and in comparison, protal is approximately 55 times faster. The memory consumption for protal is moderate with 36 GB and is between Kraken2+Bracken with 77 GB on the upper end, and MetaPhlAn 4 with 19 GB on the lower end. StrainPhlAn 4 only required 6 GB of memory.

5.4 Discussion

Protal demonstrates important advancements in reference-based metagenomic profiling and strain-resolved analysis, offering superior sensitivity for lesser-studied species through the comprehensive integration of the GTDB taxonomy. By combining a custom alignment and k-mer based approach with universal marker genes, protal not only enhances taxonomic coverage but also achieves significantly faster performance compared to existing tools.

Flex-map and technical approach

I used a custom hash table design and alignment to significantly speed up alignment and taxonomic profiling. Using a hash table over the often employed FM-index has led to performance gains in multiple aligners like minimap2 [145], strobealign [234], and accel-align [312]. Further, FM-index approaches are less memory intensive and more sensitive, albeit slower than their hash table based counterparts. This can be seen in the publications of centrifuge [123] vs. kraken [308], bowtie2 [137] and bwa2 [146] vs. minimap2 [145]. Nonetheless, hash table based approaches were popular due to their performance, especially considering the growing amounts of data to analyse. Kraken2 implemented a probabilistic hashing to speed up taxonomic classification and k-mer sub-sampling to reduce the memory footprint, countering the ever-growing reference databases [307]. Inspired by this idea, I developed and implemented the flex-map to expedite the seeding phase in alignment without compromising sensitivity. Further, I am unaware of any other bioinformatics tool that combines exact matching k-mers with flexible matching in the same data structure and processing step. I also used the remaining space per k-mer entry to indicate it's uniqueness given its species cluster. This information is not used for alignment, but later retrieved for the taxonomic profiling to improve precision for species prediction. This was inspired by KrakenUniq [32], which implements a counting of unique k-mers to improve precision, and GT-Pro [250], which pre-selects k-mers based on criteria to do both taxonomic profiling and genotyping. Protal's approach is unique and would be impossible with an external alignment tool. Instead two programs would need to be run, one for extracting and identifying unique k-mers, and another one for performing the read alignment. Protal's balance of speed and precision is currently unmatched. Tools of similar precision such as MetaPhlAn 4 are slower, and faster tools like Kraken2 don't match protal's precision and sensitivity.

Speed and memory

In addition to speed, maintaining reasonable memory requirements is crucial to ensure accessibility for users with limited access to computational resources. For GTDB marker genes, protal's index size is roughly twice as big as bowtie2 and minimap2, but only about half of that of bwa-mem2's index. In practice, protal currently requires at least 64GB memory to run, while bowtie2 and minimap2 run with 32GB memory. With respect to taxonomic profilers, MetaPhLAn 4 uses bowtie2 and requires at least 32GB. mOTUs3 is the only tool that can truly be operated on a personal computer using 16GB of memory or less. This is due to mOTUs3's small set of reference sequences with only ten marker gene sequences per species, impacting precision, as previously shown. Kraken2's standard database is 60GB in size, however, this encompasses whole-genomes as opposed to marker gene databases and hence Kraken2's database on marker genes would likely be a lot smaller. Since

32GB or more of memory space remains exceptional for the current generation of personal computers, all tools except mOTUs3 must be executed on machines with high memory capabilities, such as a dedicated computing cluster.

Taxonomic profiling with random forests

For predicting whether a taxon is present in a metagenome, taxonomic profilers often apply thresholds on parameters such as number of reads or marker genes present. However, the complex signals from read alignments, comprising of ANI, MAPQ, number of hits, number of genes, abundance distribution of genes, and, in the case of protal, unique k-mers are not easily converted into taxa predictions. Random forests on the other hand do not require data normalization and provide an accessible solution for this problem. In protal, the random forest predicts taxa with high sensitivity and high accuracy, mostly using information from unique k-mers. While I did a split between training and test data and the performance remained stable across training a random forest on multiple splits, systematic cross-validation is necessary to confirm the results. Further, as the random forest is used on the data it was trained with, a certain degree of cross-mapping is to be expected. Due to time constraints, I was unable to generate additional test data to simulate the complexity found in the CAMI datasets, which necessitates further assessments to ensure that the results are reproducible on more independent datasets and the model does not overfit. Another issue is related to the unique k-mer information. As demonstrated, some species only have few unique k-mers in the database and prediction of those species will likely require more aligned reads than other species with a high unique k-mer count. In addition to the signal from cross-mapping, a solution is still required to address this problem.

Reference-guided MSA

For the small speed benchmark dataset (10 samples), protal is able to reconstruct phylogenies faster than StrainPhlAn 4. As the runtime for building MSAs with protal scales linearly, it is also suited for larger datasets with > 1000 samples. StrainPhlAn 4 will likely take a lot longer as the datasets grow, hence being most suitable for small- to medium-sized datasets. The speed advantage ultimately hinges on the sample size and the extent of shared species between samples. However, further evaluations are needed to determine whether internal SNP calling or MSAs contribute more significantly to this speed benefit. Many issues with protal's MSAs are likely caused by cross-mapping. While the incorporation of unique k-mer information helps to detect the correct species, it does not prevent cross-mapping, which still affects the performance on strain-level.
Additionally, shortly before completing my PhD, I encountered a bug in the alignment process for reference-guided multiple sequence alignments (MSAs). This bug

resulted in the insertion of 'N's in numerous positions where references should have been correctly placed. Although I was unable to revise the results for Protal in my thesis, an initial assessment indicated a slight enhancement in the MSAs and phylogenetic trees. Nonetheless, the main results remain unchanged.

Marker gene cross-mapping

I reported that species of the genus *g__Collinsella* had below average detection rates and accuracy in the MSSS200R dataset, and further, that the subtree of *g__Collinsella* in GTDB has many closely related species. For *s__Bacteroides ovatus*, I showed that many of the short reads simulated from its member's marker genes best aligned to the species representative marker genes of its neighboring species *s__Bacteroides xylanisolvens*. Further, by investigating the marker gene alignments, cross-mapping was shown to be the problem for many species clusters in the database. In Chapter 6, I will address potential technical and biological causes for this signal and propose strategies to resolve this issue.

5.5 Author contributions

I developed and implemented protal. Concepts such as the flex-mer data structure were developed by myself, with insights drawn from discussions with Falk Hildebrand. The idea of storing any extra information in the hash table was inspired by Falk Hildebrand, which lead me to store additional genetic information around k-mers for higher precision during seeding and additional information on k-mer uniqueness for higher precision in taxonomic profiling. Monophyly based benchmarks were inspired by Falk Hildebrand as a means to assess the tool's performance independent of ANI values. Strain-level benchmark datasets were designed by myself. I conceptualized reference-guided MSAs from short-reads to avoid a costly regular MSA.

Chapter 6

Critical Assessment of Work

6.1 Overview and chapter summaries

Taxonomic profiling and strain-resolved analyses are crucial to many discoveries in microbiology and metagenomics. Yet, problems like database bias [288] and contamination [33] are two bottlenecks that can lead to a spurious detection of taxa and confound subsequent analysis [82]. This challenge is compounded by the fact that various metagenomic profilers employ different taxonomic systems and reference databases, complicating informed tool selection and data confidence. Independent efforts for benchmarking taxonomic profilers exist [288, 177, 242, 316], but none of these provided an in-depth explanation on the true source of false predictions. Further, while accuracy in profiling has increased, the improvement in speed of the most popular taxonomic profilers and strain-resolved tools has stagnated. However, particularly for strain-level analysis, computational efficiency can facilitate an unbiased analysis. Researchers limited by computational resources might have to adjust the scope of their analysis and set a focus instead of exploring their data.

In this work I benchmarked the performance and explored the sources of false predictions for three popular taxonomic profilers with benchpro. With varkit, I proposed a fast and novel alignment-free approach for taxonomic profiling and SNP detection in marker genes. Lastly, based on insights from my previous work and employing novel algorithms, I developed protal, an ultra-fast alignment-based profiler and tool for strain-resolved analysis, that allows for sensitive and precise analysis. This chapter gives a concise summary of the work covered in the three results chapters. Finally, I discuss the findings, the contributions of this research, and the implications for future studies within a broader context.

6.1.1 Benchmarking of metagenomic profilers with benchpro

In this chapter, I presented benchpro, a software for benchmarking tools for metagenomic profiling and strain-resolved analysis. By using the analysis capabilities of benchpro on simulated CAMI datasets[177], I compared three popular tools for

taxonomic profiling: MetaPhlAn 4[26], mOTUs3[232], and Kraken2+Bracken[307, 159]. By comparing all tools both using the NCBI taxonomy[69] and the GTDB taxonomy[206], I was able to isolate the performance difference caused by the different taxonomies and mapping between these systems. GTDB provides resources that allowed me to examine phylogenetic patterns in false predictions. Benchpro was able to identify and account for pairs of FP and FN species that only arose from differences in species-level resolution between databases, but not from a general lack of database coverage for those species. This facilitated the identification of FP predictions that arose from ambiguous short-read mappings between species. Further, a sliding abundance threshold for filtering low-abundant taxa revealed that Kraken2+Bracken is less accurate than mOTUs3 and MetaPhlAn4 at any threshold. MetaPhlAn 4 was the most accurate tool across all benchmarks, and exhibited the lowest rate of FP prediction caused by cross-mapping between closely related taxa.

On strain-level, I used simulated metagenomes to investigate the performance of StrainPhlAn 4[26] using ANI independent measures such as contamination in monophyly and MSA errors, as well as phylogenetic distance to gold-standard phylogenetic trees. This revealed a species-specific error signal, independent of the dataset. Lastly a runtime and memory benchmark showed that Kraken2+Bracken is fastest, and comprehensive strain-resolved analysis for all species with StrainPhlAn 4 takes significantly longer than species-level profiling.

6.1.2 Alignment-free taxonomic profiling and SNP detection with varkit

In this chapter, I explored a novel alignment-free k-mer based method for taxonomic profiling and SNP calling, called varkit. Varkit utilizes the hit patterns of sequential k-mers shared between read and reference to infer SNP positions. Varkit's database stores all k-mers of marker genes along with their position and taxonomic identity. Non-unique k-mers point to the LCA of all occurrences in the reference. Varkit further employs a custom hash table with a reduced memory footprint for storing k-mers. A pattern database, pre-built using a specific k-mer shape, contains key-value pairs of sub-patterns and SNP positions and is used to infer SNP positions. The pattern database is used to retrieve the relative SNP positions in a read by looking up each sub-pattern observed between read and reference. I selected the k-mer shape based on an extensive analysis of SNP detection sensitivity with respect to its size and number and position of gaps. This showed that irregular patterns of gap positions increase sensitivity, and longer k-mers decrease sensitivity. An analysis of simulated reads from genomes with a controlled ANI to the reference further showed that SNP detection sensitivity steadily declines with increasing distance to the reference. This is more pronounced for species with a lower rate of species-level k-mers. Next, I used benchpro to assess the taxonomic profiling performance of

varkit using a database built from GTDB r207 marker genes[206], using a similar benchmarking approach as in chapter 3. Varkit exhibits an elevated false positive rate for specific species; however, overall, it performs comparably to mOTUs3 and better than Kraken2+Bracken. A runtime benchmark demonstrated that varkit is faster than MetaPhlAn 4 and mOTUs3, and comparable in speed to Kraken2+Bracken, with a similar memory footprint.

6.1.3 Alignment-based taxonomic profiling and strain-resolved metagenomics with protal

In this chapter I presented protal, an alignment-based tool for taxonomic profiling and strain-resolved analysis I developed. Protal uses a novel data-structure, the flex-map, to find hits between read and reference with an exact-matching 15-mer (core-mer), and then filters further by inexact matching of the 8-mer flanking regions around (flex-mer). Protal's database is built from GTDB's universal marker genes (r214) of all species-representative genomes. K-mers which are unique for a species are identified by checking against all k-mers of all marker genomes within GTDB. Protal's alignment algorithm first looks for seeds between read and reference in the flex-map, groups them into anchors and passes the best candidate reference sequences to alignment. The alignment is performed with an external alignment library, the WFA2 aligner [167, 256]. Protal outputs the best alignment for each read, along with counts of unique k-mers identified during the seeding phase. It collects alignments and unique k-mer hits for each species and gene. A random forest algorithm then predicts the presence or absence of species in a sample using a diverse array of data.

First, I showed that protal's alignment performance is on par with established alignment tools regarding the specific use-case of marker gene alignment, and is as fast as the fastest currently available alignment tool. I tested protal's performance against MetaPhlAn 4 and mOTUs 3 using benchpro and the datasets from chapter 3 as well as an additional dataset, containing reads from rare species. Protal exhibited the highest F1-score among all tools. Deeper analysis with benchpro revealed that protal has an elevated rate of FP for specific species (e.g. *g__Collinsella*). I found evidence that this bias arises from clusters of closely related species within the GTDB database. On strain-level, I demonstrated that protal exhibits similar accuracy as StrainPhlAn 4 when comparing phylogenies of strains and samples detected by both approaches simultaneously.

6.2 Limitations and assessment of results

Taxonomic profiling is a highly complex task that involves several decisions, including the selection of reference sequences, the choice between a whole-genome or marker-based approach, as well as the choice between alignment-based and alignment-free

methods. Furthermore, it is crucial to consider the taxonomy employed, the methodology for predicting taxa from individual read mappings, and the approach for estimating their relative abundance within the sample. Arguably, the underlying database, including selection of reference, taxonomy, species-clustering, and marker regions, has the most profound impact on profiling. The database not only defines the detectable taxonomic diversity, but also the precision for taxonomic profiling and strain-resolved analysis.

Therefore, it is concerning that for some species alignments with both protal and bowtie2 (Fig. A.14, Fig. A.12) showed large amounts of cross-mapping, even after MAPQ filtering. The question stands whether this is a biological signal or a technical bias in reference genomes. It has been reported that bacterial databases are contaminated with human reads [33], however, contamination mostly affected small contigs of low coverage and stems from human repeat sequences and is thus unlikely to affect universal marker genes. This also does not explain why this signal is present in some species, and absent in others. Another part of the explanation is that the species clusters in GTDB[206] are based on whole-genome ANI and the phylogeny of marker genes does not necessarily reflect these species clusters. To account for this in its own database, MetaPhlAn 4 employed checking for cross-mapping as strategy for selecting the species-specific marker genes[26]. That being said, there is also indication of some species exhibiting cross-mapping in MetaPhLAn 4 and StrainPhlAn 4. The reads were simulated to resemble the error profile of the Illumina HiSeq 2500. Simulated sequencing errors can lead to false alignments for a small percentage of reads, but do not explain the systematic signal observed.

A biological explanation for the cross-mapping might be horizontal gene transfer (HGT) through natural transformation and homologous recombination (HR). There is evidence that HR is a major evolutionary factor behind core genome evolution in prokaryotes[84] and further, that a short region of similarity is sufficient for HR[275]. Additionally, natural competence, which is required for the uptake of extracellular DNA, is not uniformly distributed in the taxonomic tree, hence affecting some taxa more than others[171]. Further investigation is warranted to understand this signal for GTDB marker genes, and potential solutions are presented in the next section 6.2.

6.3 Further work

This last section discusses which work could be done to improve the tools introduced in the three results sections. Further, potential solutions are discussed to solve the aforementioned problems with cross-mapping.

6.3.1 Benchpro

To gain deeper insights into the strengths and weaknesses of various profilers, it would be necessary to extend the benchmark to include additional tools. With also including long-read based profilers, I would gain a more comprehensive picture of the overall profiling landscape. As shown in other scenarios, species exclusion benchmarks contributed to a more rounded benchmark to test the performance in absence of an appropriate reference. To dive even deeper and separate the effect of database from the tools algorithm, I would compare read origin and read mapping after quality filtering with a MAPQ threshold to assess the percentage of cross-mapped reads. For reads simulated from a single genome that map to different species in the database, we need to disentangle whether one of the following signals is present: a) Homologous recombination in either the source genome of the reads or the species wrongly mapped to, or b) chimeric contigs (not bins) in the source genomes, if MAGs. This would allow for assessing the quality of the genomes and MAGs used by CAMI, the reference sequences in the database, and hence also the underlying tool algorithm.

On strain-level some species with strain clustering errors already exhibit an erroneous signal in the MSAs generated by protal and StrainPhlAn 4, indicating that changing iqtree for a different tree building tools such as RAxML [265], will not change the underlying errors. The current benchmarks for evaluating the distance between the generated phylogenetic trees and a reference tree—constructed using Roary—could be improved. Replacing Roary with Panaroo [276], a similar tool that also accounts for genome contamination, could lead to more accurate benchmarking. However, other evaluation metrics such as the Monophyly score and MCE would remain unaffected, as they rely solely on the predicted phylogenetic trees and the known genomic origin of each sample and species.

Two limitations hindered me from including more tools. Firstly, there are not many tools in this very category providing MSAs and a phylogenetic tree while having moderate computational requirements. MetaSNV, for example, operates on the mOTUs3 output and has much higher computational requirement, which lead to it failing in my runs due to excessive data output. inStrain works on whole genomes and has a higher complexity compared to StrainPhlAn 4 and does target a different user-base. Many other strain-level tools such as GT-Pro do not quantify novel strain-level variation but instead detect known strains. The second reason is the time constraints I encountered as I approached the end of my PhD. However, in the protal chapter I use benchpro to benchmark protal's strain-level performance compared to StrainPhlAn 4. For future benchmarks, I would still want to include more tools to also disentangle benefits and downsides of utilizing whole genomes for strain-level

resolution as opposed to a limited set of marker genes. I also intentionally omitted benchmarking tools for reconstructing phylogenetic trees from multiple sequence alignments (MSAs), as all trees in this comparison were generated using iqtree. Expanded benchmarks would necessitate the inclusion of alternative tree-building tools as well.

Benchpro's automated benchmarking scripts already offer comprehensive and reliable assessments for both taxonomic profiling and strain-resolved analysis, making them highly valuable during the development of protal. However, the current lack of documentation and the limited accessibility of strain-level benchmarks—available only through built-in functions—pose usability challenges. Improvements in accessibility and user experience would significantly enhance the tool. While R and RMarkdown are commonly used for reporting figures and statistical results, in benchpro's case, the extensive automation can lead to unnecessarily large documents, primarily due to the memory-heavy nature of embedded plots. Additionally, because all analyses are included by default, the resulting report structure tends to drive the user's exploration, rather than allowing it to be guided by specific research questions. A web-based platform with customizable plots could address these issues effectively. Such a platform would allow users to generate only the analyses relevant to their needs, compile them into concise, shareable reports, and facilitate collaboration. Moreover, it could serve not only as a tool for benchmarking newly developed methods but also as a hub for exploring benchmarking results of widely used tools.

6.3.2 Varkit

The aforementioned conceptual issues with SNP calling sensitivity and database size that became apparent during varkit's development were too serious to ignore. Protal, a taxonomic profiler which I will present in the following chapter 5, builds upon varkit's foundational idea to offer an ultra-fast metagenomic species profiler and strain-level tool. This is achieved by shifting from a purely k-mer based approach to an alignment-based approach, while still utilizing concepts like unique k-mers for increased precision. Ideas that led to the development of protal's core concepts were formed during the time developing varkit. Unfortunately, varkit's approach is too limiting regarding memory consumption and quality of strain-resolution and should not be followed-up with further developments.

6.3.3 Protal

Although the results presented are already very promising, they have also shown that alignment speed, alignment quality, taxonomic profiling, and strain-resolved analysis can still be improved. Firstly, evaluating the runtime of individual tasks in

protal has shown that seeding consumes the least time by far (see Fig. 5.3.1 C).This suggests that optimizing parameter selection for the process from seed extension to alignment has the potential to yield further performance gains. Concepts like spaced k-mers introduced in varkit could also be applied to protal to increase sensitivity. To evaluate this, the alignment sensitivity would need to be assessed with respect to a k-mer shape. As protal currently does seed finding through exact matching with a consecutive 15-mer, adding spaces into this 15-mer has the potential to increase sensitivity.

Protal's profiling and strain-level performance is dependent on accurate read alignments against GTDB's marker genes. However, some species exhibit a high amount of cross-mapping. It is commonly known that whole-genome approaches like Kraken have a high rate of false signals due to HGT. However, marker gene based approaches have been shown to be affected as well. MetaPhLAn 4 described employing an additional marker gene selection criterion to avoid short-reads cross-mapping between species [26]. Nevertheless, based on the results presented in this chapter, cross-mapping may still occur with MetaPhlAn 4, impacting the same species that pose challenges for protal.

A logical extension of protal's alignment approach is to develop a stand-alone general purpose alignment tool based on the flex-map approach and WFA2[167] as an external alignment library. Protal has demonstrated that it matches the speed of the fastest current aligner, strobealign [234], and can at least compete in alignment performance for the specific use-case of read-to-marker gene alignments.

The concept of reference-guided MSAs has already been introduced in ViralMSA [181] and VIRULIGN [153] but these are limited to whole-genome sequences as input. Protal has demonstrated that it offers a great speedup over StrainPhLAn 4 by having a custom SNP calling process followed by the reference-guided alignment of single reads. A stand-alone tool for constructing reference-guided MSAs would facilitate more efficient large-scale phylogenetic analyses of thousands of metagenomes.

6.3.4 Cross-mapping between closely related species

This thesis demonstrates that GTDB marker genes exhibit varying degrees of cross-mapping between species, affecting some species and genera more than others. This signal is responsible for a decrease in accuracy for both taxonomic profiling and strain-resolved analysis, and must be accounted for. Regardless of the potential underlying causes, technical solutions must aim to preserve the original structure of the reference database (in this case GTDB) to ensure seamless integration with existing research and tools that rely on the GTDB taxonomy. Therefore, the removal

of taxa from the reference database and/or analysis should be considered only as a last resort. However, finding a solution for this problem will increase precision, sensitivity, correctness of abundance predictions, as well as purity of MSAs due to less cross-mapping.

As a proposed solution, I recommend conducting a detailed analysis of cross-mapping on a per-gene and per-species basis. This approach will help identify and mask problematic genes for specific species during taxonomic profiling and the construction of MSAs. For genes with only small regions being affected, it is further possible to mask those regions, to retain as much information as possible. The critical aspect of this approach is to identify which areas and genes in a representative marker genome are frequently subjected to cross-mapping from other species. A similar approach was used for MetaPhlAn 4 in order to select genes that do not exhibit such cross-mapping [26]. For cases where the majority of genes is affected, species could be merged in a manner that respects the phylogenetic structure until cross-mapping falls below a previously specified threshold. Another, albeit more complex, solution involves extracting species-specific *de novo* marker genes for large species clusters using all genomes within GTDB. This approach combines the precision of species-specific marker genes with the sensitivity of universal marker genes for species with very few member genomes.

It may be the case that a universal, automated approach to taxonomic classification is inherently limited, and genus-specific considerations are essential. Some species display ambiguous boundaries when applying fixed whole-genome ANI thresholds such as 95% [13]. Even at the genus level, taxonomic boundaries can be unclear, as reflected in inconsistencies within GTDB. These issues highlight the limitations of marker gene-based approaches [278]. The selection of marker-genes already introduces a bias because other phylogenetic signals across the genome are neglected. On the other hand, efforts like GTDB successfully moved towards a unified computational approach to build their taxonomic framework, and considering clade-specific solutions contradicts this idea.

Therefore, it seems inevitable that a combination of revised species concepts (to define clearer species clusters), standardized taxonomic approaches and a better selection of marker genes would be required for the next performance jumps in metagenomic species profiling, and further strain-resolved analysis.

6.4 Outlook

The future of taxonomic profiling and strain-resolved metagenomics is undergoing a transformation, driven by advancements in sequencing technologies and evolving taxonomic standards. Long-read sequencing platforms such as Oxford Nanopore

and PacBio are revolutionizing metagenomics by enabling the assembly of complete, high-contiguity genomes directly from environmental samples. These technologies overcome limitations of short reads by resolving repetitive regions and large structural variants, which are critical for accurate strain delineation and pangenome analysis. Recent advances in long-read-only assemblers have significantly improved the quality of metagenome-assembled genomes (MAGs), with PacBio HiFi data alone now capable of producing (near-)complete circular genomes [21, 25]. Although Oxford Nanopore Technologies (ONT) still faces challenges with read accuracy, hybrid assemblies combining ONT and short-read data can also yield complete circular genomes from isolate DNA [299]. In metagenomic studies, hybrid assemblies have been shown to markedly increase contig lengths, improving the overall quality of recovered genomes [315]. Long-read sequencing further addresses two key limitations in MAG reconstruction: chimeric contigs and chimeric bins. By spanning repetitive and structurally complex regions, long reads reduce misassemblies and improve bin purity, thereby enabling the recovery of more accurate and complete MAGs [70]. As the accessibility of long-read technologies improves and sequencing costs decline, hybrid—and eventually long-read-only—approaches are expected to become standard practice in high-resolution metagenomic research. Nonetheless, long-read sequencing remains significantly more expensive than short-read methods [65]. The additional labour and coordination required when combining multiple sequencing platforms can present a further barrier for many laboratories.

Proximity ligation methods like Hi-C offer valuable complementary data for genome binning by linking DNA fragments that are physically co-located within the same cell—for example, plasmids that have been acquired via horizontal gene transfer. This spatial co-association allows for more confident assignment of contigs to individual genomes, even in complex microbial communities where strain variation, mobile genetic elements, and horizontal gene transfer can confound binning based solely on sequence composition or coverage [59, 110]. While Hi-C alone does not resolve sequence continuity, it can significantly enhance binning accuracy when used alongside long-read assemblies, which improve contiguity and resolve repetitive regions. The integration of Hi-C with long-read metagenomics thus enables more precise reconstruction of MAGs [25]. Long-read sequencing not only enhances metagenome assembly but is also increasingly proving valuable for taxonomic classification, yielding similarly or more accurate results than traditional short-read approaches [217]. The longer read lengths reduce ambiguity in mapping, helping to avoid issues such as cross-mapping between closely related taxa—an artifact observed in this thesis. However, while long reads mitigate such technical artifacts, they do not inherently resolve challenges related to the use of incomplete or low-quality MAGs as representative genomes in taxonomic databases. As a result, the broader adoption of long-read sequencing must be accompanied by efforts to expand and improve reference databases with high-quality, near-complete MAGs. These improved references

will help disentangle true biological signals, such as homologous recombination, from assembly- or binning-related inconsistencies in phylogenetic placement.

At the same time, the advancement of long-read technologies does not obviate the need for robust tools designed for short-read data. The vast majority of publicly available metagenomic datasets are generated using short-read sequencing, and this trend is likely to persist in the near term due to its cost-effectiveness and accessibility. Therefore, continued development of computational methods optimized for short-read data remains critical. High-performing tools that maximize taxonomic and functional resolution from short-read datasets will ensure that existing resources remain valuable and that ongoing studies without access to long-read sequencing can still contribute meaningfully to microbial ecology and genomics.

Bibliography

[1] Lama Izzat Hasan Abdel-Rahman and Xochitl C Morgan. 'Searching for a Consensus Among Inflammatory Bowel Disease Studies: A Systematic Meta-Analysis'. en. In: *Inflammatory Bowel Diseases* 29.1 (Jan. 2023), pp. 125–139. ISSN: 1078-0998, 1536-4844. DOI: 10.1093/ibd/izac194.

[2] Muhammad Afzaal et al. 'Human gut microbiota in health and disease: Unveiling the relationship'. In: *Frontiers in Microbiology* 13 (Sept. 2022), p. 999001. ISSN: 1664-302X. DOI: 10.3389/fmicb.2022.999001.

[3] Zahraa Al Bander et al. 'The Gut Microbiota and Inflammation: An Overview'. en. In: *International Journal of Environmental Research and Public Health* 17.20 (Oct. 2020), p. 7618. ISSN: 1660-4601. DOI: 10.3390/ijerph17207618.

[4] Alexandre Almeida et al. 'A unified catalog of 204,938 reference genomes from the human gut microbiome'. en. In: *Nature Biotechnology* 39.1 (Jan. 2021), pp. 105–114. ISSN: 1087-0156, 1546-1696. DOI: 10.1038/s41587-020-0603-3.

[5] Johannes Alneberg et al. 'Binning metagenomic contigs by coverage and composition'. en. In: *Nature Methods* 11.11 (Nov. 2014), pp. 1144–1146. ISSN: 1548-7091, 1548-7105. DOI: 10.1038/nmeth.3103.

[6] Stephen F. Altschul et al. 'Basic local alignment search tool'. en. In: *Journal of Molecular Biology* 215.3 (Oct. 1990), pp. 403–410. ISSN: 00222836. DOI: 10.1016/S0022-2836(05)80360-2.

[7] R I Amann, W Ludwig and K H Schleifer. 'Phylogenetic identification and in situ detection of individual microbial cells without cultivation'. en. In: *Microbiological Reviews* 59.1 (Mar. 1995), pp. 143–169. ISSN: 0146-0749. DOI: 10.1128/mr.59.1.143-169.1995.

[8] Sergio Andreu-Sánchez et al. 'A Benchmark of Genetic Variant Calling Pipelines Using Metagenomic Short-Read Sequencing'. In: *Frontiers in Genetics* 12 (May 2021), p. 648229. ISSN: 1664-8021. DOI: 10.3389/fgene.2021.648229.

[9] Dmitry Antipov et al. 'hybrid SPA des : an algorithm for hybrid assembly of short and long reads'. en. In: *Bioinformatics* 32.7 (Apr. 2016), pp. 1009–1015. ISSN: 1367-4811, 1367-4803. DOI: 10.1093/bioinformatics/btv688.

[10] Barbara Arbeithuber, Kateryna D. Makova and Irene Tiemann-Boege. 'Artifactual mutations resulting from DNA lesions limit detection levels in ultrasensitive sequencing applications'. en. In: *DNA Research* 23.6 (Dec. 2016), pp. 547–559. ISSN: 1340-2838, 1756-1663. DOI: 10.1093/dnares/dsw038.

[11] Manimozhiyan Arumugam et al. 'Enterotypes of the human gut microbiome'. en. In: *Nature* 473.7346 (May 2011). Publisher: Nature Publishing Group, pp. 174–180. ISSN: 1476-4687. DOI: 10.1038/nature09944.

[12] Eliran Avni and Sagi Snir. 'A New Phylogenomic Approach For Quantifying Horizontal Gene Transfer Trends in Prokaryotes'. en. In: *Scientific Reports* 10.1 (July 2020), p. 12425. ISSN: 2045-2322. DOI: 10.1038/s41598-020-62446-5.

[13] Evelise Bach et al. 'Genome-based taxonomy of Burkholderia sensu lato: Distinguishing closely related species'. eng. In: *Genetics and Molecular Biology* 46.3 Suppl 1 (2023), e20230122. ISSN: 1415-4757. DOI: 10.1590/1678-4685-GMB-2023-0122.

[14] Thomas Bäck. *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. en. Oxford University Press, Feb. 1996. ISBN: 978-0-19-509971-3 978-0-19-756092-1. DOI: 10.1093/oso/9780195099713.001.0001.

[15] Fredrik Bäckhed et al. 'The gut microbiota as an environmental factor that regulates fat storage'. en. In: *Proceedings of the National Academy of Sciences* 101.44 (Nov. 2004), pp. 15718–15723. ISSN: 0027-8424, 1091-6490. DOI: 10.1073/pnas.0407076101.

[16] Varsha D. Badal et al. 'The Gut Microbiome, Aging, and Longevity: A Systematic Review'. en. In: *Nutrients* 12.12 (Dec. 2020), p. 3759. ISSN: 2072-6643. DOI: 10.3390/nu12123759.

[17] John A. Baross and Jody W. Deming. 'Growth of 'black smoker' bacteria at temperatures of at least 250 °C'. en. In: *Nature* 303.5916 (June 1983). Publisher: Nature Publishing Group, pp. 423–426. ISSN: 1476-4687. DOI: 10.1038/303423a0.

[18] J. R. Bedarf et al. 'Das Darmmikrobiom bei der Parkinson-Krankheit'. de. In: *Der Nervenarzt* 90.2 (Feb. 2019), pp. 160–166. ISSN: 1433-0407. DOI: 10.1007/s00115-018-0601-6.

[19] Francesco Beghini et al. *Integrating taxonomic, functional, and strain-level profiling of diverse microbial communities with bioBakery 3*. en. Publisher: eLife Sciences Publications Limited. May 2021. DOI: 10.7554/eLife.65088.

[20] M. J. C. Beld and F. A. G. Reubsaet. 'Differentiation between Shigella, enteroinvasive Escherichia coli (EIEC) and noninvasive Escherichia coli'. en. In: *European Journal of Clinical Microbiology & Infectious Diseases* 31.6 (June 2012), pp. 899–904. ISSN: 0934-9723, 1435-4373. DOI: 10.1007/s10096-011-1395-7.

[21] Gaëtan Benoit et al. 'High-quality metagenome assembly from long accurate reads with metaMDBG'. en. In: *Nature Biotechnology* (Jan. 2024). ISSN: 1087-0156, 1546-1696. DOI: 10.1038/s41587-023-01983-6.

[22] David R. Bentley et al. 'Accurate whole human genome sequencing using reversible terminator chemistry'. en. In: *Nature* 456.7218 (Nov. 2008), pp. 53–59. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/nature07517.

[23] S. D. Bentley et al. 'Complete genome sequence of the model actinomycete Streptomyces coelicolor A3(2)'. en. In: *Nature* 417.6885 (May 2002), pp. 141–147. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/417141a.

[24] 'Bergey's Manual of Determinative Bacteriology:' en. In: *Academic Medicine* 9.4 (July 1934), p. 256. ISSN: 1040-2446. DOI: 10.1097/00001888-193407000-00027.

[25] Derek M. Bickhart et al. 'Generating lineage-resolved, complete metagenome-assembled genomes from complex microbial communities'. en. In: *Nature Biotechnology* 40.5 (May 2022), pp. 711–719. ISSN: 1087-0156, 1546-1696. DOI: 10.1038/s41587-021-01130-z.

[26] Aitor Blanco-Míguez et al. 'Extending and improving metagenomic taxonomic profiling with uncharacterized species using MetaPhlAn 4'. en. In: *Nature Biotechnology* (Feb. 2023). Publisher: Nature Publishing Group, pp. 1–12. ISSN: 1546-1696. DOI: 10.1038/s41587-023-01688-w.

[27] Martin J. Blaser. 'Fecal Microbiota Transplantation for Dysbiosis — Predictable Risks'. en. In: *New England Journal of Medicine* 381.21 (Nov. 2019), pp. 2064–2066. ISSN: 0028-4793, 1533-4406. DOI: 10.1056/NEJMe1913807.

[28] Sébastien Boisvert et al. 'Ray Meta: scalable de novo metagenome assembly and profiling'. en. In: *Genome Biology* 13.12 (2012), R122. ISSN: 1465-6906. DOI: 10.1186/gb-2012-13-12-r122.

[29] Camille Bonneaud et al. 'Experimental evidence for stabilizing selection on virulence in a bacterial pathogen'. en. In: *Evolution Letters* 4.6 (Dec. 2020), pp. 491–501. ISSN: 2056-3744. DOI: 10.1002/evl3.203.

[30] Régis Bonnet et al. 'Differences in rDNA libraries of faecal bacteria derived from 10- and 25-cycle PCRs.' en. In: *International Journal of Systematic and Evolutionary Microbiology* 52.3 (May 2002), pp. 757–763. ISSN: 1466-5026, 1466-5034. DOI: 10.1099/00207713-52-3-757.

[31] Leo Breiman. 'Random Forests'. In: *Machine Learning* 45.1 (2001), pp. 5–32. ISSN: 08856125. DOI: 10.1023/A:1010933404324.

[32] F. P. Breitwieser, D. N. Baker and S. L. Salzberg. 'KrakenUniq: confident and fast metagenomics classification using unique k-mer counts'. In: *Genome Biology* 19.1 (Nov. 2018), p. 198. ISSN: 1474-760X. DOI: 10.1186/s13059-018-1568-0.

[33] Florian P. Breitwieser et al. 'Human contamination in bacterial genomes has created thousands of spurious proteins'. en. In: *Genome Research* 29.6 (June 2019), pp. 954–960. ISSN: 1088-9051, 1549-5469. DOI: 10.1101/gr.245373.118.

[34] Karel Břinda, Maciej Sykulski and Gregory Kucherov. 'Spaced seeds improve k -mer-based metagenomic classification'. en. In: *Bioinformatics* 31.22 (Nov. 2015), pp. 3584–3592. ISSN: 1367-4811, 1367-4803. DOI: 10.1093/bioinformatics/btv419.

[35] Hilary P. Browne et al. 'Culturing of 'unculturable' human microbiota reveals novel taxa and extensive sporulation'. en. In: *Nature* 533.7604 (May 2016), pp. 543–546. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/nature17645.

[36] Patrick Denis Browne et al. 'GC bias affects genomic and metagenomic reconstructions, underrepresenting GC-poor organisms'. en. In: *GigaScience* 9.2 (Feb. 2020), giaa008. ISSN: 2047-217X. DOI: 10.1093/gigascience/giaa008.

[37] Benjamin Buchfink, Chao Xie and Daniel H Huson. 'Fast and sensitive protein alignment using DIAMOND'. en. In: *Nature Methods* 12.1 (Jan. 2015), pp. 59–60. ISSN: 1548-7091, 1548-7105. DOI: 10.1038/nmeth.3176.

[38] Charlie G. Buffie and Eric G. Pamer. 'Microbiota-mediated colonization resistance against intestinal pathogens'. en. In: *Nature Reviews Immunology* 13.11 (Nov. 2013), pp. 790–801. ISSN: 1474-1733, 1474-1741. DOI: 10.1038/nri3535.

[39] Benjamin J Callahan et al. 'DADA2: High-resolution sample inference from Illumina amplicon data'. en. In: *Nature Methods* 13.7 (July 2016), pp. 581–583. ISSN: 1548-7091, 1548-7105. DOI: 10.1038/nmeth.3869.

[40] J Gregory Caporaso et al. 'QIIME allows analysis of high-throughput community sequencing data'. en. In: *Nature Methods* 7.5 (May 2010), pp. 335–336. ISSN: 1548-7091, 1548-7105. DOI: 10.1038/nmeth.f.303.

[41] Roberta Caruso, Bernard C. Lo and Gabriel Núñez. 'Host–microbiota interactions in inflammatory bowel disease'. en. In: *Nature Reviews Immunology* 20.7 (July 2020), pp. 411–426. ISSN: 1474-1733, 1474-1741. DOI: 10.1038/s41577-019-0268-7.

[42] Pedro Celis, Per-Ake Larson and J. Ian Munro. 'Robin hood hashing'. In: *26th Annual Symposium on Foundations of Computer Science (sfcs 1985)*. Portland, OR, USA: IEEE, 1985, pp. 281–288. ISBN: 978-0-8186-0644-1. DOI: 10.1109/SFCS.1985.48.

[43] Pierre-Alain Chaumeil et al. 'GTDB-Tk v2: memory friendly classification with the genome taxonomy database'. In: *Bioinformatics* 38.23 (Dec. 2022), pp. 5315–5316. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btac672.

[44] Alex Chklovski et al. 'CheckM2: a rapid, scalable and accurate tool for assessing microbial genome quality using machine learning'. en. In: *Nature Methods* 20.8 (Aug. 2023), pp. 1203–1212. ISSN: 1548-7091, 1548-7105. DOI: 10.1038/s41592-023-01940-w.

[45] Hani Choudhry. 'The Microbiome and Its Implications in Cancer Immunotherapy'. en. In: *Molecules* 26.1 (Jan. 2021), p. 206. ISSN: 1420-3049. DOI: 10.3390/molecules26010206.

[46] Brianna Chrisman et al. 'The human "contaminome": bacterial, viral, and computational contamination in whole genome sequences from 1000 families'. en. In: *Scientific Reports* 12.1 (June 2022), p. 9863. ISSN: 2045-2322. DOI: 10.1038/s41598-022-13269-z.

[47] Liam Chung et al. 'Bacteroides fragilis Toxin Coordinates a Pro-carcinogenic Inflammatory Cascade via Targeting of Colonic Epithelial Cells'. en. In: *Cell Host & Microbe* 23.2 (Feb. 2018), 203–214.e5. ISSN: 19313128. DOI: 10.1016/j.chom.2018.01.007.

[48] Maria Chuvochina et al. 'The importance of designating type material for uncultured taxa'. en. In: *Systematic and Applied Microbiology* 42.1 (Jan. 2019), pp. 15–21. ISSN: 07232020. DOI: 10.1016/j.syapm.2018.07.003.

[49] Francesca D. Ciccarelli et al. 'Toward Automatic Reconstruction of a Highly Resolved Tree of Life'. en. In: *Science* 311.5765 (Mar. 2006), pp. 1283–1287. ISSN: 0036-8075, 1095-9203. DOI: 10.1126/science.1123061.

[50] Stacy Ciufo et al. 'Using average nucleotide identity to improve taxonomic assignments in prokaryotic genomes at the NCBI'. en. In: *International Journal of Systematic and Evolutionary Microbiology* 68.7 (July 2018), pp. 2386–2392. ISSN: 1466-5026, 1466-5034. DOI: 10.1099/ijsem.0.002809.

[51] Marcus J. Claesson et al. 'Gut microbiota composition correlates with diet and health in the elderly'. en. In: *Nature* 488.7410 (Aug. 2012), pp. 178–184. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/nature11319.

[52] Karen Clark et al. 'GenBank'. en. In: *Nucleic Acids Research* 44.D1 (Jan. 2016), pp. D67–D72. ISSN: 0305-1048, 1362-4962. DOI: 10.1093/nar/gkv1276.

[53] Mark M. Collery et al. 'What's a SNP between friends: The influence of single nucleotide polymorphisms on virulence and phenotypes of *Clostridium difficile* strain 630 and derivatives'. en. In: *Virulence* 8.6 (Aug. 2017), pp. 767–781. ISSN: 2150-5594, 2150-5608. DOI: 10.1080/21505594.2016.1237333.

[54] R. R. Colwell. 'Polyphasic Taxonomy of the Genus *Vibrio:* Numerical Taxonomy of *Vibrio cholerae, Vibrio parahaemolyticus* , and Related *Vibrio* Species'. en. In: *Journal of Bacteriology* 104.1 (Oct. 1970), pp. 410–433. ISSN: 0021-9193, 1098-5530. DOI: 10.1128/jb.104.1.410-433.1970.

[55] Paul Igor Costea et al. 'metaSNV: A tool for metagenomic strain level analysis'. en. In: *PLOS ONE* 12.7 (July 2017). Publisher: Public Library of Science, e0182392. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0182392.

[56] Arianna K. DeGruttola et al. 'Current Understanding of Dysbiosis in Disease in Human and Animal Models:' en. In: *Inflammatory Bowel Diseases* 22.5 (May 2016), pp. 1137–1150. ISSN: 1078-0998. DOI: 10.1097/MIB.0000000000000750.

[57] T. Z. DeSantis et al. 'Greengenes, a Chimera-Checked 16S rRNA Gene Database and Workbench Compatible with ARB'. en. In: *Applied and Environmental Microbiology* 72.7 (July 2006), pp. 5069–5072. ISSN: 0099-2240, 1098-5336. DOI: 10.1128/AEM.03006-05.

[58] Hongdo Do and Alexander Dobrovic. 'Sequence Artifacts in DNA from Formalin-Fixed Tissues: Causes and Strategies for Minimization'. en. In: *Clinical Chemistry* 61.1 (Jan. 2015), pp. 64–71. ISSN: 0009-9147, 1530-8561. DOI: 10.1373/clinchem.2014.223040.

[59] Yuxuan Du and Fengzhu Sun. 'HiCBin: binning metagenomic contigs and recovering metagenome-assembled genomes using Hi-C contact maps'. en. In: *Genome Biology* 23.1 (Feb. 2022), p. 63. ISSN: 1474-760X. DOI: 10.1186/s13059-022-02626-w.

[60] David J. Durgan et al. 'Role of the Gut Microbiome in Obstructive Sleep Apnea–Induced Hypertension'. en. In: *Hypertension* 67.2 (Feb. 2016), pp. 469–474. ISSN: 0194-911X, 1524-4563. DOI: 10.1161/HYPERTENSIONAHA.115.06672.

[61] Robert Edgar. 'Syncmers are more sensitive than minimizers for selecting conserved k -mers in biological sequences'. en. In: *PeerJ* 9 (Feb. 2021), e10805. ISSN: 2167-8359. DOI: 10.7717/peerj.10805.

[62] Robert C Edgar. 'UPARSE: highly accurate OTU sequences from microbial amplicon reads'. en. In: *Nature Methods* 10.10 (Oct. 2013), pp. 996–998. ISSN: 1548-7091, 1548-7105. DOI: 10.1038/nmeth.2604.

[63] S. Dusko Ehrlich. 'MetaHIT: The European Union Project on Metagenomics of the Human Intestinal Tract'. en. In: *Metagenomics of the Human Body*. Ed. by Karen E. Nelson. New York, NY: Springer, 2011, pp. 307–316. ISBN: 978-1-4419-7089-3. DOI: 10.1007/978-1-4419-7089-3_15.

[64] John Eid et al. 'Real-Time DNA Sequencing from Single Polymerase Molecules'. en. In: *Science* 323.5910 (Jan. 2009), pp. 133–138. ISSN: 0036-8075, 1095-9203. DOI: 10.1126/science.1162986.

[65] Raphael Eisenhofer et al. 'A comparison of short-read, HiFi long-read, and hybrid strategies for genome-resolved metagenomics'. eng. In: *Microbiology Spectrum* 12.4 (Apr. 2024), e0359023. ISSN: 2165-0497. DOI: 10.1128/spectrum.03590-23.

[66] Jeremiah J. Faith et al. 'The Long-Term Stability of the Human Gut Microbiota'. en. In: *Science* 341.6141 (July 2013), p. 1237439. ISSN: 0036-8075, 1095-9203. DOI: 10.1126/science.1237439.

[67] Gwen Falony et al. 'Population-level analysis of gut microbiome variation'. en. In: *Science* 352.6285 (Apr. 2016), pp. 560–564. ISSN: 0036-8075, 1095-9203. DOI: 10.1126/science.aad3503.

[68] Karoline Faust and Jeroen Raes. 'Microbial interactions: from networks to models'. en. In: *Nature Reviews Microbiology* 10.8 (Aug. 2012), pp. 538–550. ISSN: 1740-1526, 1740-1534. DOI: 10.1038/nrmicro2832.

[69] Scott Federhen. 'The NCBI Taxonomy database'. In: *Nucleic Acids Research* 40.D1 (Jan. 2012), pp. D136–D143. ISSN: 0305-1048. DOI: 10.1093/nar/gkr1178.

[70] Xiaowen Feng et al. 'Metagenome assembly of high-fidelity long reads with hifiasm-meta'. en. In: *Nature Methods* 19.6 (June 2022), pp. 671–674. ISSN: 1548-7091, 1548-7105. DOI: 10.1038/s41592-022-01478-3.

[71] P. Ferragina and G. Manzini. 'Opportunistic data structures with applications'. In: *Proceedings 41st Annual Symposium on Foundations of Computer Science*. Redondo Beach, CA, USA: IEEE Comput. Soc, 2000, pp. 390–398. ISBN: 978-0-7695-0850-4. DOI: 10.1109/SFCS.2000.892127.

[72] Klas Flärdh and Mark J. Buttner. 'Streptomyces morphogenetics: dissecting differentiation in a filamentous bacterium'. en. In: *Nature Reviews Microbiology* 7.1 (Jan. 2009), pp. 36–49. ISSN: 1740-1526, 1740-1534. DOI: 10.1038/nrmicro1968.

[73] Alexander Fleming. 'THE DISCOVERY OF PENICILLIN'. en. In: *British Medical Bulletin* 2.1 (1944), pp. 4–5. ISSN: 1471-8391, 0007-1420. DOI: 10.1093/oxfordjournals.bmb.a071032.

[74] Harry J. Flint et al. 'The role of the gut microbiota in nutrition and health'. en. In: *Nature Reviews Gastroenterology & Hepatology* 9.10 (Oct. 2012), pp. 577–589. ISSN: 1759-5045, 1759-5053. DOI: 10.1038/nrgastro.2012.156.

[75] Kristoffer Forslund et al. 'Disentangling type 2 diabetes and metformin treatment signatures in the human gut microbiota'. en. In: *Nature* 528.7581 (Dec. 2015). Number: 7581 Publisher: Nature Publishing Group, pp. 262–266. ISSN: 1476-4687. DOI: 10.1038/nature15766.

[76] Clémence Frioux et al. 'Enterosignatures define common bacterial guilds in the human gut microbiome'. en. In: *Cell Host & Microbe* 31.7 (July 2023), 1111–1125.e6. ISSN: 19313128. DOI: 10.1016/j.chom.2023.05.024.

[77] Adrian Fritz et al. 'CAMISIM: simulating metagenomes and microbial communities'. en. In: *Microbiome* 7.1 (Dec. 2019), p. 17. ISSN: 2049-2618. DOI: 10.1186/s40168-019-0633-6.

[78] Limin Fu et al. 'CD-HIT: accelerated for clustering the next-generation sequencing data'. en. In: *Bioinformatics* 28.23 (Dec. 2012), pp. 3150–3152. ISSN: 1367-4803, 1367-4811. DOI: 10.1093/bioinformatics/bts565.

[79] Asami Fukuda et al. 'DDBJ update: streamlining submission and access of human data'. eng. In: *Nucleic Acids Research* 49.D1 (Jan. 2021), pp. D71–D75. ISSN: 1362-4962. DOI: 10.1093/nar/gkaa982.

[80] Erik Garrison and Gabor Marth. *Haplotype-based variant detection from short-read sequencing*. Version Number: 2. 2012. DOI: 10.48550/ARXIV.1207.3907.

[81] Nandita R. Garud et al. 'Evolutionary dynamics of bacteria in the gut microbiome within and across hosts'. en. In: *PLOS Biology* 17.1 (Jan. 2019). Ed. by Isabel Gordo, e3000102. ISSN: 1545-7885. DOI: 10.1371/journal.pbio.3000102.

[82] Abraham Gihawi et al. 'Major data analysis errors invalidate cancer microbiome findings'. en. In: *mBio* 14.5 (Oct. 2023). Ed. by Igor B. Zhulin, e01607–23. ISSN: 2150-7511. DOI: 10.1128/mbio.01607-23.

[83] David M. Golombos et al. 'The Role of Gut Microbiome in the Pathogenesis of Prostate Cancer: A Prospective, Pilot Study'. en. In: *Urology* 111 (Jan. 2018), pp. 122–128. ISSN: 00904295. DOI: 10.1016/j.urology.2017.08.039.

[84] Pedro González-Torres et al. 'Impact of Homologous Recombination on the Evolution of Prokaryotic Core Genomes'. en. In: *mBio* 10.1 (Feb. 2019). Ed. by Joseph Heitman, e02494–18. ISSN: 2161-2129, 2150-7511. DOI: 10.1128/mBio.02494-18.

[85] Julia K. Goodrich et al. 'Human Genetics Shape the Gut Microbiome'. en. In: *Cell* 159.4 (Nov. 2014), pp. 789–799. ISSN: 00928674. DOI: 10.1016/j.cell.2014.09.053.

[86] Johan Goris et al. 'DNA–DNA hybridization values and their relationship to whole-genome sequence similarities'. en. In: *International Journal of Systematic and Evolutionary Microbiology* 57.1 (Jan. 2007), pp. 81–91. ISSN: 1466-5026, 1466-5034. DOI: 10.1099/ijs.0.64483-0.

[87] Daniel B. Graham and Ramnik J. Xavier. 'Conditioning of the immune system by the microbiome'. en. In: *Trends in Immunology* 44.7 (July 2023), pp. 499–511. ISSN: 14714906. DOI: 10.1016/j.it.2023.05.002.

[88] Alastair Grant et al. *Improved taxonomic annotation of Archaea communities using LotuS2, the Genome Taxonomy Database and RNAseq data*. en. Aug. 2023. DOI: 10.1101/2023.08.21.554127.

[89] Catherine Grasso et al. 'Assessing Copy Number Alterations in Targeted, Amplicon-Based Next-Generation Sequencing Data'. en. In: *The Journal of Molecular Diagnostics* 17.1 (Jan. 2015), pp. 53–63. ISSN: 15251578. DOI: 10.1016/j.jmoldx.2014.09.008.

[90] Manoj Gurung et al. 'Role of gut microbiota in type 2 diabetes pathophysiology'. English. In: *eBioMedicine* 51 (Jan. 2020). Publisher: Elsevier. ISSN: 2352-3964. DOI: 10.1016/j.ebiom.2019.11.051.

[91] Barry G. Hall, Garth D. Ehrlich and Fen Z. Hu. 'Pan-genome analysis provides much higher strain typing resolution than multi-locus sequence typing'. en. In: *Microbiology* 156.4 (Apr. 2010), pp. 1060–1068. ISSN: 1350-0872, 1465-2080. DOI: 10.1099/mic.0.035188-0.

[92] Benjamin D. Hall and S. Spiegelman. 'SEQUENCE COMPLEMENTARITY OF T2-DNA AND T2-SPECIFIC RNA'. en. In: *Proceedings of the National Academy of Sciences* 47.2 (Feb. 1961), pp. 137–146. ISSN: 0027-8424, 1091-6490. DOI: 10.1073/pnas.47.2.137.

[93] Jo Handelsman et al. 'Molecular biological access to the chemistry of unknown soil microbes: a new frontier for natural products'. en. In: *Chemistry & Biology* 5.10 (Oct. 1998), R245–R249. ISSN: 10745521. DOI: 10.1016/S1074-5521(98)90108-9.

[94] Brian P. Hedlund et al. 'SeqCode: a nomenclatural code for prokaryotes described from sequence data'. en. In: *Nature Microbiology* (Sept. 2022). ISSN: 2058-5276. DOI: 10.1038/s41564-022-01214-9.

[95] Falk Hildebrand. 'Ultra-resolution Metagenomics: When Enough Is Not Enough'. en. In: *mSystems* 6.4 (Aug. 2021), e00881–21. ISSN: 2379-5077. DOI: 10.1128/mSystems.00881-21.

[96] Falk Hildebrand et al. 'LotuS: an efficient and user-friendly OTU processing pipeline'. en. In: *Microbiome* 2.1 (Dec. 2014), p. 30. ISSN: 2049-2618. DOI: 10.1186/2049-2618-2-30.

[97] Falk Hildebrand et al. 'Antibiotics-induced monodominance of a novel gut bacterial order'. eng. In: *Gut* 68.10 (Oct. 2019), pp. 1781–1790. ISSN: 1468-3288. DOI: 10.1136/gutjnl-2018-317715.

[98] Falk Hildebrand et al. 'Dispersal strategies shape persistence and evolution of human gut bacteria'. en. In: *Cell Host & Microbe* 29.7 (July 2021), 1167–1176.e9. ISSN: 19313128. DOI: 10.1016/j.chom.2021.05.008.

[99] Pranvera Hiseni et al. 'HumGut: a comprehensive human gut prokaryotic genomes collection filtered by metagenome data'. In: *Microbiome* 9.1 (July 2021), p. 165. ISSN: 2049-2618. DOI: 10.1186/s40168-021-01114-w.

[100] Ian Holmes, Keith Harris and Christopher Quince. 'Dirichlet Multinomial Mixtures: Generative Models for Microbial Metagenomics'. en. In: *PLoS ONE* 7.2 (Feb. 2012). Ed. by Jack Anthony Gilbert, e30126. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0030126.

[101] L. V. Hooper and J. I. Gordon. 'Commensal host-bacterial relationships in the gut'. eng. In: *Science (New York, N.Y.)* 292.5519 (May 2001), pp. 1115–1118. ISSN: 0036-8075. DOI: 10.1126/science.1058709.

[102] Weichun Huang et al. 'ART: a next-generation sequencing read simulator'. en. In: *Bioinformatics* 28.4 (Feb. 2012), pp. 593–594. ISSN: 1367-4811, 1367-4803. DOI: 10.1093/bioinformatics/btr708.

[103] Philip Hugenholtz et al. 'Prokaryotic taxonomy and nomenclature in the age of big sequence data'. en. In: *The ISME Journal* 15.7 (July 2021), pp. 1879–1892. ISSN: 1751-7362, 1751-7370. DOI: 10.1038/s41396-021-00941-x.

[104] Susan M Huse et al. 'Accuracy and quality of massively parallel DNA pyrosequencing'. en. In: *Genome Biology* 8.7 (July 2007), R143. ISSN: 1474-760X. DOI: 10.1186/gb-2007-8-7-r143.

[105] Daniel H. Huson et al. 'MEGAN analysis of metagenomic data'. en. In: *Genome Research* 17.3 (Mar. 2007), pp. 377–386. ISSN: 1088-9051. DOI: 10.1101/gr.5969107.

[106] Curtis Huttenhower et al. 'Structure, function and diversity of the healthy human microbiome'. en. In: *Nature* 486.7402 (June 2012). Publisher: Nature Publishing Group, pp. 207–214. ISSN: 1476-4687. DOI: 10.1038/nature11234.

[107] Cláudia Maria Dos Santos Pereira Indiani et al. 'Childhood Obesity and Firmicutes/Bacteroidetes Ratio in the Gut Microbiota: A Systematic Review'. en. In: *Childhood Obesity* 14.8 (Dec. 2018), pp. 501–509. ISSN: 2153-2168, 2153-2176. DOI: 10.1089/chi.2018.0040.

[108] 'International Code of Nomenclature of Prokaryotes: Prokaryotic Code (2008 Revision)'. en. In: *International Journal of Systematic and Evolutionary Microbiology* 69.1A (Jan. 2019), S1–S111. ISSN: 1466-5026, 1466-5034. DOI: 10.1099/ijsem.0.000778.

[109] Ivaylo I. Ivanov et al. 'Induction of Intestinal Th17 Cells by Segmented Filamentous Bacteria'. en. In: *Cell* 139.3 (Oct. 2009), pp. 485–498. ISSN: 00928674. DOI: 10.1016/j.cell.2009.09.033.

[110] Valeriia Ivanova et al. 'Hi-C Metagenomics in the ICU: Exploring Clinically Relevant Features of Gut Microbiome in Chronically Critically Ill Patients'. eng. In: *Frontiers in Microbiology* 12 (2021), p. 770323. ISSN: 1664-302X. DOI: 10.3389/fmicb.2021.770323.

[111] Chirag Jain et al. 'High throughput ANI analysis of 90K prokaryotic genomes reveals clear species boundaries'. en. In: *Nature Communications* 9.1 (Nov. 2018), p. 5114. ISSN: 2041-1723. DOI: 10.1038/s41467-018-07641-9.

[112] Lei Jiang and Farzaneh Zokaee. 'EXMA: A Genomics Accelerator for Exact-Matching'. In: *2021 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*. Seoul, Korea (South): IEEE, Feb. 2021, pp. 399–411. ISBN: 978-1-66542-235-2. DOI: 10.1109/HPCA51647.2021.00041.

[113] Jethro S. Johnson et al. 'Evaluation of 16S rRNA gene sequencing for species and strain-level microbiome analysis'. en. In: *Nature Communications* 10.1 (Nov. 2019). Publisher: Nature Publishing Group, p. 5029. ISSN: 2041-1723. DOI: 10.1038/s41467-019-13036-1.

[114] Raphaela Joos et al. 'Examining the healthy human microbiome concept'. en. In: *Nature Reviews Microbiology* 23.3 (Mar. 2025), pp. 192–205. ISSN: 1740-1526, 1740-1534. DOI: 10.1038/s41579-024-01107-0.

[115] Dongwan D. Kang et al. 'MetaBAT 2: an adaptive binning algorithm for robust and efficient genome reconstruction from metagenome assemblies'. en. In: *PeerJ* 7 (July 2019). Publisher: PeerJ Inc., e7359. ISSN: 2167-8359. DOI: 10.7717/peerj.7359.

[116] Gabriela Kapinusova, Marco A. Lopez Marin and Ondrej Uhlik. 'Reaching unreachables: Obstacles and successes of microbial cultivation and their reasons'. In: *Frontiers in Microbiology* 14 (Mar. 2023), p. 1089630. ISSN: 1664-302X. DOI: 10.3389/fmicb.2023.1089630.

[117] Samuel Kariin and Chris Burge. 'Dinucleotide relative abundance extremes: a genomic signature'. en. In: *Trends in Genetics* 11.7 (July 1995), pp. 283–290. ISSN: 01689525. DOI: 10.1016/S0168-9525(00)89076-9.

[118] K. Katoh. 'MAFFT: a novel method for rapid multiple sequence alignment based on fast Fourier transform'. In: *Nucleic Acids Research* 30.14 (July 2002), pp. 3059–3066. ISSN: 13624962. DOI: 10.1093/nar/gkf436.

[119] Asma Kazemi et al. 'Effect of probiotic and prebiotic vs placebo on psychological outcomes in patients with major depressive disorder: A randomized clinical trial'. en. In: *Clinical Nutrition* 38.2 (Apr. 2019), pp. 522–528. ISSN: 02615614. DOI: 10.1016/j.clnu.2018.04.010.

[120] Marisa Isabell Keller et al. *Refined Enterotyping Reveals Dysbiosis in Global Fecal Metagenomes*. en. Aug. 2024. DOI: 10.1101/2024.08.13.607711.

[121] Israr Khan et al. 'Alteration of Gut Microbiota in Inflammatory Bowel Disease (IBD): Cause or Consequence? IBD Treatment Targeting the Gut Microbiome'. In: *Pathogens* 8.3 (Aug. 2019), p. 126. ISSN: 2076-0817. DOI: 10.3390/pathogens8030126.

[122] Bryce Kille et al. *Minmers are a generalization of minimizers that enable unbiased local Jaccard estimation*. en. May 2023. DOI: 10.1101/2023.05.16.540882.

[123] Daehwan Kim et al. 'Centrifuge: rapid and sensitive classification of metagenomic sequences'. en. In: *Genome Research* 26.12 (Dec. 2016), pp. 1721–1729. ISSN: 1088-9051, 1549-5469. DOI: 10.1101/gr.210641.116.

[124] Nayeon Kim et al. 'Genome-resolved metagenomics: a game changer for microbiome medicine'. en. In: *Experimental & Molecular Medicine* 56.7 (July 2024), pp. 1501–1512. ISSN: 2092-6413. DOI: 10.1038/s12276-024-01262-7.

[125] Martin Kircher, Susanna Sawyer and Matthias Meyer. 'Double indexing overcomes inaccuracies in multiplex sequencing on the Illumina platform'. en. In: *Nucleic Acids Research* 40.1 (Jan. 2012), e3–e3. ISSN: 1362-4962, 0305-1048. DOI: 10.1093/nar/gkr771.

[126] Ellen Knierim et al. 'Systematic Comparison of Three Methods for Fragmentation of Long-Range PCR Products for Next Generation Sequencing'. en. In: *PLoS ONE* 6.11 (Nov. 2011). Ed. by M. Thomas P. Gilbert, e28240. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0028240.

[127] Mikhail Kolmogorov et al. 'Assembly of long, error-prone reads using repeat graphs'. en. In: *Nature Biotechnology* 37.5 (May 2019), pp. 540–546. ISSN: 1087-0156, 1546-1696. DOI: 10.1038/s41587-019-0072-8.

[128] Sergey Koren et al. 'Canu: scalable and accurate long-read assembly via adaptive k-mer weighting and repeat separation'. eng. In: *Genome Research* 27.5 (May 2017), pp. 722–736. ISSN: 1549-5469. DOI: 10.1101/gr.215087.116.

[129] Hans-Peter Kriegel, Erich Schubert and Arthur Zimek. 'The (black) art of runtime evaluation: Are we comparing algorithms or implementations?' en. In: *Knowledge and Information Systems* 52.2 (Aug. 2017), pp. 341–378. ISSN: 0219-1377, 0219-3116. DOI: 10.1007/s10115-016-1004-2.

[130] Mary K. Kuhner and Jon Yamato. 'Practical Performance of Tree Comparison Metrics'. en. In: *Systematic Biology* 64.2 (Mar. 2015), pp. 205–214. ISSN: 1076-836X, 1063-5157. DOI: 10.1093/sysbio/syu085.

[131] J.-C. Lagier et al. 'Microbial culturomics: paradigm shift in the human gut microbiome study'. en. In: *Clinical Microbiology and Infection* 18.12 (Dec. 2012), pp. 1185–1193. ISSN: 1198743X. DOI: 10.1111/1469-0691.12023.

[132] Anthony LaMarca and Richard E Ladner. 'The Influence of Caches on the Performance of Sorting'. en. In: *Journal of Algorithms* 31.1 (Apr. 1999), pp. 66–104. ISSN: 01966774. DOI: 10.1006/jagm.1998.0985.

[133] Miriam Land et al. 'Insights from 20 years of bacterial genome sequencing'. eng. In: *Functional & Integrative Genomics* 15.2 (Mar. 2015), pp. 141–161. ISSN: 1438-7948. DOI: 10.1007/s10142-015-0433-4.

[134] Eric S. Lander and Michael S. Waterman. 'Genomic mapping by fingerprinting random clones: A mathematical analysis'. en. In: *Genomics* 2.3 (Apr. 1988), pp. 231–239. ISSN: 0888-7543. DOI: 10.1016/0888-7543(88)90007-9.

[135] Nick Lane. 'The unseen world: reflections on Leeuwenhoek (1677) 'Concerning little animals''. en. In: *Philosophical Transactions of the Royal Society B: Biological Sciences* 370.1666 (Apr. 2015), p. 20140344. ISSN: 0962-8436, 1471-2970. DOI: 10.1098/rstb.2014.0344.

[136] Ben Langmead and Steven L. Salzberg. 'Fast gapped-read alignment with Bowtie 2'. en. In: *Nature Methods* 9.4 (Apr. 2012). Number: 4 Publisher: Nature Publishing Group, pp. 357–359. ISSN: 1548-7105. DOI: 10.1038/nmeth.1923.

[137] Ben Langmead et al. 'Ultrafast and memory-efficient alignment of short DNA sequences to the human genome'. en. In: *Genome Biology* 10.3 (2009), R25. ISSN: 1465-6906. DOI: 10.1186/gb-2009-10-3-r25.

[138] S. P. Lapage et al., eds. *International Code of Nomenclature of Bacteria: Bacteriological Code, 1990 Revision*. eng. Washington (DC): ASM Press, 1992. ISBN: 978-1-55581-039-9.

[139] Aonghus Lavelle and Harry Sokol. 'Beyond metagenomics, metatranscriptomics illuminates microbiome functionality in IBD'. en. In: *Nature Reviews Gastroenterology & Hepatology* 15.4 (Apr. 2018). Number: 4 Publisher: Nature Publishing Group, pp. 193–194. ISSN: 1759-5053. DOI: 10.1038/nrgastro.2018.15.

[140] Jeffrey G. Lawrence and Adam C. Retchless. 'The Interplay of Homologous Recombination and Horizontal Gene Transfer in Bacterial Speciation'. In: *Horizontal Gene Transfer*. Ed. by John M. Walker et al. Vol. 532. Series Title: Methods in Molecular Biology. Totowa, NJ: Humana Press, 2009, pp. 29–53.

ISBN: 978-1-60327-852-2 978-1-60327-853-9. DOI: 10.1007/978-1-60327-853-9_3.

[141] Heewook Lee et al. 'Rate and molecular spectrum of spontaneous mutations in the bacterium Escherichia coli as determined by whole-genome sequencing'. eng. In: *Proceedings of the National Academy of Sciences of the United States of America* 109.41 (Oct. 2012), E2774–2783. ISSN: 1091-6490. DOI: 10.1073/pnas.1210309109.

[142] Andreas Leimbach, Jörg Hacker and Ulrich Dobrindt. 'E. coli as an all-rounder: the thin line between commensalism and pathogenicity'. eng. In: *Current Topics in Microbiology and Immunology* 358 (2013), pp. 3–32. ISSN: 0070-217X. DOI: 10.1007/82_2012_303.

[143] Dinghua Li et al. 'MEGAHIT: an ultra-fast single-node solution for large and complex metagenomics assembly via succinct *de Bruijn* graph'. en. In: *Bioinformatics* 31.10 (May 2015), pp. 1674–1676. ISSN: 1367-4811, 1367-4803. DOI: 10.1093/bioinformatics/btv033.

[144] Heng Li. 'A statistical framework for SNP calling, mutation discovery, association mapping and population genetical parameter estimation from sequencing data'. en. In: *Bioinformatics* 27.21 (Nov. 2011), pp. 2987–2993. ISSN: 1367-4811, 1367-4803. DOI: 10.1093/bioinformatics/btr509.

[145] Heng Li. 'Minimap2: pairwise alignment for nucleotide sequences'. In: *Bioinformatics* 34.18 (Sept. 2018), pp. 3094–3100. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/bty191.

[146] Heng Li and Richard Durbin. 'Fast and accurate short read alignment with Burrows–Wheeler transform'. In: *Bioinformatics* 25.14 (July 2009), pp. 1754–1760. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btp324.

[147] Heng Li et al. 'The Sequence Alignment/Map format and SAMtools'. eng. In: *Bioinformatics (Oxford, England)* 25.16 (Aug. 2009), pp. 2078–2079. ISSN: 1367-4811. DOI: 10.1093/bioinformatics/btp352.

[148] Wenjun Li et al. 'RefSeq: expanding the Prokaryotic Genome Annotation Pipeline reach with protein family model curation'. en. In: *Nucleic Acids Research* 49.D1 (Jan. 2021), pp. D1020–D1028. ISSN: 0305-1048, 1362-4962. DOI: 10.1093/nar/gkaa1105.

[149] Yanbo Li, Hardip Patel and Yu Lin. 'Kmer2SNP: Reference-Free Heterozygous SNP Calling Using k-mer Frequency Distributions'. en. In: *Variant Calling*. Ed. by Charlotte Ng and Salvatore Piscuoglio. Vol. 2493. Series Title: Methods in Molecular Biology. New York, NY: Springer US, 2022, pp. 257–265. ISBN: 978-1-07-162292-6 978-1-07-162293-3. DOI: 10.1007/978-1-0716-2293-3_16.

[150] Herui Liao, Yongxin Ji and Yanni Sun. 'High-resolution strain-level microbiome composition analysis from short reads'. en. In: *Microbiome* 11.1 (Aug. 2023), p. 183. ISSN: 2049-2618. DOI: 10.1186/s40168-023-01615-w.

[151] Yi-Lun Liao et al. 'Adaptively Banded Smith-Waterman Algorithm for Long Reads and Its Hardware Accelerator'. In: *2018 IEEE 29th International Conference on Application-specific Systems, Architectures and Processors (ASAP)*. Milan: IEEE, July 2018, pp. 1–9. ISBN: 978-1-5386-7479-6. DOI: 10.1109/ASAP.2018.8445105.

[152] Andy Liaw and Matthew Wiener. 'Classification and Regression by randomForest'. In: *R News* 2.3 (2002), pp. 18–22.

[153] Pieter J K Libin et al. 'VIRULIGN: fast codon-correct alignment and annotation of viral genomes'. en. In: *Bioinformatics* 35.10 (May 2019). Ed. by John Hancock, pp. 1763–1765. ISSN: 1367-4803, 1367-4811. DOI: 10.1093/bioinformatics/bty851.

[154] Jörg Linde et al. 'Comparison of Illumina and Oxford Nanopore Technology for genome analysis of Francisella tularensis, Bacillus anthracis, and Brucella suis'. en. In: *BMC Genomics* 24.1 (May 2023), p. 258. ISSN: 1471-2164. DOI: 10.1186/s12864-023-09343-z.

[155] C Linnaeus. *Systema naturae per regna tria naturae, secundum classes, ordines, genera, species, cum characteribus, differentiis, synonymis, locis. Tomus I. Editio decima, reformata*. Holmiae [= Stockholm]: L. Salvii, 1758.

[156] Bing-Nan Liu et al. 'Gut microbiota in obesity'. In: *World Journal of Gastroenterology* 27.25 (July 2021), pp. 3837–3850. ISSN: 1007-9327. DOI: 10.3748/wjg.v27.i25.3837.

[157] Jason Lloyd-Price et al. 'Strains, functions and dynamics in the expanded Human Microbiome Project'. en. In: *Nature* 550.7674 (Oct. 2017), pp. 61–66. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/nature23889.

[158] Petra Louis and Harry J. Flint. 'Diversity, metabolism and microbial ecology of butyrate-producing bacteria from the human large intestine'. en. In: *FEMS Microbiology Letters* 294.1 (May 2009), pp. 1–8. ISSN: 03781097, 15746968. DOI: 10.1111/j.1574-6968.2009.01514.x.

[159] Jennifer Lu et al. 'Bracken: estimating species abundance in metagenomics data'. en. In: *PeerJ Computer Science* 3 (Jan. 2017), e104. ISSN: 2376-5992. DOI: 10.7717/peerj-cs.104.

[160] Chengwei Luo et al. 'ConStrains identifies microbial strains in metagenomic datasets'. en. In: *Nature Biotechnology* 33.10 (Oct. 2015), pp. 1045–1052. ISSN: 1087-0156, 1546-1696. DOI: 10.1038/nbt.3319.

[161] J Felsenstein M K Kuhner. 'A simulation comparison of phylogeny algorithms under equal and unequal evolutionary rates.' en. In: *Molecular Biology and Evolution* (May 1994). ISSN: 1537-1719. DOI: 10.1093/oxfordjournals.molbev.a040126.

[162] Xiaotu Ma et al. 'Analysis of error profiles in deep next-generation sequencing data'. en. In: *Genome Biology* 20.1 (Dec. 2019), p. 50. ISSN: 1474-760X. DOI: 10.1186/s13059-019-1659-6.

[163] Yongshun Ma et al. 'Metagenome Analysis of Intestinal Bacteria in Healthy People, Patients With Inflammatory Bowel Disease and Colorectal Cancer'. In: *Frontiers in Cellular and Infection Microbiology* 11 (Feb. 2021), p. 599734. ISSN: 2235-2988. DOI: 10.3389/fcimb.2021.599734.

[164] Oleksandr M Maistrenko et al. 'Disentangling the impact of environmental and phylogenetic constraints on prokaryotic within-species diversity'. en. In: *The ISME Journal* 14.5 (May 2020), pp. 1247–1259. ISSN: 1751-7362, 1751-7370. DOI: 10.1038/s41396-020-0600-z.

[165] Chaysavanh Manichanh et al. 'The gut microbiota in IBD'. eng. In: *Nature Reviews. Gastroenterology & Hepatology* 9.10 (Oct. 2012), pp. 599–608. ISSN: 1759-5053. DOI: 10.1038/nrgastro.2012.152.

[166] Ohad Manor et al. 'Health and disease markers correlate with gut microbiome composition across thousands of people'. en. In: *Nature Communications* 11.1 (Oct. 2020), p. 5206. ISSN: 2041-1723. DOI: 10.1038/s41467-020-18871-1.

[167] Santiago Marco-Sola et al. 'Fast gap-affine pairwise alignment using the wavefront algorithm'. In: *Bioinformatics* 37.4 (Feb. 2021), pp. 456–463. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btaa777.

[168] Marcel Margulies et al. 'Genome sequencing in microfabricated high-density picolitre reactors'. en. In: *Nature* 437.7057 (Sept. 2005), pp. 376–380. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/nature03959.

[169] Vasimuddin Md et al. *Efficient Architecture-Aware Acceleration of BWA-MEM for Multicore Systems*. arXiv:1907.12931 [cs, q-bio]. July 2019. DOI: 10.48550/arXiv.1907.12931.

[170] Raaj S. Mehta et al. 'Association of Dietary Patterns With Risk of Colorectal Cancer Subtypes Classified by *Fusobacterium nucleatum* in Tumor Tissue'. en. In: *JAMA Oncology* 3.7 (July 2017), p. 921. ISSN: 2374-2437. DOI: 10.1001/jamaoncol.2016.6374.

[171] Joshua Chang Mell and Rosemary J. Redfield. 'Natural Competence and the Evolution of DNA Uptake Specificity'. en. In: *Journal of Bacteriology* 196.8 (Apr. 2014), pp. 1471–1483. ISSN: 0021-9193, 1098-5530. DOI: 10.1128/JB.01293-13.

[172] Daniel R. Mende et al. 'proGenomes: a resource for consistent functional and taxonomic annotations of prokaryotic genomes'. en. In: *Nucleic Acids Research* 45.D1 (Jan. 2017), pp. D529–D534. ISSN: 0305-1048, 1362-4962. DOI: 10.1093/nar/gkw989.

[173] Peter Menzel, Kim Lee Ng and Anders Krogh. 'Fast and sensitive taxonomic classification for metagenomics with Kaiju'. en. In: *Nature Communications* 7.1 (Apr. 2016). Number: 1 Publisher: Nature Publishing Group, p. 11257. ISSN: 2041-1723. DOI: 10.1038/ncomms11257.

[174] MetaHIT Consortium et al. 'A human gut microbial gene catalogue established by metagenomic sequencing'. en. In: *Nature* 464.7285 (Mar. 2010), pp. 59–65. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/nature08821.

[175] MetaHIT Consortium et al. 'Identification and assembly of genomes and genetic elements in complex metagenomic samples without using reference genomes'. en. In: *Nature Biotechnology* 32.8 (Aug. 2014), pp. 822–828. ISSN: 1087-0156, 1546-1696. DOI: 10.1038/nbt.2939.

[176] Fernando Meyer et al. 'Assessing taxonomic metagenome profilers with OPAL'. In: *Genome Biology* 20.1 (Mar. 2019), p. 51. ISSN: 1474-760X. DOI: 10.1186/s13059-019-1646-y.

[177] Fernando Meyer et al. 'Critical Assessment of Metagenome Interpretation: the second round of challenges'. en. In: *Nature Methods* 19.4 (Apr. 2022). Number: 4 Publisher: Nature Publishing Group, pp. 429–440. ISSN: 1548-7105. DOI: 10.1038/s41592-022-01431-4.

[178] Alla Mikheenko, Vladislav Saveliev and Alexey Gurevich. 'MetaQUAST: evaluation of metagenome assemblies'. en. In: *Bioinformatics* 32.7 (Apr. 2016), pp. 1088–1090. ISSN: 1367-4811, 1367-4803. DOI: 10.1093/bioinformatics/btv697.

[179] Alessio Milanese et al. 'Microbial abundance, activity and population genomic profiling with mOTUs2'. en. In: *Nature Communications* 10.1 (Mar. 2019), p. 1014. ISSN: 2041-1723. DOI: 10.1038/s41467-019-08844-4.

[180] Bui Quang Minh et al. 'Corrigendum to: IQ-TREE 2: New Models and Efficient Methods for Phylogenetic Inference in the Genomic Era'. en. In: *Molecular Biology and Evolution* 37.8 (Aug. 2020), pp. 2461–2461. ISSN: 0737-4038, 1537-1719. DOI: 10.1093/molbev/msaa131.

[181] Niema Moshiri. 'ViralMSA: massively scalable reference-guided multiple sequence alignment of viral genomes'. en. In: *Bioinformatics* 37.5 (May 2021). Ed. by Peter Robinson, pp. 714–716. ISSN: 1367-4803, 1367-4811. DOI: 10.1093/bioinformatics/btaa743.

[182] Nadja Mostacci et al. 'Informed interpretation of metagenomic data by StrainPhlAn enables strain retention analyses of the upper airway microbiome'. en. In: *mSystems* 8.6 (Dec. 2023). Ed. by Nicola Segata, e00724–23. ISSN: 2379-5077. DOI: 10.1128/msystems.00724-23.

[183] Paul Muir et al. 'The real cost of sequencing: scaling computation to keep pace with data generation'. en. In: *Genome Biology* 17.1 (Dec. 2016), p. 53. ISSN: 1474-760X. DOI: 10.1186/s13059-016-0917-0.

[184] Supratim Mukherjee et al. 'Large-scale contamination of microbial isolate genomes by Illumina PhiX control'. en. In: *Standards in Genomic Sciences* 10.1 (Mar. 2015), p. 18. ISSN: 1944-3277. DOI: 10.1186/1944-3277-10-18.

[185] Sowmya Nagarajan et al. 'Functions of the Duplicated *hik31* Operons in Central Metabolism and Responses to Light, Dark, and Carbon Sources in Synechocystis sp. Strain PCC 6803'. en. In: *Journal of Bacteriology* 194.2 (Jan. 2012), pp. 448–459. ISSN: 0021-9193, 1098-5530. DOI: 10.1128/JB.06207-11.

[186] Toshiaki Namiki et al. 'MetaVelvet: an extension of Velvet assembler to de novo metagenome assembly from short sequence reads'. en. In: *Nucleic Acids Research* 40.20 (Nov. 2012), e155–e155. ISSN: 1362-4962, 0305-1048. DOI: 10.1093/nar/gks678.

[187] Stephen Nayfach et al. 'New insights from uncultivated genomes of the global human gut microbiome'. en. In: *Nature* 568.7753 (Apr. 2019), pp. 505–510. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/s41586-019-1058-x.

[188] Antonio Nesci et al. 'Gut Microbiota and Cardiovascular Disease: Evidence on the Metabolic and Inflammatory Background of a Complex Relationship'. en. In: *International Journal of Molecular Sciences* 24.10 (May 2023), p. 9087. ISSN: 1422-0067. DOI: 10.3390/ijms24109087.

[189] T Nogami, T Mizuno and S Mizushima. 'Construction of a series of ompF-ompC chimeric genes by in vivo homologous recombination in Escherichia coli and characterization of the translational products'. en. In: *Journal of Bacteriology* 164.2 (Nov. 1985), pp. 797–801. ISSN: 0021-9193, 1098-5530. DOI: 10.1128/jb.164.2.797-801.1985.

[190] Sergey Nurk et al. 'metaSPAdes: a new versatile metagenomic assembler'. In: *Genome Research* 27.5 (May 2017), pp. 824–834. ISSN: 1088-9051. DOI: 10.1101/gr.213959.116.

[191] Sergey Nurk et al. 'HiCanu: accurate assembly of segmental duplications, satellites, and allelic variants from high-fidelity long reads'. eng. In: *Genome Research* 30.9 (Sept. 2020), pp. 1291–1305. ISSN: 1549-5469. DOI: 10.1101/gr.263566.120.

[192] R Nussinov and A B Jacobson. 'Fast algorithm for predicting the secondary structure of single-stranded RNA.' en. In: *Proceedings of the National Academy of Sciences* 77.11 (Nov. 1980), pp. 6309–6313. ISSN: 0027-8424, 1091-6490. DOI: 10.1073/pnas.77.11.6309.

[193] Nuala A. O'Leary et al. 'Reference sequence (RefSeq) database at NCBI: current status, taxonomic expansion, and functional annotation'. en. In: *Nucleic Acids Research* 44.D1 (Jan. 2016), pp. D733–D745. ISSN: 0305-1048, 1362-4962. DOI: 10.1093/nar/gkv1189.

[194] Scott W. Olesen and Eric J. Alm. 'Dysbiosis is not an answer'. en. In: *Nature Microbiology* 1.12 (Nov. 2016), p. 16228. ISSN: 2058-5276. DOI: 10.1038/nmicrobiol.2016.228.

[195] Leonardo de Oliveira Martins et al. 'Taxonomic resolution of the ribosomal RNA operon in bacteria: implications for its use with long-read sequencing'. en. In: *NAR Genomics and Bioinformatics* 2.1 (Mar. 2020), lqz016. ISSN: 2631-9268. DOI: 10.1093/nargab/lqz016.

[196] Matthew R. Olm et al. 'inStrain profiles population microdiversity from metagenomic data and sensitively detects shared microbial strains'. en. In: *Nature Biotechnology* 39.6 (June 2021), pp. 727–736. ISSN: 1087-0156, 1546-1696. DOI: 10.1038/s41587-020-00797-0.

[197] Askarbek Orakov et al. 'GUNC: detection of chimerism and contamination in prokaryotic genomes'. en. In: *Genome Biology* 22.1 (Dec. 2021), p. 178. ISSN: 1474-760X. DOI: 10.1186/s13059-021-02393-0.

[198] A. Oren. 'Nomenclature of prokaryotic 'Candidatus' taxa: establishing order in the current chaos'. eng. In: *New Microbes and New Infections* 44 (Nov. 2021), p. 100932. ISSN: 2052-2975. DOI: 10.1016/j.nmni.2021.100932.

[199] Aharon Oren et al. 'International Code of Nomenclature of Prokaryotes. Prokaryotic Code (2022 Revision)'. en. In: *International Journal of Systematic and Evolutionary Microbiology* 73.5a (May 2023). Publisher: Microbiology Society. ISSN: 1466-5026, 1466-5034. DOI: 10.1099/ijsem.0.005585.

[200] Rachid Ounit et al. 'CLARK: fast and accurate classification of metagenomic and genomic sequences using discriminative k-mers'. en. In: *BMC Genomics* 16.1 (Dec. 2015), p. 236. ISSN: 1471-2164. DOI: 10.1186/s12864-015-1419-2.

[201] Ezgi Özkurt et al. 'LotuS2: an ultrafast and highly accurate tool for amplicon sequencing analysis'. en. In: *Microbiome* 10.1 (Oct. 2022), p. 176. ISSN: 2049-2618. DOI: 10.1186/s40168-022-01365-1.

[202] Fanny-Dhelia Pajuste et al. 'FastGT: an alignment-free method for calling common SNVs directly from raw sequencing reads'. en. In: *Scientific Reports* 7.1 (May 2017), p. 2537. ISSN: 2045-2322. DOI: 10.1038/s41598-017-02487-5.

[203] Shaojun Pan et al. 'A deep siamese neural network improves metagenome-assembled genomes in microbiome datasets across different environments'. en. In: *Nature Communications* 13.1 (Apr. 2022), p. 2326. ISSN: 2041-1723. DOI: 10.1038/s41467-022-29843-y.

[204] Lucas Paoli et al. 'Biosynthetic potential of the global ocean microbiome'. en. In: *Nature* 607.7917 (July 2022), pp. 111–118. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/s41586-022-04862-3.

[205] R. B. Parker and M. L. Snyder. 'Interactions of the Oral Microbiota I. A System for the Defined Study of Mixed Cultures.' en. In: *Experimental Biology and Medicine* 108.3 (Dec. 1961), pp. 749–752. ISSN: 1535-3702, 1535-3699. DOI: 10.3181/00379727-108-27055.

[206] Donovan H Parks et al. 'GTDB: an ongoing census of bacterial and archaeal diversity through a phylogenetically consistent, rank normalized and complete genome-based taxonomy'. In: *Nucleic Acids Research* 50.D1 (Jan. 2022), pp. D785–D794. ISSN: 0305-1048. DOI: 10.1093/nar/gkab776.

[207] Donovan H. Parks et al. 'CheckM: assessing the quality of microbial genomes recovered from isolates, single cells, and metagenomes'. en. In: *Genome Research* 25.7 (July 2015), pp. 1043–1055. ISSN: 1088-9051, 1549-5469. DOI: 10.1101/gr.186072.114.

[208] Donovan H. Parks et al. 'A standardized bacterial taxonomy based on genome phylogeny substantially revises the tree of life'. en. In: *Nature Biotechnology* 36.10 (Nov. 2018). Publisher: Nature Publishing Group, pp. 996–1004. ISSN: 1546-1696. DOI: 10.1038/nbt.4229.

[209] Donovan H. Parks et al. 'A complete domain-to-species taxonomy for Bacteria and Archaea'. en. In: *Nature Biotechnology* 38.9 (Sept. 2020), pp. 1079–1086. ISSN: 1087-0156, 1546-1696. DOI: 10.1038/s41587-020-0501-8.

[210] Edoardo Pasolli et al. 'Machine Learning Meta-analysis of Large Metagenomic Datasets: Tools and Biological Insights'. en. In: *PLOS Computational Biology* 12.7 (July 2016). Ed. by Jonathan A. Eisen, e1004977. ISSN: 1553-7358. DOI: 10.1371/journal.pcbi.1004977.

[211] Edoardo Pasolli et al. 'Extensive Unexplored Human Microbiome Diversity Revealed by Over 150,000 Genomes from Metagenomes Spanning Age, Geography, and Lifestyle'. en. In: *Cell* 176.3 (Jan. 2019), 649–662.e20. ISSN: 00928674. DOI: 10.1016/j.cell.2019.01.001.

[212] Yu Peng et al. 'Meta-IDBA: a *de Novo* assembler for metagenomic data'. en. In: *Bioinformatics* 27.13 (July 2011), pp. i94–i101. ISSN: 1367-4811, 1367-4803. DOI: 10.1093/bioinformatics/btr216.

[213] Enrico Petrucci et al. 'Iterative Spaced Seed Hashing: Closing the Gap Between Spaced Seed Hashing and k-mer Hashing'. en. In: *Bioinformatics Research and Applications*. Ed. by Zhipeng Cai, Pavel Skums and Min Li. Vol. 11490. Series Title: Lecture Notes in Computer Science. Cham: Springer International Publishing, 2019, pp. 208–219. ISBN: 978-3-030-20241-5 978-3-030-20242-2. DOI: 10.1007/978-3-030-20242-2_18.

[214] Franziska Pfeiffer et al. 'Systematic evaluation of error rates and causes in short samples in next-generation sequencing'. en. In: *Scientific Reports* 8.1 (July 2018), p. 10950. ISSN: 2045-2322. DOI: 10.1038/s41598-018-29325-6.

[215] Daniel Podlesny et al. 'Metagenomic strain detection with SameStr: identification of a persisting core gut microbiota transferable by fecal transplantation'. en. In: *Microbiome* 10.1 (Mar. 2022), p. 53. ISSN: 2049-2618. DOI: 10.1186/s40168-022-01251-w.

[216] Ryan Poplin et al. *Scaling accurate genetic variant discovery to tens of thousands of samples*. en. Nov. 2017. DOI: 10.1101/201178.

[217] Daniel M. Portik, C. Titus Brown and N. Tessa Pierce-Ward. 'Evaluation of taxonomic classification and profiling methods for long-read shotgun metagenomic sequencing datasets'. en. In: *BMC Bioinformatics* 23.1 (Dec. 2022), p. 541. ISSN: 1471-2105. DOI: 10.1186/s12859-022-05103-0.

[218] James M. Prober et al. 'A System for Rapid DNA Sequencing with Fluorescent Chain-Terminating Dideoxynucleotides'. en. In: *Science* 238.4825 (Oct. 1987), pp. 336–341. ISSN: 0036-8075, 1095-9203. DOI: 10.1126/science.2443975.

[219] Vaidehi Pusadkar and Rajeev K. Azad. 'Benchmarking Metagenomic Classifiers on Simulated Ancient and Modern Metagenomic Data'. en. In: *Microorganisms* 11.10 (Oct. 2023), p. 2478. ISSN: 2076-2607. DOI: 10.3390/microorganisms11102478.

[220] Yujia Qin et al. 'Effects of error, chimera, bias, and GC content on the accuracy of amplicon sequencing'. en. In: *mSystems* 8.6 (Dec. 2023). Ed. by Jack A. Gilbert, e01025–23. ISSN: 2379-5077. DOI: 10.1128/msystems.01025-23.

[221] Christopher Quince et al. 'Shotgun metagenomics, from sampling to analysis'. en. In: *Nature Biotechnology* 35.9 (Sept. 2017), pp. 833–844. ISSN: 1087-0156, 1546-1696. DOI: 10.1038/nbt.3935.

[222] Mahmudur Rahman Hera, N. Tessa Pierce-Ward and David Koslicki. 'Deriving confidence intervals for mutation rates across a wide range of evolutionary distances using FracMinHash'. en. In: *Genome Research* (June 2023), genome, gr.277651.123v2. ISSN: 1088-9051, 1549-5469. DOI: 10.1101/gr.277651.123.

[223] J E Rebollo, V François and J M Louarn. 'Detection and possible role of two large nondivisible zones on the Escherichia coli chromosome.' en. In: *Proceedings of the National Academy of Sciences* 85.24 (Dec. 1988), pp. 9391–9395. ISSN: 0027-8424, 1091-6490. DOI: 10.1073/pnas.85.24.9391.

[224] *Reconciling Microbial Systematics & Genomics: This report is based on a colloquium, sponsored by the American Academy of Microbiology, convened September 27–28, 2006, in Washington, DC.* eng. American Academy of Microbiology Colloquia Reports. Washington (DC): American Society for Microbiology, 2006.

[225] David Ribet and Pascale Cossart. 'How bacterial pathogens colonize their hosts and invade deeper tissues'. en. In: *Microbes and Infection* 17.3 (Mar. 2015), pp. 173–183. ISSN: 12864579. DOI: 10.1016/j.micinf.2015.01.004.

[226] Michael Richter and Ramon Rosselló-Móra. 'Shifting the genomic gold standard for the prokaryotic species definition'. en. In: *Proceedings of the National Academy of Sciences* 106.45 (Nov. 2009), pp. 19126–19131. ISSN: 0027-8424, 1091-6490. DOI: 10.1073/pnas.0906412106.

[227] Michael Roberts et al. 'Reducing storage requirements for biological sequence comparison'. en. In: *Bioinformatics* 20.18 (Dec. 2004), pp. 3363–3369. ISSN: 1367-4811, 1367-4803. DOI: 10.1093/bioinformatics/bth408.

[228] D.F. Robinson and L.R. Foulds. 'Comparison of phylogenetic trees'. en. In: *Mathematical Biosciences* 53.1-2 (Feb. 1981), pp. 131–147. ISSN: 00255564. DOI: 10.1016/0025-5564(81)90043-2.

[229] Luis M. Rodriguez-R et al. 'How Much Do rRNA Gene Surveys Underestimate Extant Bacterial Diversity?' en. In: *Applied and Environmental Microbiology* 84.6 (Mar. 2018). Ed. by Frank E. Löffler, e00014–18. ISSN: 0099-2240, 1098-5336. DOI: 10.1128/AEM.00014-18.

[230] Stefano Romano et al. 'Meta-analysis of the Parkinson's disease gut microbiome suggests alterations linked to intestinal inflammation'. en. In: *npj Parkinson's Disease* 7.1 (Mar. 2021). Number: 1 Publisher: Nature Publishing Group, pp. 1–13. ISSN: 2373-8057. DOI: 10.1038/s41531-021-00156-z.

[231] Mostafa Ronaghi, Mathias Uhlén and Pål Nyrén. 'A Sequencing Method Based on Real-Time Pyrophosphate'. en. In: *Science* 281.5375 (July 1998), pp. 363–365. ISSN: 0036-8075, 1095-9203. DOI: 10.1126/science.281.5375.363.

[232] Hans-Joachim Ruscheweyh et al. *Reference genome-independent taxonomic profiling of microbiomes with mOTUs3.* en. Pages: 2021.04.20.440600 Section: New Results. Apr. 2022. DOI: 10.1101/2021.04.20.440600.

[233] P. S.Hiremath, Parashuram Bannigidad and Soumyashree S. Yelgond. 'Identification of Flagellated or Fimbriated Bacterial Cells using Digital Image Processing Techniques'. In: *International Journal of Computer Applications* 59.12 (Dec. 2012), pp. 12–16. ISSN: 09758887. DOI: 10.5120/9599-4223.

[234] Kristoffer Sahlin. 'Strobealign: flexible seed size enables ultra-fast and accurate read alignment'. en. In: *Genome Biology* 23.1 (Dec. 2022), p. 260. ISSN: 1474-760X. DOI: 10.1186/s13059-022-02831-7.

[235] Jesse J. Salk, Michael W. Schmitt and Lawrence A. Loeb. 'Enhancing the accuracy of next-generation sequencing for detecting rare and subclonal mutations'. en. In: *Nature Reviews Genetics* 19.5 (May 2018), pp. 269–285. ISSN: 1471-0056, 1471-0064. DOI: 10.1038/nrg.2017.117.

[236] F. Sanger, S. Nicklen and A. R. Coulson. 'DNA sequencing with chain-terminating inhibitors'. en. In: *Proceedings of the National Academy of Sciences* 74.12 (Dec. 1977), pp. 5463–5467. ISSN: 0027-8424, 1091-6490. DOI: 10.1073/pnas.74.12.5463.

[237] Melanie Schirmer et al. 'Illumina error profiles: resolving fine-scale variation in metagenomic sequencing data'. en. In: *BMC Bioinformatics* 17.1 (Mar. 2016), p. 125. ISSN: 1471-2105. DOI: 10.1186/s12859-016-0976-y.

[238] Klaus Peter Schliep. 'phangorn: phylogenetic analysis in R'. en. In: *Bioinformatics* 27.4 (Feb. 2011), pp. 592–593. ISSN: 1367-4811, 1367-4803. DOI: 10.1093/bioinformatics/btq706.

[239] Patrick D. Schloss et al. 'Introducing mothur: Open-Source, Platform-Independent, Community-Supported Software for Describing and Comparing Microbial Communities'. en. In: *Applied and Environmental Microbiology* 75.23 (Dec. 2009), pp. 7537–7541. ISSN: 0099-2240, 1098-5336. DOI: 10.1128/AEM.01541-09.

[240] Conrad L Schoch et al. 'NCBI Taxonomy: a comprehensive update on curation, resources and tools'. en. In: *Database* 2020 (Jan. 2020), baaa062. ISSN: 1758-0463. DOI: 10.1093/database/baaa062.

[241] Matthias Scholz et al. 'Strain-level microbial epidemiology and population genomics from shotgun metagenomics'. en. In: *Nature Methods* 13.5 (May 2016), pp. 435–438. ISSN: 1548-7091, 1548-7105. DOI: 10.1038/nmeth.3802.

[242] Alexander Sczyrba et al. 'Critical Assessment of Metagenome Interpretation—a benchmark of metagenomics software'. en. In: *Nature Methods* 14.11 (Nov. 2017), pp. 1063–1071. ISSN: 1548-7091, 1548-7105. DOI: 10.1038/nmeth.4458.

[243] Karel Sedlar, Kristyna Kupkova and Ivo Provaznik. 'Bioinformatics strategies for taxonomy independent binning and visualization of sequences in shotgun metagenomics'. en. In: *Computational and Structural Biotechnology Journal* 15 (2017), pp. 48–55. ISSN: 20010370. DOI: 10.1016/j.csbj.2016.11.005.

[244] Torsten Seemann. 'Prokka: rapid prokaryotic genome annotation'. en. In: *Bioinformatics* 30.14 (July 2014), pp. 2068–2069. ISSN: 1367-4811, 1367-4803. DOI: 10.1093/bioinformatics/btu153.

[245] Torsten Seemann. *snippy: fast bacterial variant calling from NGS reads*. 2015.

[246] Nicola Segata et al. 'Metagenomic microbial community profiling using unique clade-specific marker genes'. en. In: *Nature Methods* 9.8 (Aug. 2012), pp. 811–814. ISSN: 1548-7091, 1548-7105. DOI: 10.1038/nmeth.2066.

[247] Ron Sender, Shai Fuchs and Ron Milo. 'Revised Estimates for the Number of Human and Bacteria Cells in the Body'. eng. In: *PLoS biology* 14.8 (Aug. 2016), e1002533. ISSN: 1545-7885. DOI: 10.1371/journal.pbio.1002533.

[248] Fergus Shanahan, Tarini S. Ghosh and Paul W. O'Toole. 'The Healthy Microbiome—What Is the Definition of a Healthy Gut Microbiome?' en. In: *Gastroenterology* 160.2 (Jan. 2021), pp. 483–494. ISSN: 00165085. DOI: 10.1053/j.gastro.2020.09.057.

[249] Richard Shen et al. 'High-throughput SNP genotyping on universal bead arrays'. en. In: *Mutation Research/Fundamental and Molecular Mechanisms of Mutagenesis* 573.1-2 (June 2005), pp. 70–82. ISSN: 00275107. DOI: 10.1016/j.mrfmmm.2004.07.022.

[250] Zhou Jason Shi et al. 'Fast and accurate metagenotyping of the human gut microbiome with GT-Pro'. en. In: *Nature Biotechnology* 40.4 (Apr. 2022), pp. 507–516. ISSN: 1087-0156, 1546-1696. DOI: 10.1038/s41587-021-01102-3.

[251] Christian M. K. Sieber et al. 'Recovery of genomes from metagenomes via a dereplication, aggregation and scoring strategy'. en. In: *Nature Microbiology* 3.7 (May 2018), pp. 836–843. ISSN: 2058-5276. DOI: 10.1038/s41564-018-0171-1.

[252] Michael Silverman et al. 'Protective major histocompatibility complex allele prevents type 1 diabetes by shaping the intestinal microbiota early in ontogeny'. en. In: *Proceedings of the National Academy of Sciences* 114.36 (Sept. 2017), pp. 9671–9676. ISSN: 0027-8424, 1091-6490. DOI: 10.1073/pnas.1712280114.

[253] Jared T Simpson et al. 'Detecting DNA cytosine methylation using nanopore sequencing'. en. In: *Nature Methods* 14.4 (Apr. 2017), pp. 407–410. ISSN: 1548-7091, 1548-7105. DOI: 10.1038/nmeth.4184.

[254] 'Skani enables accurate and efficient genome comparison for modern metagenomic datasets'. en. In: *Nature Methods* 20.11 (Nov. 2023), pp. 1633–1634. ISSN: 1548-7091, 1548-7105. DOI: 10.1038/s41592-023-02019-2.

[255] V. B. D. Skerman, P. H. A. Sneath and Vicki McGOWAN. 'Approved Lists of Bacterial Names'. en. In: *International Journal of Systematic and Evolutionary Microbiology* 30.1 (Jan. 1980). Publisher: Microbiology Society, pp. 225–420. ISSN: 1466-5026, 1466-5034. DOI: 10.1099/00207713-30-1-225.

[256] *smarco/WFA2-lib: WFA2-lib: Wavefront alignment algorithm library v2.*

[257] Christopher S. Smillie et al. 'Strain Tracking Reveals the Determinants of Bacterial Engraftment in the Human Gut Following Fecal Microbiota Transplantation'. en. In: *Cell Host & Microbe* 23.2 (Feb. 2018), 229–240.e5. ISSN: 19313128. DOI: 10.1016/j.chom.2018.01.003.

[258] Alan Jay Smith. 'Cache Memories'. en. In: *ACM Computing Surveys* 14.3 (Sept. 1982), pp. 473–530. ISSN: 0360-0300, 1557-7341. DOI: 10.1145/356887.356892.

[259] T.F. Smith and M.S. Waterman. 'Identification of common molecular subsequences'. en. In: *Journal of Molecular Biology* 147.1 (Mar. 1981), pp. 195–197. ISSN: 00222836. DOI: 10.1016/0022-2836(81)90087-5.

[260] Alice J. Sommer et al. 'A randomization-based causal inference framework for uncovering environmental exposure effects on human gut microbiota'. en. In: *PLOS Computational Biology* 18.5 (May 2022). Ed. by Simon Anders, e1010044. ISSN: 1553-7358. DOI: 10.1371/journal.pcbi.1010044.

[261] Aymé Spor, Omry Koren and Ruth Ley. 'Unravelling the effects of the environment and host genotype on the gut microbiome'. en. In: *Nature Reviews Microbiology* 9.4 (Apr. 2011), pp. 279–290. ISSN: 1740-1526, 1740-1534. DOI: 10.1038/nrmicro2540.

[262] E. Stackebrandt and B. M. Goebel. 'Taxonomic Note: A Place for DNA-DNA Reassociation and 16S rRNA Sequence Analysis in the Present Species Definition in Bacteriology'. en. In: *International Journal of Systematic and Evolutionary Microbiology* 44.4 (Oct. 1994), pp. 846–849. ISSN: 1466-5026, 1466-5034. DOI: 10.1099/00207713-44-4-846.

[263] Erko Stackebrandt, ed. *Molecular Identification, Systematics, and Population Structure of Prokaryotes*. en. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006. ISBN: 978-3-540-23155-4 978-3-540-31292-5. DOI: 10.1007/978-3-540-31292-5.

[264] Erko Stackebrandt and J. Ebers. 'Taxonomic parameters revisited: Tarnished gold standards'. In: *Microbiol Today* 8 (Jan. 2006), pp. 6–9.

[265] Alexandros Stamatakis. 'RAxML version 8: a tool for phylogenetic analysis and post-analysis of large phylogenies'. en. In: *Bioinformatics* 30.9 (May 2014), pp. 1312–1313. ISSN: 1367-4811, 1367-4803. DOI: 10.1093/bioinformatics/btu033.

[266] M. A. Steel and D. Penny. 'Distributions of Tree Comparison Metrics–Some New Results'. en. In: *Systematic Biology* 42.2 (June 1993), pp. 126–141. ISSN: 1063-5157, 1076-836X. DOI: 10.1093/sysbio/42.2.126.

[267] S. Stefani and P.E. Varaldo. 'Epidemiology of methicillin-resistant staphylococci in Europe'. en. In: *Clinical Microbiology and Infection* 9.12 (Dec. 2003), pp. 1179–1186. ISSN: 1198743X. DOI: 10.1111/j.1469-0691.2003.00698.x.

[268] Martin Steinegger and Johannes Söding. 'MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets'. en. In: *Nature Biotechnology* 35.11 (Nov. 2017), pp. 1026–1028. ISSN: 1087-0156, 1546-1696. DOI: 10.1038/nbt.3988.

[269] Robert D. Stewart et al. 'MAGpy: a reproducible pipeline for the downstream analysis of metagenome-assembled genomes (MAGs)'. eng. In: *Bioinformatics (Oxford, England)* 35.12 (June 2019), pp. 2150–2152. ISSN: 1367-4811. DOI: 10.1093/bioinformatics/bty905.

[270] Antonia Suau et al. 'Direct Analysis of Genes Encoding 16S rRNA from Complex Communities Reveals Many Novel Molecular Species within the Human Gut'. en. In: *Applied and Environmental Microbiology* 65.11 (Nov. 1999), pp. 4799–4807. ISSN: 0099-2240, 1098-5336. DOI: 10.1128/AEM.65.11.4799-4807.1999.

[271] Dongchang Sun et al. 'Editorial: Horizontal gene transfer mediated bacterial antibiotic resistance, volume II'. In: *Frontiers in Microbiology* 14 (June 2023), p. 1221606. ISSN: 1664-302X. DOI: 10.3389/fmicb.2023.1221606.

[272] Shinichi Sunagawa et al. 'Metagenomic species profiling using universal phylogenetic marker genes'. en. In: *Nature Methods* 10.12 (Dec. 2013), pp. 1196–1199. ISSN: 1548-7091, 1548-7105. DOI: 10.1038/nmeth.2693.

[273] Lorenzo Tattini, Romina D'Aurizio and Alberto Magi. 'Detection of Genomic Structural Variants from Next-Generation Sequencing Data'. In: *Frontiers in Bioengineering and Biotechnology* 3 (June 2015). ISSN: 2296-4185. DOI: 10.3389/fbioe.2015.00092.

[274] The Genome Standards Consortium et al. 'Minimum information about a single amplified genome (MISAG) and a metagenome-assembled genome (MIMAG) of bacteria and archaea'. en. In: *Nature Biotechnology* 35.8 (Aug. 2017), pp. 725–731. ISSN: 1087-0156, 1546-1696. DOI: 10.1038/nbt.3893.

[275] Christopher M. Thomas and Kaare M. Nielsen. 'Mechanisms of, and Barriers to, Horizontal Gene Transfer between Bacteria'. en. In: *Nature Reviews Microbiology* 3.9 (Sept. 2005), pp. 711–721. ISSN: 1740-1526, 1740-1534. DOI: 10.1038/nrmicro1234.

[276] Gerry Tonkin-Hill et al. 'Producing polished prokaryotic pangenomes with the Panaroo pipeline'. en. In: *Genome Biology* 21.1 (Dec. 2020), p. 180. ISSN: 1474-760X. DOI: 10.1186/s13059-020-02090-4.

[277] Duy Tin Truong et al. 'MetaPhlAn2 for enhanced metagenomic taxonomic profiling'. en. In: *Nature Methods* 12.10 (Oct. 2015), pp. 902–903. ISSN: 1548-7091, 1548-7105. DOI: 10.1038/nmeth.3589.

[278] Ming-Hsin Tsai et al. 'A New Genome-to-Genome Comparison Approach for Large-Scale Revisiting of Current Microbial Taxonomy'. eng. In: *Microorganisms* 7.6 (June 2019), p. 161. ISSN: 2076-2607. DOI: 10.3390/microorganisms7060161.

[279] Tracy Tucker, Marco Marra and Jan M. Friedman. 'Massively parallel sequencing: the next big thing in genetic medicine'. eng. In: *American Journal of Human Genetics* 85.2 (Aug. 2009), pp. 142–154. ISSN: 1537-6605. DOI: 10.1016/j.ajhg.2009.06.022.

[280] Peter J. Turnbaugh et al. 'The Human Microbiome Project'. en. In: *Nature* 449.7164 (Oct. 2007), pp. 804–810. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/nature06244.

[281] Gene W. Tyson et al. 'Community structure and metabolism through reconstruction of microbial genomes from the environment'. en. In: *Nature* 428.6978 (Mar. 2004), pp. 37–43. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/nature02340.

[282] Gherman V. Uritskiy, Jocelyne DiRuggiero and James Taylor. 'MetaWRAP—a flexible pipeline for genome-resolved metagenomic data analysis'. en. In: *Microbiome* 6.1 (Dec. 2018), p. 158. ISSN: 2049-2618. DOI: 10.1186/s40168-018-0541-1.

[283] Mireia Valles-Colomer et al. 'The person-to-person transmission landscape of the gut and oral microbiomes'. en. In: *Nature* 614.7946 (Feb. 2023), pp. 125–135. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/s41586-022-05620-1.

[284] Lucas R. Van Dijk et al. 'StrainGE: a toolkit to track and characterize low-abundance strains in complex microbial communities'. en. In: *Genome Biology* 23.1 (Dec. 2022), p. 74. ISSN: 1474-760X. DOI: 10.1186/s13059-022-02630-0.

[285] Thea Van Rossum et al. 'Diversity within species: interpreting strains in microbiomes'. en. In: *Nature Reviews Microbiology* 18.9 (Sept. 2020), pp. 491–506. ISSN: 1740-1526, 1740-1534. DOI: 10.1038/s41579-020-0368-1.

[286] Thea Van Rossum et al. 'metaSNV v2: detection of SNVs and subspecies in prokaryotic metagenomes'. en. In: *Bioinformatics* 38.4 (Jan. 2022). Ed. by Russell Schwartz, pp. 1162–1164. ISSN: 1367-4803, 1367-4811. DOI: 10.1093/bioinformatics/btab789.

[287] Doris Vandeputte et al. 'Quantitative microbiome profiling links gut community variation to microbial load'. en. In: *Nature* 551.7681 (Nov. 2017), pp. 507–511. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/nature24460.

[288] Irina M. Velsko et al. 'Selection of Appropriate Metagenome Taxonomic Classifiers for Ancient Microbiome Research'. en. In: *mSystems* 3.4 (Aug. 2018). Ed. by Thomas J. Sharpton, e00080–18. ISSN: 2379-5077. DOI: 10.1128/mSystems.00080-18.

[289] E. Viguera. 'Replication slippage involves DNA polymerase pausing and dissociation'. In: *The EMBO Journal* 20.10 (May 2001), pp. 2587–2595. ISSN: 14602075. DOI: 10.1093/emboj/20.10.2587.

[290] F. A. Bastiaan Von Meijenfeldt et al. 'Robust taxonomic classification of uncharted microbial sequences and bins with CAT and BAT'. en. In: *Genome Biology* 20.1 (Dec. 2019), p. 217. ISSN: 1474-760X. DOI: 10.1186/s13059-019-1817-x.

[291] Solize Vosloo et al. 'Evaluating de Novo Assembly and Binning Strategies for Time Series Drinking Water Metagenomes'. eng. In: *Microbiology Spectrum* 9.3 (Dec. 2021), e0143421. ISSN: 2165-0497. DOI: 10.1128/Spectrum.01434-21.

[292] Ji Wang, Wei-Dong Chen and Yan-Dong Wang. 'The Relationship Between Gut Microbiota and Inflammatory Diseases: The Role of Macrophages'. In: *Frontiers in Microbiology* 11 (June 2020), p. 1065. ISSN: 1664-302X. DOI: 10.3389/fmicb.2020.01065.

[293] Qiong Wang et al. 'Naïve Bayesian Classifier for Rapid Assignment of rRNA Sequences into the New Bacterial Taxonomy'. en. In: *Applied and Environmental Microbiology* 73.16 (Aug. 2007), pp. 5261–5267. ISSN: 0099-2240, 1098-5336. DOI: 10.1128/AEM.00062-07.

[294] Jillian L. Waters and Ruth E. Ley. 'The human gut bacteria Christensenellaceae are widespread, heritable, and associated with health'. en. In: *BMC Biology* 17.1 (Dec. 2019), p. 83. ISSN: 1741-7007. DOI: 10.1186/s12915-019-0699-4.

[295] Peter Weiner. 'Linear pattern matching algorithms'. In: *14th Annual Symposium on Switching and Automata Theory (swat 1973)*. USA: IEEE, Oct. 1973, pp. 1–11. DOI: 10.1109/SWAT.1973.13.

[296] Franziska Wemheuer et al. 'Tax4Fun2: prediction of habitat-specific functional profiles and functional redundancy based on 16S rRNA gene sequences'. en. In: *Environmental Microbiome* 15.1 (May 2020), p. 11. ISSN: 2524-6372. DOI: 10.1186/s40793-020-00358-7.

[297] Patrick T West, Rachael B Chanin and Ami S Bhatt. 'From genome structure to function: insights into structural variation in microbiology'. en. In: *Current Opinion in Microbiology* 69 (Oct. 2022), p. 102192. ISSN: 13695274. DOI: 10.1016/j.mib.2022.102192.

[298] William B. Whitman et al. 'Development of the SeqCode: A proposed nomenclatural code for uncultivated prokaryotes with DNA sequences as type'. en. In: *Systematic and Applied Microbiology* 45.5 (Sept. 2022), p. 126305. ISSN: 07232020. DOI: 10.1016/j.syapm.2022.126305.

[299] Ryan R. Wick, Louise M. Judd and Kathryn E. Holt. 'Assembling the perfect bacterial genome using Oxford Nanopore and Illumina sequencing'. en. In: *PLOS Computational Biology* 19.3 (Mar. 2023). Ed. by Francis Ouellette, e1010905. ISSN: 1553-7358. DOI: 10.1371/journal.pcbi.1010905.

[300] Ryan R. Wick et al. 'Unicycler: Resolving bacterial genome assemblies from short and long sequencing reads'. en. In: *PLOS Computational Biology* 13.6 (June 2017). Ed. by Adam M. Phillippy, e1005595. ISSN: 1553-7358. DOI: 10.1371/journal.pcbi.1005595.

[301] Daniel James Wilkinson et al. 'Genomic diversity of *Helicobacter pylori* populations from different regions of the human stomach'. en. In: *Gut Microbes* 14.1 (Dec. 2022), p. 2152306. ISSN: 1949-0976, 1949-0984. DOI: 10.1080/19490976.2022.2152306.

[302] K H Wilson and R B Blitchington. 'Human colonic biota studied by ribosomal DNA sequence analysis'. en. In: *Applied and Environmental Microbiology* 62.7 (July 1996), pp. 2273–2278. ISSN: 0099-2240, 1098-5336. DOI: 10.1128/aem.62.7.2273-2278.1996.

[303] Thierry Wirth et al. 'Origin, Spread and Demography of the Mycobacterium tuberculosis Complex'. en. In: *PLoS Pathogens* 4.9 (Sept. 2008). Ed. by Mark Achtman, e1000160. ISSN: 1553-7374. DOI: 10.1371/journal.ppat.1000160.

[304] C. R. Woese and G. E. Fox. 'Phylogenetic structure of the prokaryotic domain: the primary kingdoms'. eng. In: *Proceedings of the National Academy of Sciences of the United States of America* 74.11 (Nov. 1977), pp. 5088–5090. ISSN: 0027-8424. DOI: 10.1073/pnas.74.11.5088.

[305] Yuri I. Wolf et al. 'Two fundamentally different classes of microbial genes'. en. In: *Nature Microbiology* 2.3 (Nov. 2016), p. 16208. ISSN: 2058-5276. DOI: 10.1038/nmicrobiol.2016.208.

[306] Richard Wolff, William Shoemaker and Nandita Garud. 'Ecological Stability Emerges at the Level of Strains in the Human Gut Microbiome'. en. In: *mBio* 14.2 (Apr. 2023). Ed. by Rachel Whitaker and John W. Taylor, e02502–22. ISSN: 2150-7511. DOI: 10.1128/mbio.02502-22.

[307] Derrick E. Wood, Jennifer Lu and Ben Langmead. 'Improved metagenomic analysis with Kraken 2'. In: *Genome Biology* 20.1 (Nov. 2019), p. 257. ISSN: 1474-760X. DOI: 10.1186/s13059-019-1891-0.

[308] Derrick E. Wood and Steven L. Salzberg. 'Kraken: ultrafast metagenomic sequence classification using exact alignments'. In: *Genome Biology* 15.3 (Mar. 2014), R46. ISSN: 1474-760X. DOI: 10.1186/gb-2014-15-3-r46.

[309] Dongying Wu, Ladan Doroud and Jonathan A. Eisen. *TreeOTU: Operational Taxonomic Unit Classification Based on Phylogenetic Trees*. Version Number: 1. 2013. DOI: 10.48550/ARXIV.1308.6333.

[310] Gary D. Wu et al. 'Linking Long-Term Dietary Patterns with Gut Microbial Enterotypes'. en. In: *Science* 334.6052 (Oct. 2011), pp. 105–108. ISSN: 0036-8075, 1095-9203. DOI: 10.1126/science.1208344.

[311] Zeyu Xia et al. 'A Review of Parallel Implementations for the Smith–Waterman Algorithm'. en. In: *Interdisciplinary Sciences: Computational Life Sciences* 14.1 (Mar. 2022), pp. 1–14. ISSN: 1913-2751, 1867-1462. DOI: 10.1007/s12539-021-00473-0.

[312] Yiqing Yan, Nimisha Chaturvedi and Raja Appuswamy. 'Accel-Align: a fast sequence mapper and aligner based on the seed–embed–extend method'. In: *BMC Bioinformatics* 22.1 (May 2021), p. 257. ISSN: 1471-2105. DOI: 10.1186/s12859-021-04162-z.

[313] Chao Yang et al. 'A review of computational tools for generating metagenome-assembled genomes from metagenomic sequencing data'. en. In: *Computational and Structural Biotechnology Journal* 19 (2021), pp. 6301–6314. ISSN: 20010370. DOI: 10.1016/j.csbj.2021.11.028.

[314] Moran Yassour et al. 'Strain-Level Analysis of Mother-to-Child Bacterial Transmission during the First Few Months of Life'. eng. In: *Cell Host & Microbe* 24.1 (July 2018), 146–154.e4. ISSN: 1934-6069. DOI: 10.1016/j.chom.2018.06.007.

[315] Lianwei Ye et al. 'High-Resolution Metagenomics of Human Gut Microbiota Generated by Nanopore and Illumina Hybrid Metagenome Assembly'. In: *Frontiers in Microbiology* 13 (May 2022), p. 801587. ISSN: 1664-302X. DOI: 10.3389/fmicb.2022.801587.

[316] Simon H. Ye et al. 'Benchmarking Metagenomics Tools for Taxonomic Classification'. en. In: *Cell* 178.4 (Aug. 2019), pp. 779–794. ISSN: 00928674. DOI: 10.1016/j.cell.2019.07.010.

[317] Pelin Yilmaz et al. 'The SILVA and "All-species Living Tree Project (LTP)" taxonomic frameworks'. en. In: *Nucleic Acids Research* 42.D1 (Jan. 2014), pp. D643–D648. ISSN: 0305-1048, 1362-4962. DOI: 10.1093/nar/gkt1209.

[318] Byung-Jun Yoon. 'Hidden Markov Models and their Applications in Biological Sequence Analysis'. en. In: *Current Genomics* 10.6 (Sept. 2009), pp. 402–415. ISSN: 13892029. DOI: 10.2174/138920209789177575.

[319] Cheng Yuan et al. 'Reconstructing 16S rRNA genes in metagenomic data'. en. In: *Bioinformatics* 31.12 (June 2015), pp. i35–i43. ISSN: 1367-4811, 1367-4803. DOI: 10.1093/bioinformatics/btv231.

[320] Xuan Zhang et al. 'The oral and gut microbiomes are perturbed in rheumatoid arthritis and partly normalized after treatment'. en. In: *Nature Medicine* 21.8 (Aug. 2015), pp. 895–905. ISSN: 1078-8956, 1546-170X. DOI: 10.1038/nm.3914.

[321] Yadong Zhang et al. 'ProPan: a comprehensive database for profiling prokaryotic pan-genome dynamics'. en. In: *Nucleic Acids Research* 51.D1 (Jan. 2023), pp. D767–D776. ISSN: 0305-1048, 1362-4962. DOI: 10.1093/nar/gkac832.

[322] Zhenmiao Zhang et al. 'Benchmarking genome assembly methods on metagenomic sequencing data'. en. In: *Briefings in Bioinformatics* 24.2 (Mar. 2023), bbad087. ISSN: 1467-5463, 1477-4054. DOI: 10.1093/bib/bbad087.

[323] Chunyu Zhao et al. 'MIDAS2: Metagenomic Intra-species Diversity Analysis System'. en. In: *Bioinformatics* 39.1 (Jan. 2023). Ed. by Russell Schwartz, btac713. ISSN: 1367-4811. DOI: 10.1093/bioinformatics/btac713.

[324] Danping Zheng, Timur Liwinski and Eran Elinav. 'Interaction between microbiota and immunity in health and disease'. en. In: *Cell Research* 30.6 (June 2020). Number: 6 Publisher: Nature Publishing Group, pp. 492–506. ISSN: 1748-7838. DOI: 10.1038/s41422-020-0332-7.

[325] Ana Zhu et al. 'Inter-individual differences in the gene content of human gut bacterial species'. en. In: *Genome Biology* 16.1 (Dec. 2015), p. 82. ISSN: 1474-760X. DOI: 10.1186/s13059-015-0646-9.

[326] Aleksey V. Zimin et al. 'Hybrid assembly of the large and highly repetitive genome of Aegilops tauschii, a progenitor of bread wheat, with the MaSuRCA mega-reads algorithm'. eng. In: *Genome Research* 27.5 (May 2017), pp. 787–792. ISSN: 1549-5469. DOI: 10.1101/gr.213405.116.

Appendix A

Appendix

A.1 Chapter 5

A.1.1 Benchpro

Table A.1: Number of species for utilized benchmark datasets CAMI Human, CAMI Mouse, and CAMI Human for NCBI, GTDB r207, and GTDB r214.

Dataset	NCBI	GTDB r207	GTDB r214
Airways 10	69	67	67
Airways 11	34	33	33
Airways 12	45	44	44
Airways 23	32	31	31
Airways 26	71	67	67
Airways 27	90	92	92
Airways 4	109	110	109
Airways 7	75	78	78
Airways 8	57	56	56
Airways 9	93	96	96
Gastrointestinal 0	35	35	35
Gastrointestinal 10	40	43	43
Gastrointestinal 11	55	59	59
Gastrointestinal 12	26	29	29
Gastrointestinal 1	38	39	39
Gastrointestinal 2	21	23	23
Gastrointestinal 3	45	45	45
Gastrointestinal 4	45	45	45
Gastrointestinal 5	19	20	20
Gastrointestinal 9	36	39	39
Oral 13	73	75	75
Oral 14	64	65	65

Oral 15	78	79	79
Oral 16	129	139	139
Oral 17	53	58	57
Oral 18	74	79	79
Oral 19	96	101	101
Oral 6	46	48	48
Oral 7	59	65	66
Oral 8	60	63	63
Skin 13	42	42	42
Skin 14	30	30	30
Skin 15	27	26	25
Skin 16	11	10	10
Skin 17	48	47	47
Skin 18	28	27	27
Skin 19	105	111	111
Skin 1	51	52	52
Skin 20	82	81	81
Skin 28	13	13	13
Urogenital 0	34	33	33
Urogenital 21	34	34	34
Urogenital 22	21	21	21
Urogenital 24	27	27	27
Urogenital 25	43	46	46
Urogenital 2	27	26	26
Urogenital 3	48	47	47
Urogenital 5	25	25	25
Urogenital 6	15	16	16
Marine 0	303	333	331
Marine 1	277	313	312
Marine 2	384	430	429
Marine 3	280	318	317
Marine 4	334	371	370
Marine 5	329	364	365
Marine 6	285	317	317
Marine 7	348	390	390
Marine 8	274	302	302
Mouse 0	64	64	64
Mouse 1	82	81	81
Mouse 2	75	75	73
Mouse 5	206	201	200
Mouse 6	213	204	204

Mouse 7	68	69	69
Mouse 8	92	84	85
Mouse 9	95	89	90
Mouse 10	120	122	121
Mouse 11	158	145	144
Mouse 12	162	158	158
Mouse 13	111	111	111
Mouse 14	162	153	153
Mouse 15	170	166	165
Mouse 16	75	76	76
Mouse 17	138	138	138
Mouse 18	73	75	75
Mouse 19	132	128	128
Mouse 20	147	149	148
Mouse 21	105	107	106
Mouse 22	206	204	204
Mouse 23	102	105	104
Mouse 24	203	203	202
Mouse 25	201	194	193
Mouse 26	101	102	101
Mouse 27	204	198	199
Mouse 28	196	190	191
Mouse 29	108	111	109
Mouse 30	207	198	198
Mouse 31	154	155	153
Mouse 32	49	48	47
Mouse 33	204	206	204
Mouse 34	225	222	222
Mouse 35	63	62	61
Mouse 36	193	188	187
Mouse 37	212	210	209
Mouse 38	97	97	97
Mouse 52	100	91	91
Mouse 53	154	156	155
Mouse 54	165	163	163
Mouse 55	183	178	177
Mouse 56	164	157	157
Mouse 57	168	162	161
Mouse 58	222	215	213
Mouse 59	215	205	206
Mouse 60	225	217	216

Mouse 61	231	223	222
Mouse 62	83	83	82
Mouse 63	59	60	59

Table A.4: All (isolate) genomes from humgut that are part of the strain-analysis dataset and their GTDB r214 species annotations generated with GTDB-tk 2.32.

HumGut ID	Species
GUT_GENOME000002	s__Anaerobutyricum hallii
GUT_GENOME000003	s__Blautia_A wexlerae
GUT_GENOME000005	s__Mediterraneibacter faecis
GUT_GENOME000012	s__Bacteroides caccae
GUT_GENOME000013	s__Bacteroides cellulosilyticus
GUT_GENOME000028	s__Parabacteroides distasonis
GUT_GENOME000033	s__Bacteroides fragilis
GUT_GENOME000039	s__Bacteroides xylanisolvens
GUT_GENOME000051	s__Thomasclavelia ramosa
GUT_GENOME000061	s__Agathobacter rectalis
GUT_GENOME000070	s__Parabacteroides distasonis
GUT_GENOME000074	s__Ruminococcus_D bicirculans
GUT_GENOME000080	s__Agathobaculum butyriciproducens
GUT_GENOME000084	s__Anaerobutyricum hallii
GUT_GENOME000087	s__Enterobacter hormaechei_A
GUT_GENOME000091	s__Anaerobutyricum hallii
GUT_GENOME000099	s__Bifidobacterium adolescentis
GUT_GENOME000105	s__Enterobacter hormaechei_A
GUT_GENOME000108	s__Anaerostipes hadrus
GUT_GENOME000112	s__Coprococcus eutactus_A
GUT_GENOME000115	s__Bariatricus comes
GUT_GENOME000116	s__Mediterraneibacter faecis
GUT_GENOME000117	s__Roseburia inulinivorans
GUT_GENOME000120	s__Blautia_A obeum
GUT_GENOME000121	s__Blautia_A wexlerae
GUT_GENOME000125	s__Coprococcus eutactus_A
GUT_GENOME000127	s__Anaerobutyricum hallii
GUT_GENOME000132	s__Fusicatenibacter saccharivorans
GUT_GENOME000135	s__Ruminococcus_B gnavus
GUT_GENOME000142	s__Agathobacter rectalis
GUT_GENOME000144	s__Bifidobacterium adolescentis
GUT_GENOME000150	s__Mediterraneibacter faecis
GUT_GENOME000151	s__Anaerostipes hadrus

GUT_GENOME000155	s__Bacteroides thetaiotaomicron
GUT_GENOME000157	s__Faecalibacillus intestinalis
GUT_GENOME000160	s__Bifidobacterium bifidum
GUT_GENOME000164	s__Blautia_A obeum
GUT_GENOME000169	s__Parabacteroides distasonis
GUT_GENOME000170	s__RUG115 sp900066395
GUT_GENOME000171	s__Roseburia inulinivorans
GUT_GENOME000172	s__Coprococcus eutactus_A
GUT_GENOME000173	s__Sarcina perfringens
GUT_GENOME000176	s__Ruminococcus_D bicirculans
GUT_GENOME000178	s__Bifidobacterium adolescentis
GUT_GENOME000179	s__Anaerobutyricum hallii
GUT_GENOME000182	s__Bariatricus comes
GUT_GENOME000192	s__Parabacteroides merdae
GUT_GENOME000194	s__Bifidobacterium adolescentis
GUT_GENOME000196	s__Faecalibacillus intestinalis
GUT_GENOME000204	s__Blautia_A wexlerae
GUT_GENOME000206	s__Blautia_A wexlerae
GUT_GENOME000209	s__Bariatricus comes
GUT_GENOME000210	s__Dorea formicigenerans
GUT_GENOME000216	s__Lachnospira eligens_A
GUT_GENOME000218	s__Anaerostipes hadrus
GUT_GENOME000227	s__Enterococcus faecalis
GUT_GENOME000235	s__Ruminococcus_D bicirculans
GUT_GENOME000237	s__Lachnospira eligens_A
GUT_GENOME000240	s__Bacteroides thetaiotaomicron
GUT_GENOME000241	s__Bacteroides cellulosilyticus
GUT_GENOME000242	s__Parabacteroides distasonis
GUT_GENOME000244	s__Longicatena caecimuris
GUT_GENOME000249	s__Anaerostipes hadrus
GUT_GENOME000253	s__Bariatricus comes
GUT_GENOME000257	s__Bacteroides caccae
GUT_GENOME000261	s__Blautia_A obeum
GUT_GENOME000263	s__Fusicatenibacter saccharivorans
GUT_GENOME000265	s__Agathobaculum butyriciproducens
GUT_GENOME000281	s__Bacteroides ovatus
GUT_GENOME000283	s__Bacteroides caccae
GUT_GENOME000284	s__Anaerobutyricum hallii
GUT_GENOME000286	s__Parabacteroides distasonis
GUT_GENOME000288	s__Faecalibacillus intestinalis
GUT_GENOME000292	s__Agathobacter rectalis

GUT_GENOME000293	s__Blautia_A wexlerae
GUT_GENOME000294	s__Anaerostipes hadrus
GUT_GENOME000297	s__Bariatricus comes
GUT_GENOME000298	s__Coprococcus eutactus_A
GUT_GENOME000300	s__Anaerobutyricum hallii
GUT_GENOME000301	s__Fusicatenibacter saccharivorans
GUT_GENOME000303	s__Agathobacter rectalis
GUT_GENOME000306	s__Ruminococcus_D bicirculans
GUT_GENOME000307	s__Ruminococcus_E bromii_B
GUT_GENOME000315	s__Ruminococcus_E bromii_B
GUT_GENOME000318	s__Enterococcus faecalis
GUT_GENOME000321	s__Lacticaseibacillus rhamnosus
GUT_GENOME000323	s__Enterococcus faecalis
GUT_GENOME000324	s__Sarcina perfringens
GUT_GENOME000326	s__Thomasclavelia ramosa
GUT_GENOME000328	s__Ruminococcus_B gnavus
GUT_GENOME000329	s__Enterococcus faecalis
GUT_GENOME000331	s__Enterococcus faecalis
GUT_GENOME000332	s__Lacticaseibacillus paracasei
GUT_GENOME000334	s__Thomasclavelia ramosa
GUT_GENOME000336	s__Sarcina perfringens
GUT_GENOME000339	s__Lacticaseibacillus rhamnosus
GUT_GENOME000341	s__Enterococcus faecalis
GUT_GENOME000342	s__Sarcina perfringens
GUT_GENOME000344	s__Thomasclavelia ramosa
GUT_GENOME000346	s__Ruminococcus_B gnavus
GUT_GENOME000347	s__Enterococcus faecalis
GUT_GENOME000348	s__Enterococcus faecalis
GUT_GENOME000349	s__Sarcina perfringens
GUT_GENOME000351	s__Enterococcus faecalis
GUT_GENOME000352	s__Sarcina perfringens
GUT_GENOME000355	s__Roseburia inulinivorans
GUT_GENOME000356	s__Blautia_A wexlerae
GUT_GENOME000357	s__Mediterraneibacter faecis
GUT_GENOME000359	s__Bacteroides stercoris
GUT_GENOME000360	s__Coprococcus eutactus_A
GUT_GENOME000362	s__Bariatricus comes
GUT_GENOME000368	s__Parabacteroides distasonis
GUT_GENOME000369	s__Blautia_A obeum
GUT_GENOME000371	s__Agathobacter rectalis
GUT_GENOME000372	s__Bacteroides ovatus

GUT_GENOME000373	s__Bifidobacterium adolescentis
GUT_GENOME000375	s__Bacteroides fragilis
GUT_GENOME000376	s__Faecalibacillus intestinalis
GUT_GENOME000386	s__Dorea formicigenerans
GUT_GENOME000388	s__Lachnospira eligens_A
GUT_GENOME000389	s__Enterococcus faecalis
GUT_GENOME000390	s__Sarcina perfringens
GUT_GENOME000392	s__Enterococcus faecalis
GUT_GENOME000393	s__Sarcina perfringens
GUT_GENOME000397	s__Roseburia inulinivorans
GUT_GENOME000398	s__Blautia_A wexlerae
GUT_GENOME000399	s__Mediterraneibacter faecis
GUT_GENOME000401	s__Bacteroides stercoris
GUT_GENOME000402	s__Coprococcus eutactus_A
GUT_GENOME000404	s__Bariatricus comes
GUT_GENOME000410	s__Parabacteroides distasonis
GUT_GENOME000411	s__Blautia_A obeum
GUT_GENOME000413	s__Agathobacter rectalis
GUT_GENOME000414	s__Bacteroides ovatus
GUT_GENOME000415	s__Bifidobacterium adolescentis
GUT_GENOME000417	s__Bacteroides fragilis
GUT_GENOME000418	s__Faecalibacillus intestinalis
GUT_GENOME000429	s__Dorea formicigenerans
GUT_GENOME000431	s__Lachnospira eligens_A
GUT_GENOME000433	s__Blautia_A wexlerae
GUT_GENOME000435	s__Bifidobacterium breve
GUT_GENOME000437	s__Bifidobacterium adolescentis
GUT_GENOME000441	s__Dorea formicigenerans
GUT_GENOME000446	s__Dorea formicigenerans
GUT_GENOME000449	s__Agathobacter rectalis
GUT_GENOME000453	s__Bacteroides stercoris
GUT_GENOME000454	s__Bacteroides ovatus
GUT_GENOME000455	s__Blautia_A obeum
GUT_GENOME000458	s__Blautia_A obeum
GUT_GENOME000462	s__Blautia_A obeum
GUT_GENOME000465	s__Parabacteroides distasonis
GUT_GENOME000468	s__Blautia_A obeum
GUT_GENOME000470	s__Bacteroides cellulosilyticus
GUT_GENOME000472	s__Bacteroides thetaiotaomicron
GUT_GENOME000474	s__Blautia_A obeum
GUT_GENOME000476	s__Sarcina perfringens

GUT_GENOME000480	s__Bifidobacterium adolescentis
GUT_GENOME000482	s__Bacteroides fragilis
GUT_GENOME000486	s__Blautia_A obeum
GUT_GENOME000492	s__Bacteroides thetaiotaomicron
GUT_GENOME000493	s__Dorea formicigenerans
GUT_GENOME000497	s__Enterococcus faecalis
GUT_GENOME000502	s__Sarcina perfringens
GUT_GENOME000503	s__Lacticaseibacillus paracasei
GUT_GENOME000511	s__Bacteroides fragilis
GUT_GENOME000516	s__Lacticaseibacillus paracasei
GUT_GENOME000517	s__Lacticaseibacillus paracasei
GUT_GENOME000526	s__Bacteroides thetaiotaomicron
GUT_GENOME000527	s__Enterococcus faecalis
GUT_GENOME000528	s__Enterococcus faecalis
GUT_GENOME000529	s__Enterococcus faecalis
GUT_GENOME000538	s__Bifidobacterium adolescentis
GUT_GENOME000541	s__Lacticaseibacillus paracasei
GUT_GENOME000542	s__Bacteroides cellulosilyticus
GUT_GENOME000544	s__Sarcina perfringens
GUT_GENOME000557	s__Enterococcus faecalis
GUT_GENOME000560	s__Lactiplantibacillus plantarum
GUT_GENOME000569	s__Bacteroides caccae
GUT_GENOME000579	s__Bacteroides xylanisolvens
GUT_GENOME000599	s__Lacticaseibacillus rhamnosus
GUT_GENOME000613	s__Enterococcus faecalis
GUT_GENOME000636	s__Bacteroides stercoris
GUT_GENOME000638	s__Bacteroides xylanisolvens
GUT_GENOME000639	s__Bacteroides xylanisolvens
GUT_GENOME000642	s__Bacteroides thetaiotaomicron
GUT_GENOME000643	s__Bacteroides cellulosilyticus
GUT_GENOME000644	s__Ruminococcus_D bicirculans
GUT_GENOME000646	s__Bacteroides stercoris
GUT_GENOME000648	s__Enterococcus faecalis
GUT_GENOME000651	s__Faecalibacillus intestinalis
GUT_GENOME000654	s__Blautia_A obeum
GUT_GENOME000661	s__Enterococcus faecalis
GUT_GENOME000668	s__Enterococcus faecalis
GUT_GENOME000669	s__Bacteroides stercoris
GUT_GENOME000671	s__Bacteroides fragilis
GUT_GENOME000673	s__Enterococcus faecalis
GUT_GENOME000675	s__Enterococcus faecalis

GUT_GENOME000684	s__Bacteroides fragilis
GUT_GENOME000689	s__Bacteroides stercoris
GUT_GENOME000694	s__Parabacteroides distasonis
GUT_GENOME000696	s__Clostridium_Q fessum
GUT_GENOME000697	s__Sarcina perfringens
GUT_GENOME000702	s__Bifidobacterium adolescentis
GUT_GENOME000706	s__Dorea formicigenerans
GUT_GENOME000712	s__Bacteroides stercoris
GUT_GENOME000716	s__Sarcina perfringens
GUT_GENOME000717	s__Ruminococcus_B gnavus
GUT_GENOME000728	s__Thomasclavelia ramosa
GUT_GENOME000735	s__RUG115 sp900066395
GUT_GENOME000736	s__Faecalibacillus intestinalis
GUT_GENOME000738	s__Clostridium_Q fessum
GUT_GENOME000745	s__Bacteroides thetaiotaomicron
GUT_GENOME000751	s__Bacteroides ovatus
GUT_GENOME000756	s__Bacteroides ovatus
GUT_GENOME000762	s__Bacteroides stercoris
GUT_GENOME000765	s__Lachnospira eligens_A
GUT_GENOME000766	s__Agathobacter rectalis
GUT_GENOME000771	s__Thomasclavelia ramosa
GUT_GENOME000777	s__Agathobaculum butyriciproducens
GUT_GENOME000783	s__Bifidobacterium bifidum
GUT_GENOME000788	s__Dorea formicigenerans
GUT_GENOME000789	s__RUG115 sp900066395
GUT_GENOME000794	s__Clostridium_Q fessum
GUT_GENOME000797	s__Mediterraneibacter faecis
GUT_GENOME000798	s__Bacteroides stercoris
GUT_GENOME000801	s__Agathobacter rectalis
GUT_GENOME000803	s__Ruminococcus_B gnavus
GUT_GENOME000804	s__Faecalibacillus intestinalis
GUT_GENOME000806	s__Faecalibacillus intestinalis
GUT_GENOME000807	s__Blautia_A sp003471165
GUT_GENOME000811	s__Blautia_A sp003471165
GUT_GENOME000816	s__Faecalibacillus intestinalis
GUT_GENOME000819	s__Bacteroides ovatus
GUT_GENOME000826	s__Bacteroides thetaiotaomicron
GUT_GENOME000828	s__Blautia_A obeum
GUT_GENOME000834	s__Fusicatenibacter saccharivorans
GUT_GENOME000840	s__Parabacteroides distasonis
GUT_GENOME000841	s__Blautia_A obeum

GUT_GENOME000842	s__Bifidobacterium adolescentis
GUT_GENOME000844	s__Blautia_A wexlerae
GUT_GENOME000847	s__Bacteroides xylanisolvens
GUT_GENOME000850	s__Bacteroides fragilis
GUT_GENOME000854	s__RUG115 sp900066395
GUT_GENOME000858	s__Faecalibacillus intestinalis
GUT_GENOME000860	s__Ruminococcus_E bromii_B
GUT_GENOME000861	s__Bifidobacterium adolescentis
GUT_GENOME000863	s__Agathobaculum butyriciproducens
GUT_GENOME000882	s__Coprococcus eutactus_A
GUT_GENOME000885	s__Bacteroides stercoris
GUT_GENOME000886	s__Bariatricus comes
GUT_GENOME000887	s__Ruminococcus_E bromii_B
GUT_GENOME000888	s__Faecalibacillus intestinalis
GUT_GENOME000891	s__Ruminococcus_D bicirculans
GUT_GENOME000892	s__Coprococcus eutactus_A
GUT_GENOME000895	s__Faecalibacillus intestinalis
GUT_GENOME000896	s__Ruminococcus_E bromii_B
GUT_GENOME000900	s__Faecalibacillus intestinalis
GUT_GENOME000901	s__Ruminococcus_D bicirculans
GUT_GENOME000907	s__Ruminococcus_D bicirculans
GUT_GENOME000908	s__Bacteroides cellulosilyticus
GUT_GENOME000909	s__Agathobacter rectalis
GUT_GENOME000915	s__Ruminococcus_D bicirculans
GUT_GENOME000916	s__Ruminococcus_D bicirculans
GUT_GENOME000920	s__Fusicatenibacter saccharivorans
GUT_GENOME000922	s__Parabacteroides distasonis
GUT_GENOME000923	s__Bariatricus comes
GUT_GENOME000926	s__Agathobacter rectalis
GUT_GENOME000927	s__Agathobacter rectalis
GUT_GENOME000932	s__Ruminococcus_D bicirculans
GUT_GENOME000935	s__Collinsella sp003466125
GUT_GENOME000939	s__Longicatena caecimuris
GUT_GENOME000942	s__Lachnospira eligens_A
GUT_GENOME000945	s__Collinsella sp003466125
GUT_GENOME000946	s__Collinsella sp003466125
GUT_GENOME000951	s__Fusicatenibacter saccharivorans
GUT_GENOME000952	s__Dorea formicigenerans
GUT_GENOME000953	s__Parabacteroides distasonis
GUT_GENOME000954	s__Ruminococcus_D bicirculans
GUT_GENOME000955	s__Ruminococcus_B gnavus

GUT_GENOME000957	s__Longicatena caecimuris
GUT_GENOME000960	s__Fusicatenibacter saccharivorans
GUT_GENOME000961	s__Thomasclavelia ramosa
GUT_GENOME000966	s__Blautia_A wexlerae
GUT_GENOME000968	s__Bacteroides stercoris
GUT_GENOME000970	s__Agathobacter rectalis
GUT_GENOME000972	s__Faecalibacillus intestinalis
GUT_GENOME000973	s__Dorea formicigenerans
GUT_GENOME000974	s__Mediterraneibacter faecis
GUT_GENOME000975	s__Agathobacter rectalis
GUT_GENOME000977	s__Coprococcus eutactus_A
GUT_GENOME000978	s__Parabacteroides merdae
GUT_GENOME000986	s__RUG115 sp900066395
GUT_GENOME000989	s__Thomasclavelia ramosa
GUT_GENOME000990	s__Clostridium_Q fessum
GUT_GENOME000991	s__Bacteroides ovatus
GUT_GENOME000992	s__Ruminococcus_E bromii_B
GUT_GENOME000993	s__Ruminococcus_D bicirculans
GUT_GENOME000997	s__Clostridium_Q fessum
GUT_GENOME000998	s__Blautia_A obeum
GUT_GENOME000999	s__Dorea formicigenerans
GUT_GENOME001000	s__Bifidobacterium adolescentis
GUT_GENOME001001	s__Roseburia inulinivorans
GUT_GENOME001002	s__Blautia_A sp003471165
GUT_GENOME001005	s__Parabacteroides distasonis
GUT_GENOME001009	s__Bacteroides xylanisolvens
GUT_GENOME001012	s__Clostridium_Q fessum
GUT_GENOME001016	s__Agathobaculum butyriciproducens
GUT_GENOME001017	s__RUG115 sp900066395
GUT_GENOME001018	s__Bacteroides cellulosilyticus
GUT_GENOME001021	s__RUG115 sp900066395
GUT_GENOME001031	s__Blautia_A obeum
GUT_GENOME001035	s__Mediterraneibacter faecis
GUT_GENOME001036	s__Agathobacter rectalis
GUT_GENOME001037	s__Agathobaculum butyriciproducens
GUT_GENOME001038	s__Bacteroides thetaiotaomicron
GUT_GENOME001040	s__Faecalibacillus intestinalis
GUT_GENOME001043	s__Ruminococcus_B gnavus
GUT_GENOME001044	s__Bacteroides ovatus
GUT_GENOME001045	s__Bacteroides caccae
GUT_GENOME001047	s__Clostridium_Q fessum

GUT_GENOME001049	s__Blautia_A sp003471165
GUT_GENOME001050	s__Roseburia inulinivorans
GUT_GENOME001053	s__Agathobacter rectalis
GUT_GENOME001054	s__Dorea formicigenerans
GUT_GENOME001057	s__RUG115 sp900066395
GUT_GENOME001058	s__Mediterraneibacter faecis
GUT_GENOME001059	s__RUG115 sp900066395
GUT_GENOME001060	s__Agathobacter rectalis
GUT_GENOME001062	s__Mediterraneibacter faecis
GUT_GENOME001063	s__RUG115 sp900066395
GUT_GENOME001065	s__Ruminococcus_D bicirculans
GUT_GENOME001067	s__Blautia_A obeum
GUT_GENOME001068	s__Blautia_A wexlerae
GUT_GENOME001069	s__RUG115 sp900066395
GUT_GENOME001071	s__Blautia_A wexlerae
GUT_GENOME001073	s__Bacteroides stercoris
GUT_GENOME001075	s__Parabacteroides distasonis
GUT_GENOME001077	s__Mediterraneibacter faecis
GUT_GENOME001079	s__Bacteroides stercoris
GUT_GENOME001080	s__Ruminococcus_E bromii_B
GUT_GENOME001084	s__Bacteroides ovatus
GUT_GENOME001085	s__Bacteroides stercoris
GUT_GENOME001086	s__Parabacteroides merdae
GUT_GENOME001087	s__Ruminococcus_D bicirculans
GUT_GENOME001088	s__Fusicatenibacter saccharivorans
GUT_GENOME001091	s__Bacteroides fragilis
GUT_GENOME001094	s__Bacteroides fragilis
GUT_GENOME001095	s__Mediterraneibacter faecis
GUT_GENOME001097	s__Mediterraneibacter faecis
GUT_GENOME001099	s__Parabacteroides distasonis
GUT_GENOME001100	s__Bacteroides ovatus
GUT_GENOME001104	s__Faecalibacillus intestinalis
GUT_GENOME001108	s__Mediterraneibacter faecis
GUT_GENOME001109	s__Ruminococcus_B gnavus
GUT_GENOME001110	s__Clostridium_Q fessum
GUT_GENOME001116	s__Thomasclavelia ramosa
GUT_GENOME001117	s__Roseburia inulinivorans
GUT_GENOME001118	s__Thomasclavelia ramosa
GUT_GENOME001120	s__Bacteroides thetaiotaomicron
GUT_GENOME001121	s__Bifidobacterium adolescentis
GUT_GENOME001126	s__Bacteroides ovatus

GUT_GENOME001130	s__Bacteroides cellulosilyticus
GUT_GENOME001132	s__Blautia_A obeum
GUT_GENOME001134	s__Faecalibacillus intestinalis
GUT_GENOME001137	s__Clostridium_Q fessum
GUT_GENOME001141	s__Sarcina perfringens
GUT_GENOME001143	s__Collinsella sp003466125
GUT_GENOME001144	s__Dorea formicigenerans
GUT_GENOME001146	s__Mediterraneibacter faecis
GUT_GENOME001147	s__Blautia_A sp003471165
GUT_GENOME001148	s__Anaerobutyricum hallii
GUT_GENOME001149	s__Parabacteroides merdae
GUT_GENOME001150	s__Faecalibacillus intestinalis
GUT_GENOME001156	s__Bacteroides caccae
GUT_GENOME001160	s__Bacteroides fragilis
GUT_GENOME001164	s__Mediterraneibacter faecis
GUT_GENOME001165	s__Fusicatenibacter saccharivorans
GUT_GENOME001166	s__Clostridium_Q fessum
GUT_GENOME001168	s__Bacteroides ovatus
GUT_GENOME001169	s__Blautia_A wexlerae
GUT_GENOME001171	s__Ruminococcus_B gnavus
GUT_GENOME001178	s__Bacteroides fragilis
GUT_GENOME001179	s__Parabacteroides distasonis
GUT_GENOME001180	s__Parabacteroides merdae
GUT_GENOME001182	s__Fusicatenibacter saccharivorans
GUT_GENOME001183	s__Ruminococcus_D bicirculans
GUT_GENOME001186	s__RUG115 sp900066395
GUT_GENOME001191	s__Ruminococcus_B gnavus
GUT_GENOME001195	s__Blautia_A obeum
GUT_GENOME001196	s__Bacteroides stercoris
GUT_GENOME001198	s__Clostridium_Q fessum
GUT_GENOME001199	s__Bacteroides stercoris
GUT_GENOME001200	s__Lachnospira eligens_A
GUT_GENOME001202	s__Parabacteroides merdae
GUT_GENOME001203	s__Agathobaculum butyriciproducens
GUT_GENOME001205	s__Faecalibacillus intestinalis
GUT_GENOME001207	s__Faecalibacillus intestinalis
GUT_GENOME001218	s__Dorea formicigenerans
GUT_GENOME001221	s__Agathobacter rectalis
GUT_GENOME001225	s__Parabacteroides distasonis
GUT_GENOME001228	s__Lachnospira eligens_A
GUT_GENOME001229	s__Bacteroides stercoris

GUT_GENOME001231	s__Bacteroides thetaiotaomicron
GUT_GENOME001235	s__Ruminococcus_D bicirculans
GUT_GENOME001237	s__Faecalibacillus intestinalis
GUT_GENOME001238	s__Ruminococcus_E bromii_B
GUT_GENOME001242	s__Blautia_A obeum
GUT_GENOME001243	s__Blautia_A sp003471165
GUT_GENOME001244	s__Clostridium_Q fessum
GUT_GENOME001247	s__Coprococcus eutactus_A
GUT_GENOME001248	s__Agathobacter rectalis
GUT_GENOME001249	s__Bacteroides xylanisolvens
GUT_GENOME001250	s__Collinsella sp003466125
GUT_GENOME001251	s__Parabacteroides merdae
GUT_GENOME001252	s__Parabacteroides distasonis
GUT_GENOME001258	s__Agathobacter rectalis
GUT_GENOME001261	s__Blautia_A obeum
GUT_GENOME001263	s__Bacteroides xylanisolvens
GUT_GENOME001265	s__Lachnospira eligens_A
GUT_GENOME001267	s__Bacteroides stercoris
GUT_GENOME001271	s__Ruminococcus_E bromii_B
GUT_GENOME001275	s__Dorea formicigenerans
GUT_GENOME001276	s__Bacteroides ovatus
GUT_GENOME001277	s__Lachnospira eligens_A
GUT_GENOME001278	s__Parabacteroides distasonis
GUT_GENOME001280	s__RUG115 sp900066395
GUT_GENOME001281	s__Blautia_A sp003471165
GUT_GENOME001284	s__Ruminococcus_E bromii_B
GUT_GENOME001287	s__Bifidobacterium bifidum
GUT_GENOME001288	s__Lachnospira eligens_A
GUT_GENOME001289	s__Anaerobutyricum hallii
GUT_GENOME001290	s__Bacteroides thetaiotaomicron
GUT_GENOME001291	s__Bacteroides ovatus
GUT_GENOME001293	s__Bifidobacterium adolescentis
GUT_GENOME001296	s__Bacteroides xylanisolvens
GUT_GENOME001299	s__Bacteroides fragilis
GUT_GENOME001301	s__Bacteroides caccae
GUT_GENOME001306	s__Clostridium_Q fessum
GUT_GENOME001311	s__Agathobaculum butyriciproducens
GUT_GENOME001314	s__Longicatena caecimuris
GUT_GENOME001319	s__Blautia_A sp003471165
GUT_GENOME001321	s__Longicatena caecimuris
GUT_GENOME001326	s__Mediterraneibacter faecis

GUT_GENOME001329	s__Bacteroides thetaiotaomicron
GUT_GENOME001330	s__Faecalibacillus intestinalis
GUT_GENOME001338	s__Collinsella sp003466125
GUT_GENOME001343	s__Collinsella sp003466125
GUT_GENOME001344	s__Collinsella sp003466125
GUT_GENOME001347	s__Collinsella sp003466125
GUT_GENOME001348	s__Anaerostipes hadrus
GUT_GENOME001350	s__Collinsella sp003466125
GUT_GENOME001357	s__Bifidobacterium bifidum
GUT_GENOME001360	s__Bifidobacterium adolescentis
GUT_GENOME001362	s__Blautia_A wexlerae
GUT_GENOME001363	s__Ruminococcus_B gnavus
GUT_GENOME001365	s__Bifidobacterium adolescentis
GUT_GENOME001366	s__Bifidobacterium bifidum
GUT_GENOME001368	s__Bifidobacterium adolescentis
GUT_GENOME001369	s__Bacteroides fragilis
GUT_GENOME001374	s__Ruminococcus_B gnavus
GUT_GENOME001378	s__Parabacteroides merdae
GUT_GENOME001379	s__Coprococcus eutactus_A
GUT_GENOME001380	s__Lachnospira eligens_A
GUT_GENOME001382	s__Bacteroides ovatus
GUT_GENOME001384	s__Bacteroides ovatus
GUT_GENOME001385	s__Bifidobacterium adolescentis
GUT_GENOME001386	s__Bacteroides ovatus
GUT_GENOME001388	s__Bacteroides thetaiotaomicron
GUT_GENOME001389	s__Parabacteroides distasonis
GUT_GENOME001391	s__Bacteroides fragilis
GUT_GENOME001393	s__Agathobacter rectalis
GUT_GENOME001394	s__Bacteroides xylanisolvens
GUT_GENOME001397	s__Bacteroides fragilis
GUT_GENOME001407	s__Mediterraneibacter faecis
GUT_GENOME001410	s__Bacteroides caccae
GUT_GENOME001411	s__Parabacteroides distasonis
GUT_GENOME001412	s__Parabacteroides merdae
GUT_GENOME001423	s__Faecalibacillus intestinalis
GUT_GENOME001426	s__Parabacteroides distasonis
GUT_GENOME001427	s__Bacteroides ovatus
GUT_GENOME001428	s__Ruminococcus_B gnavus
GUT_GENOME001430	s__Enterobacter hormaechei_A
GUT_GENOME001433	s__Bifidobacterium bifidum
GUT_GENOME001434	s__Bifidobacterium bifidum

GUT_GENOME001435	s__Parabacteroides distasonis
GUT_GENOME001437	s__Blautia_A wexlerae
GUT_GENOME001439	s__Bifidobacterium bifidum
GUT_GENOME001441	s__Bifidobacterium bifidum
GUT_GENOME001442	s__Blautia_A obeum
GUT_GENOME001444	s__Agathobaculum butyriciproducens
GUT_GENOME001446	s__Blautia_A wexlerae
GUT_GENOME001448	s__Clostridium_Q fessum
GUT_GENOME001463	s__Thomasclavelia ramosa
GUT_GENOME001464	s__Ruminococcus_B gnavus
GUT_GENOME001467	s__Anaerostipes hadrus
GUT_GENOME001469	s__Ruminococcus_B gnavus
GUT_GENOME001472	s__Bariatricus comes
GUT_GENOME001473	s__Bacteroides caccae
GUT_GENOME001474	s__Blautia_A sp003471165
GUT_GENOME001476	s__Mediterraneibacter faecis
GUT_GENOME001489	s__Agathobacter rectalis
GUT_GENOME001491	s__Ruminococcus_B gnavus
GUT_GENOME001492	s__Blautia_A obeum
GUT_GENOME001496	s__Bacteroides thetaiotaomicron
GUT_GENOME001497	s__Fusicatenibacter saccharivorans
GUT_GENOME001504	s__Blautia_A wexlerae
GUT_GENOME001509	s__Roseburia inulinivorans
GUT_GENOME001512	s__Faecalibacillus intestinalis
GUT_GENOME001513	s__Bariatricus comes
GUT_GENOME001515	s__Dorea formicigenerans
GUT_GENOME001516	s__Anaerostipes hadrus
GUT_GENOME001517	s__Faecalibacillus intestinalis
GUT_GENOME001521	s__Parabacteroides distasonis
GUT_GENOME001524	s__Bacteroides stercoris
GUT_GENOME001527	s__Ruminococcus_B gnavus
GUT_GENOME001530	s__Lactiplantibacillus plantarum
GUT_GENOME001535	s__Lactiplantibacillus plantarum
GUT_GENOME001536	s__Anaerostipes hadrus
GUT_GENOME001537	s__Thomasclavelia ramosa
GUT_GENOME001541	s__Parabacteroides distasonis
GUT_GENOME001548	s__Parabacteroides distasonis
GUT_GENOME001551	s__Bacteroides fragilis
GUT_GENOME001553	s__Bacteroides thetaiotaomicron
GUT_GENOME001554	s__Anaerostipes hadrus
GUT_GENOME001555	s__Bacteroides xylanisolvens

GUT_GENOME001556	s__Parabacteroides distasonis
GUT_GENOME001559	s__Agathobacter rectalis
GUT_GENOME001561	s__Faecalibacillus intestinalis
GUT_GENOME001562	s__Parabacteroides merdae
GUT_GENOME001563	s__Bacteroides caccae
GUT_GENOME001564	s__Mediterraneibacter faecis
GUT_GENOME001565	s__Roseburia inulinivorans
GUT_GENOME001568	s__Blautia_A wexlerae
GUT_GENOME001570	s__Bacteroides stercoris
GUT_GENOME001575	s__Mediterraneibacter faecis
GUT_GENOME001576	s__Clostridium_Q fessum
GUT_GENOME001579	s__Blautia_A obeum
GUT_GENOME001580	s__Ruminococcus_B gnavus
GUT_GENOME001582	s__Agathobaculum butyriciproducens
GUT_GENOME001583	s__Parabacteroides distasonis
GUT_GENOME001587	s__Fusicatenibacter saccharivorans
GUT_GENOME001588	s__Collinsella sp003466125
GUT_GENOME001589	s__Ruminococcus_D bicirculans
GUT_GENOME001591	s__Faecalibacillus intestinalis
GUT_GENOME001592	s__Bacteroides stercoris
GUT_GENOME001595	s__Clostridium_Q fessum
GUT_GENOME001599	s__Agathobaculum butyriciproducens
GUT_GENOME001600	s__Fusicatenibacter saccharivorans
GUT_GENOME001601	s__Ruminococcus_E bromii_B
GUT_GENOME001603	s__Fusicatenibacter saccharivorans
GUT_GENOME001605	s__Blautia_A sp003471165
GUT_GENOME001608	s__Longicatena caecimuris
GUT_GENOME001610	s__Mediterraneibacter faecis
GUT_GENOME001611	s__Agathobacter rectalis
GUT_GENOME001613	s__Dorea formicigenerans
GUT_GENOME001615	s__Blautia_A wexlerae
GUT_GENOME001616	s__Faecalibacillus intestinalis
GUT_GENOME001618	s__Mediterraneibacter faecis
GUT_GENOME001620	s__Mediterraneibacter faecis
GUT_GENOME001621	s__Mediterraneibacter faecis
GUT_GENOME001623	s__Blautia_A obeum
GUT_GENOME001624	s__Blautia_A sp003471165
GUT_GENOME001625	s__Fusicatenibacter saccharivorans
GUT_GENOME001629	s__Mediterraneibacter faecis
GUT_GENOME001631	s__Agathobacter rectalis
GUT_GENOME001632	s__Bacteroides ovatus

GUT_GENOME001633	s__Blautia_A sp003471165
GUT_GENOME001637	s__Bacteroides thetaiotaomicron
GUT_GENOME001640	s__Parabacteroides merdae
GUT_GENOME001641	s__Bacteroides xylanisolvens
GUT_GENOME001643	s__Parabacteroides distasonis
GUT_GENOME001648	s__Bacteroides fragilis
GUT_GENOME001650	s__Ruminococcus_D bicirculans
GUT_GENOME001651	s__Bacteroides caccae
GUT_GENOME001654	s__Ruminococcus_E bromii_B
GUT_GENOME001655	s__Agathobacter rectalis
GUT_GENOME001658	s__Faecalibacillus intestinalis
GUT_GENOME001659	s__Ruminococcus_E bromii_B
GUT_GENOME001660	s__Bacteroides ovatus
GUT_GENOME001661	s__Bacteroides stercoris
GUT_GENOME001662	s__Blautia_A wexlerae
GUT_GENOME001663	s__Agathobaculum butyriciproducens
GUT_GENOME001665	s__Parabacteroides distasonis
GUT_GENOME001667	s__Lachnospira eligens_A
GUT_GENOME001669	s__Ruminococcus_B gnavus
GUT_GENOME001670	s__Roseburia inulinivorans
GUT_GENOME001673	s__Mediterraneibacter faecis
GUT_GENOME001674	s__Parabacteroides distasonis
GUT_GENOME001675	s__Lacticaseibacillus paracasei
GUT_GENOME001680	s__Ruminococcus_E bromii_B
GUT_GENOME001683	s__Bifidobacterium adolescentis
GUT_GENOME001684	s__Blautia_A obeum
GUT_GENOME001685	s__Parabacteroides merdae
GUT_GENOME001689	s__Anaerobutyricum hallii
GUT_GENOME001693	s__Blautia_A wexlerae
GUT_GENOME001695	s__Dorea formicigenerans
GUT_GENOME001697	s__Longicatena caecimuris
GUT_GENOME001699	s__Blautia_A wexlerae
GUT_GENOME001701	s__Ruminococcus_E bromii_B
GUT_GENOME001702	s__Bifidobacterium bifidum
GUT_GENOME001705	s__Agathobacter rectalis
GUT_GENOME001706	s__Bifidobacterium adolescentis
GUT_GENOME001710	s__Bacteroides stercoris
GUT_GENOME001711	s__Dorea formicigenerans
GUT_GENOME001715	s__Lachnospira eligens_A
GUT_GENOME001716	s__Blautia_A obeum
GUT_GENOME001717	s__Dorea formicigenerans

GUT_GENOME001719	s__Faecalibacillus intestinalis
GUT_GENOME001720	s__Bacteroides xylanisolvens
GUT_GENOME001721	s__Ruminococcus_B gnavus
GUT_GENOME001722	s__Parabacteroides distasonis
GUT_GENOME001727	s__Thomasclavelia ramosa
GUT_GENOME001737	s__Blautia_A obeum
GUT_GENOME001741	s__Blautia_A wexlerae
GUT_GENOME001742	s__Dorea formicigenerans
GUT_GENOME001746	s__Bacteroides stercoris
GUT_GENOME001747	s__Bacteroides fragilis
GUT_GENOME001748	s__Bacteroides ovatus
GUT_GENOME001754	s__Bifidobacterium adolescentis
GUT_GENOME001755	s__Mediterraneibacter faecis
GUT_GENOME001758	s__Blautia_A obeum
GUT_GENOME001759	s__Blautia_A wexlerae
GUT_GENOME001765	s__Bifidobacterium bifidum
GUT_GENOME001767	s__Agathobacter rectalis
GUT_GENOME001770	s__Roseburia inulinivorans
GUT_GENOME001771	s__Agathobacter rectalis
GUT_GENOME001774	s__Fusicatenibacter saccharivorans
GUT_GENOME001781	s__Dorea formicigenerans
GUT_GENOME001788	s__Clostridium_Q fessum
GUT_GENOME001789	s__Lachnospira eligens_A
GUT_GENOME001795	s__Ruminococcus_D bicirculans
GUT_GENOME001801	s__Parabacteroides distasonis
GUT_GENOME001802	s__Bacteroides caccae
GUT_GENOME001804	s__Agathobacter rectalis
GUT_GENOME001808	s__Agathobacter rectalis
GUT_GENOME001809	s__Blautia_A wexlerae
GUT_GENOME001810	s__Blautia_A wexlerae
GUT_GENOME001812	s__Blautia_A wexlerae
GUT_GENOME001813	s__Dorea formicigenerans
GUT_GENOME001814	s__Blautia_A wexlerae
GUT_GENOME001817	s__Ruminococcus_E bromii_B
GUT_GENOME001821	s__Ruminococcus_D bicirculans
GUT_GENOME001822	s__Fusicatenibacter saccharivorans
GUT_GENOME001823	s__Bacteroides stercoris
GUT_GENOME001824	s__Agathobacter rectalis
GUT_GENOME001825	s__Bacteroides fragilis
GUT_GENOME001826	s__Longicatena caecimuris
GUT_GENOME001827	s__Clostridium_Q fessum

GUT_GENOME001829	s__Anaerobutyricum hallii
GUT_GENOME001830	s__Parabacteroides merdae
GUT_GENOME001832	s__Agathobacter rectalis
GUT_GENOME001833	s__Lachnospira eligens_A
GUT_GENOME001834	s__Mediterraneibacter faecis
GUT_GENOME001838	s__Bacteroides stercoris
GUT_GENOME001840	s__Mediterraneibacter faecis
GUT_GENOME001841	s__Bacteroides ovatus
GUT_GENOME001842	s__Blautia_A wexlerae
GUT_GENOME001843	s__Parabacteroides merdae
GUT_GENOME001844	s__Parabacteroides distasonis
GUT_GENOME001846	s__Bacteroides stercoris
GUT_GENOME001848	s__Bacteroides stercoris
GUT_GENOME001849	s__Bacteroides thetaiotaomicron
GUT_GENOME001850	s__Ruminococcus_B gnavus
GUT_GENOME001855	s__Parabacteroides distasonis
GUT_GENOME001856	s__Blautia_A wexlerae
GUT_GENOME001858	s__Bacteroides caccae
GUT_GENOME001859	s__Ruminococcus_D bicirculans
GUT_GENOME001860	s__Agathobacter rectalis
GUT_GENOME001862	s__Bacteroides xylanisolvens
GUT_GENOME001863	s__Bacteroides ovatus
GUT_GENOME001865	s__Blautia_A obeum
GUT_GENOME001868	s__Bacteroides ovatus
GUT_GENOME001869	s__Mediterraneibacter faecis
GUT_GENOME001874	s__Blautia_A sp003471165
GUT_GENOME015911	s__Bacteroides ovatus
GUT_GENOME015913	s__Ruminococcus_E bromii_B
GUT_GENOME015915	s__Bacteroides fragilis
GUT_GENOME095937	s__Lacticaseibacillus rhamnosus
GUT_GENOME095939	s__Lacticaseibacillus rhamnosus
GUT_GENOME095940	s__Lactiplantibacillus plantarum
GUT_GENOME095945	s__Enterococcus faecalis
GUT_GENOME095946	s__Enterococcus faecalis
GUT_GENOME095947	s__Enterococcus faecalis
GUT_GENOME095948	s__Enterococcus faecalis
GUT_GENOME095949	s__Enterococcus faecalis
GUT_GENOME095950	s__Enterococcus faecalis
GUT_GENOME095951	s__Enterococcus faecalis
GUT_GENOME095954	s__Blautia_A obeum
GUT_GENOME095957	s__Bifidobacterium adolescentis

GUT_GENOME095958	s__Parabacteroides merdae
GUT_GENOME095959	s__Bacteroides ovatus
GUT_GENOME095972	s__Thomasclavelia ramosa
GUT_GENOME095974	s__Bacteroides stercoris
GUT_GENOME095975	s__Anaerostipes hadrus
GUT_GENOME095984	s__Bifidobacterium bifidum
GUT_GENOME095990	s__Lacticaseibacillus paracasei
GUT_GENOME095993	s__Bariatricus comes
GUT_GENOME096013	s__Parabacteroides distasonis
GUT_GENOME096014	s__Bacteroides ovatus
GUT_GENOME096016	s__Bacteroides xylanisolvens
GUT_GENOME096023	s__Bifidobacterium breve
GUT_GENOME096024	s__Bacteroides cellulosilyticus
GUT_GENOME096044	s__Thomasclavelia ramosa
GUT_GENOME096052	s__Enterococcus faecalis
GUT_GENOME096053	s__Enterococcus faecalis
GUT_GENOME096059	s__Lacticaseibacillus paracasei
GUT_GENOME096063	s__Bacteroides fragilis
GUT_GENOME096064	s__Bacteroides thetaiotaomicron
GUT_GENOME096067	s__Blautia_A wexlerae
GUT_GENOME096068	s__Lacticaseibacillus rhamnosus
GUT_GENOME096079	s__Bacteroides fragilis
GUT_GENOME096080	s__Bacteroides xylanisolvens
GUT_GENOME096081	s__Parabacteroides distasonis
GUT_GENOME096085	s__Parabacteroides distasonis
GUT_GENOME096087	s__Parabacteroides distasonis
GUT_GENOME096088	s__Bacteroides ovatus
GUT_GENOME096092	s__Longicatena caecimuris
GUT_GENOME096093	s__Parabacteroides distasonis
GUT_GENOME096094	s__Bacteroides xylanisolvens
GUT_GENOME096127	s__Bacteroides caccae
GUT_GENOME096129	s__Dorea formicigenerans
GUT_GENOME096131	s__Ruminococcus_B gnavus
GUT_GENOME096141	s__Anaerobutyricum hallii
GUT_GENOME096142	s__Roseburia inulinivorans
GUT_GENOME096145	s__Enterococcus faecalis
GUT_GENOME096146	s__Bacteroides xylanisolvens
GUT_GENOME096148	s__Bacteroides ovatus
GUT_GENOME096149	s__Bacteroides xylanisolvens
GUT_GENOME096159	s__Anaerostipes hadrus
GUT_GENOME096163	s__Eggerthella lenta

GUT_GENOME096181	s__Eggerthella lenta
GUT_GENOME096203	s__Bacteroides ovatus
GUT_GENOME096204	s__Bacteroides fragilis
GUT_GENOME096205	s__Bacteroides xylanisolvens
GUT_GENOME096210	s__Ruminococcus_B gnavus
GUT_GENOME096218	s__Dorea formicigenerans
GUT_GENOME096231	s__Lacticaseibacillus rhamnosus
GUT_GENOME096247	s__Thomasclavelia ramosa
GUT_GENOME096253	s__Longicatena caecimuris
GUT_GENOME096255	s__Sarcina perfringens
GUT_GENOME096257	s__Thomasclavelia ramosa
GUT_GENOME096265	s__Bifidobacterium bifidum
GUT_GENOME096280	s__Parabacteroides distasonis
GUT_GENOME096303	s__Anaerostipes hadrus
GUT_GENOME096379	s__Bifidobacterium breve
GUT_GENOME096397	s__Bifidobacterium bifidum
GUT_GENOME096429	s__Bacteroides xylanisolvens
GUT_GENOME096433	s__Lactiplantibacillus plantarum
GUT_GENOME096434	s__Enterococcus faecalis
GUT_GENOME096444	s__Faecalibacillus intestinalis
GUT_GENOME096453	s__Eggerthella lenta
GUT_GENOME096456	s__Bifidobacterium breve
GUT_GENOME096457	s__Bifidobacterium bifidum
GUT_GENOME096471	s__Anaerostipes hadrus
GUT_GENOME096475	s__Bifidobacterium bifidum
GUT_GENOME096477	s__Bifidobacterium breve
GUT_GENOME096488	s__Parabacteroides distasonis
GUT_GENOME096492	s__Bacteroides xylanisolvens
GUT_GENOME096515	s__Anaerostipes hadrus
GUT_GENOME096516	s__Anaerostipes hadrus
GUT_GENOME096517	s__Anaerostipes hadrus
GUT_GENOME096520	s__Lactiplantibacillus plantarum
GUT_GENOME096552	s__Blautia_A wexlerae
GUT_GENOME103690	s__Bifidobacterium breve
GUT_GENOME103700	s__Dorea formicigenerans
GUT_GENOME103702	s__Longicatena caecimuris
GUT_GENOME103720	s__Sarcina perfringens
GUT_GENOME103721	s__Enterococcus faecalis
GUT_GENOME103722	s__Enterococcus faecalis
GUT_GENOME103742	s__Coprococcus eutactus_A
GUT_GENOME103745	s__Parabacteroides distasonis

GUT_GENOME103755	s__Akkermansia muciniphila
GUT_GENOME103767	s__Fusicatenibacter saccharivorans
GUT_GENOME103768	s__Fusicatenibacter saccharivorans
GUT_GENOME103769	s__Fusicatenibacter saccharivorans
GUT_GENOME103772	s__Roseburia inulinivorans
GUT_GENOME103774	s__Bariatricus comes
GUT_GENOME103775	s__Bariatricus comes
GUT_GENOME103779	s__Roseburia inulinivorans
GUT_GENOME103781	s__Lachnospira eligens_A
GUT_GENOME103782	s__Lachnospira eligens_A
GUT_GENOME103784	s__Coprococcus eutactus_A
GUT_GENOME103786	s__Coprococcus eutactus_A
GUT_GENOME103787	s__Bariatricus comes
GUT_GENOME103790	s__Coprococcus eutactus_A
GUT_GENOME103792	s__Bariatricus comes
GUT_GENOME103793	s__Coprococcus eutactus_A
GUT_GENOME103794	s__Blautia_A wexlerae
GUT_GENOME103796	s__Bacteroides caccae
GUT_GENOME103798	s__Mediterraneibacter faecis
GUT_GENOME103799	s__Blautia_A wexlerae
GUT_GENOME103801	s__Blautia_A wexlerae
GUT_GENOME103804	s__Blautia_A obeum
GUT_GENOME103806	s__Anaerobutyricum hallii
GUT_GENOME103807	s__Blautia_A wexlerae
GUT_GENOME103808	s__Blautia_A wexlerae
GUT_GENOME103809	s__Mediterraneibacter faecis
GUT_GENOME103810	s__Mediterraneibacter faecis
GUT_GENOME103811	s__Blautia_A obeum
GUT_GENOME103814	s__Anaerobutyricum hallii
GUT_GENOME103819	s__Anaerostipes hadrus
GUT_GENOME103820	s__Anaerostipes hadrus
GUT_GENOME103821	s__Anaerostipes hadrus
GUT_GENOME103824	s__Anaerostipes hadrus
GUT_GENOME103848	s__Dorea formicigenerans
GUT_GENOME103869	s__Roseburia inulinivorans
GUT_GENOME103872	s__Bacteroides cellulosilyticus
GUT_GENOME103873	s__Anaerobutyricum hallii
GUT_GENOME103876	s__Parabacteroides distasonis
GUT_GENOME103881	s__Parabacteroides distasonis
GUT_GENOME103888	s__Anaerostipes hadrus
GUT_GENOME103894	s__Enterobacter hormaechei_A

GUT_GENOME103895	s__Agathobacter rectalis
GUT_GENOME140036	s__Parabacteroides distasonis
GUT_GENOME140040	s__Bacteroides ovatus
GUT_GENOME140052	s__Faecalibacillus intestinalis
GUT_GENOME140056	s__Bacteroides caccae
GUT_GENOME140059	s__Bacteroides caccae
GUT_GENOME140060	s__Bacteroides thetaiotaomicron
GUT_GENOME140061	s__Bacteroides ovatus
GUT_GENOME140063	s__Fusicatenibacter saccharivorans
GUT_GENOME140064	s__Blautia_A obeum
GUT_GENOME140070	s__Faecalibacillus intestinalis
GUT_GENOME140079	s__Parabacteroides distasonis
GUT_GENOME140080	s__Parabacteroides distasonis
GUT_GENOME140083	s__Thomasclavelia ramosa
GUT_GENOME140084	s__Bifidobacterium adolescentis
GUT_GENOME140085	s__Bifidobacterium adolescentis
GUT_GENOME140086	s__Bacteroides fragilis
GUT_GENOME140087	s__Bacteroides ovatus
GUT_GENOME140089	s__Bacteroides fragilis
GUT_GENOME140091	s__Bacteroides thetaiotaomicron
GUT_GENOME140094	s__Mediterraneibacter faecis
GUT_GENOME140100	s__Agathobaculum butyriciproducens
GUT_GENOME140101	s__Fusicatenibacter saccharivorans
GUT_GENOME140102	s__Agathobaculum butyriciproducens
GUT_GENOME140107	s__Longicatena caecimuris
GUT_GENOME140109	s__Agathobaculum butyriciproducens
GUT_GENOME140110	s__Fusicatenibacter saccharivorans
GUT_GENOME140113	s__Clostridium_Q fessum
GUT_GENOME140114	s__Agathobaculum butyriciproducens
GUT_GENOME140229	s__Bacteroides fragilis
GUT_GENOME140231	s__Dorea formicigenerans
GUT_GENOME140232	s__Parabacteroides distasonis
GUT_GENOME140234	s__Blautia_A wexlerae
GUT_GENOME140238	s__Anaerostipes hadrus
GUT_GENOME140252	s__Bacteroides fragilis
GUT_GENOME140254	s__RUG115 sp900066395
GUT_GENOME140255	s__Dorea formicigenerans
GUT_GENOME140257	s__Blautia_A obeum
GUT_GENOME140258	s__RUG115 sp900066395
GUT_GENOME140260	s__RUG115 sp900066395
GUT_GENOME140261	s__RUG115 sp900066395

GUT_GENOME140267	s__RUG115 sp900066395
GUT_GENOME140271	s__Mediterraneibacter faecis
GUT_GENOME140275	s__Clostridium_Q fessum
GUT_GENOME140279	s__Parabacteroides merdae
GUT_GENOME140285	s__Bacteroides caccae
GUT_GENOME140288	s__Agathobacter rectalis
GUT_GENOME140289	s__Clostridium_Q fessum
GUT_GENOME140290	s__Agathobacter rectalis
GUT_GENOME140292	s__Fusicatenibacter saccharivorans
GUT_GENOME140294	s__Blautia_A wexlerae
GUT_GENOME140295	s__Agathobacter rectalis
GUT_GENOME140298	s__Blautia_A obeum
GUT_GENOME140301	s__Fusicatenibacter saccharivorans
GUT_GENOME140303	s__Parabacteroides merdae
GUT_GENOME140305	s__Bacteroides fragilis
GUT_GENOME140306	s__Coprococcus eutactus_A
GUT_GENOME140308	s__Bacteroides fragilis
GUT_GENOME140310	s__Coprococcus eutactus_A
GUT_GENOME140312	s__Thomasclavelia ramosa
GUT_GENOME140313	s__Agathobaculum butyriciproducens
GUT_GENOME140317	s__Collinsella sp003466125
GUT_GENOME140318	s__Agathobacter rectalis
GUT_GENOME140319	s__Ruminococcus_D bicirculans
GUT_GENOME140323	s__Parabacteroides distasonis
GUT_GENOME140325	s__Mediterraneibacter faecis
GUT_GENOME140331	s__Agathobacter rectalis
GUT_GENOME140332	s__Mediterraneibacter faecis
GUT_GENOME140339	s__Faecalibacillus intestinalis
GUT_GENOME140344	s__Enterococcus faecalis
GUT_GENOME140346	s__Blautia_A sp003471165
GUT_GENOME140384	s__Bifidobacterium breve
GUT_GENOME140607	s__Blautia_A wexlerae
GUT_GENOME140721	s__Enterococcus faecalis
GUT_GENOME140722	s__Enterococcus faecalis
GUT_GENOME140724	s__Enterococcus faecalis
GUT_GENOME140725	s__Enterococcus faecalis
GUT_GENOME140726	s__Enterococcus faecalis
GUT_GENOME140786	s__Ruminococcus_D bicirculans
GUT_GENOME140790	s__Enterococcus faecalis
GUT_GENOME140791	s__Enterococcus faecalis
GUT_GENOME140792	s__Enterococcus faecalis

GUT_GENOME140793	s__Enterococcus faecalis
GUT_GENOME141010	s__Bifidobacterium bifidum
GUT_GENOME141011	s__Bifidobacterium bifidum
GUT_GENOME141012	s__Bifidobacterium bifidum
GUT_GENOME141056	s__Thomasclavelia ramosa
GUT_GENOME141096	s__Yersinia enterocolitica
GUT_GENOME141105	s__Clostridium_F botulinum
GUT_GENOME141111	s__Lacticaseibacillus paracasei
GUT_GENOME141112	s__Lacticaseibacillus paracasei
GUT_GENOME141113	s__Lacticaseibacillus paracasei
GUT_GENOME141114	s__Lacticaseibacillus paracasei
GUT_GENOME141116	s__Yersinia enterocolitica
GUT_GENOME141117	s__Yersinia enterocolitica
GUT_GENOME141118	s__Yersinia enterocolitica
GUT_GENOME141143	s__Bacteroides cellulosilyticus
GUT_GENOME141188	s__Lactiplantibacillus plantarum
GUT_GENOME141224	s__Enterobacter hormaechei_A
GUT_GENOME141241	s__Lacticaseibacillus rhamnosus
GUT_GENOME141399	s__Enterobacter hormaechei_A
GUT_GENOME141400	s__Enterobacter hormaechei_A
GUT_GENOME141448	s__Bacteroides fragilis
GUT_GENOME141449	s__Bacteroides fragilis
GUT_GENOME141450	s__Bacteroides fragilis
GUT_GENOME141451	s__Bacteroides fragilis
GUT_GENOME141452	s__Bacteroides fragilis
GUT_GENOME141453	s__Bacteroides fragilis
GUT_GENOME141454	s__Bacteroides fragilis
GUT_GENOME141455	s__Bacteroides fragilis
GUT_GENOME141456	s__Bacteroides fragilis
GUT_GENOME141457	s__Bacteroides fragilis
GUT_GENOME141458	s__Bacteroides fragilis
GUT_GENOME141459	s__Bacteroides fragilis
GUT_GENOME141460	s__Bacteroides fragilis
GUT_GENOME141468	s__Enterococcus faecalis
GUT_GENOME141469	s__Enterococcus faecalis
GUT_GENOME141470	s__Enterococcus faecalis
GUT_GENOME141471	s__Enterococcus faecalis
GUT_GENOME141472	s__Enterococcus faecalis
GUT_GENOME141687	s__Bifidobacterium breve
GUT_GENOME141688	s__Bifidobacterium breve
GUT_GENOME141689	s__Bifidobacterium breve

GUT_GENOME141690	s__Bifidobacterium breve
GUT_GENOME141691	s__Bifidobacterium breve
GUT_GENOME141692	s__Bifidobacterium breve
GUT_GENOME141693	s__Bifidobacterium breve
GUT_GENOME141694	s__Bifidobacterium breve
GUT_GENOME141695	s__Bifidobacterium breve
GUT_GENOME141696	s__Bifidobacterium breve
GUT_GENOME141698	s__Clostridium_F botulinum
GUT_GENOME141699	s__Clostridium_F botulinum
GUT_GENOME141700	s__Clostridium_F botulinum
GUT_GENOME141701	s__Clostridium_F botulinum
GUT_GENOME141702	s__Clostridium_F botulinum
GUT_GENOME141703	s__Clostridium_F botulinum
GUT_GENOME141704	s__Clostridium_F botulinum
GUT_GENOME141710	s__Bifidobacterium breve
GUT_GENOME141711	s__Bifidobacterium breve
GUT_GENOME141712	s__Bifidobacterium breve
GUT_GENOME141713	s__Bifidobacterium breve
GUT_GENOME141719	s__Lacticaseibacillus rhamnosus
GUT_GENOME141720	s__Lacticaseibacillus rhamnosus
GUT_GENOME141721	s__Lacticaseibacillus rhamnosus
GUT_GENOME141722	s__Enterococcus faecalis
GUT_GENOME141735	s__Lactiplantibacillus plantarum
GUT_GENOME141741	s__Lacticaseibacillus rhamnosus
GUT_GENOME141742	s__Clostridium_F botulinum
GUT_GENOME142059	s__Clostridium_F botulinum
GUT_GENOME142060	s__Clostridium_F botulinum
GUT_GENOME142063	s__Clostridium_F botulinum
GUT_GENOME142064	s__Clostridium_F botulinum
GUT_GENOME142065	s__Clostridium_F botulinum
GUT_GENOME142066	s__Clostridium_F botulinum
GUT_GENOME142067	s__Clostridium_F botulinum
GUT_GENOME142391	s__Sarcina perfringens
GUT_GENOME142436	s__Enterobacter hormaechei_A
GUT_GENOME142437	s__Enterobacter hormaechei_A
GUT_GENOME142438	s__Enterobacter hormaechei_A
GUT_GENOME142439	s__Enterobacter hormaechei_A
GUT_GENOME142440	s__Enterobacter hormaechei_A
GUT_GENOME142441	s__Enterobacter hormaechei_A
GUT_GENOME142442	s__Enterobacter hormaechei_A
GUT_GENOME142443	s__Enterobacter hormaechei_A

GUT_GENOME142444	s__Enterobacter hormaechei_A
GUT_GENOME142445	s__Enterobacter hormaechei_A
GUT_GENOME142446	s__Enterobacter hormaechei_A
GUT_GENOME142447	s__Enterobacter hormaechei_A
GUT_GENOME142448	s__Enterobacter hormaechei_A
GUT_GENOME142449	s__Enterobacter hormaechei_A
GUT_GENOME142450	s__Enterobacter hormaechei_A
GUT_GENOME142451	s__Enterobacter hormaechei_A
GUT_GENOME142452	s__Enterobacter hormaechei_A
GUT_GENOME142453	s__Enterobacter hormaechei_A
GUT_GENOME142454	s__Enterobacter hormaechei_A
GUT_GENOME142455	s__Enterobacter hormaechei_A
GUT_GENOME142456	s__Enterobacter hormaechei_A
GUT_GENOME142458	s__Lactiplantibacillus plantarum
GUT_GENOME142459	s__Lactiplantibacillus plantarum
GUT_GENOME142460	s__Lactiplantibacillus plantarum
GUT_GENOME142461	s__Lactiplantibacillus plantarum
GUT_GENOME142462	s__Lactiplantibacillus plantarum
GUT_GENOME142463	s__Lactiplantibacillus plantarum
GUT_GENOME142464	s__Lactiplantibacillus plantarum
GUT_GENOME142465	s__Lactiplantibacillus plantarum
GUT_GENOME142466	s__Lactiplantibacillus plantarum
GUT_GENOME142467	s__Lactiplantibacillus plantarum
GUT_GENOME142468	s__Lactiplantibacillus plantarum
GUT_GENOME142469	s__Lactiplantibacillus plantarum
GUT_GENOME142470	s__Lactiplantibacillus plantarum
GUT_GENOME142471	s__Lactiplantibacillus plantarum
GUT_GENOME142472	s__Lactiplantibacillus plantarum
GUT_GENOME142473	s__Lactiplantibacillus plantarum
GUT_GENOME142474	s__Lactiplantibacillus plantarum
GUT_GENOME142475	s__Lactiplantibacillus plantarum
GUT_GENOME142476	s__Lactiplantibacillus plantarum
GUT_GENOME142477	s__Lactiplantibacillus plantarum
GUT_GENOME142478	s__Lactiplantibacillus plantarum
GUT_GENOME142479	s__Lactiplantibacillus plantarum
GUT_GENOME142483	s__Lacticaseibacillus paracasei
GUT_GENOME142484	s__Lacticaseibacillus paracasei
GUT_GENOME142508	s__Bifidobacterium adolescentis
GUT_GENOME142509	s__Bifidobacterium adolescentis
GUT_GENOME142510	s__Bifidobacterium adolescentis
GUT_GENOME142511	s__Bifidobacterium adolescentis

GUT_GENOME142512	s__Bifidobacterium adolescentis
GUT_GENOME142513	s__Bifidobacterium adolescentis
GUT_GENOME142514	s__Bifidobacterium adolescentis
GUT_GENOME142515	s__Bifidobacterium adolescentis
GUT_GENOME142516	s__Bifidobacterium adolescentis
GUT_GENOME142517	s__Bifidobacterium adolescentis
GUT_GENOME142518	s__Bifidobacterium adolescentis
GUT_GENOME142519	s__Bifidobacterium adolescentis
GUT_GENOME142520	s__Bifidobacterium adolescentis
GUT_GENOME142521	s__Bifidobacterium adolescentis
GUT_GENOME142522	s__Bifidobacterium adolescentis
GUT_GENOME142523	s__Bifidobacterium adolescentis
GUT_GENOME142524	s__Bifidobacterium adolescentis
GUT_GENOME142525	s__Bifidobacterium bifidum
GUT_GENOME142526	s__Bifidobacterium bifidum
GUT_GENOME142527	s__Bifidobacterium bifidum
GUT_GENOME142528	s__Bifidobacterium bifidum
GUT_GENOME142529	s__Bifidobacterium bifidum
GUT_GENOME142530	s__Bifidobacterium bifidum
GUT_GENOME142531	s__Bifidobacterium bifidum
GUT_GENOME142532	s__Bifidobacterium bifidum
GUT_GENOME142533	s__Bifidobacterium bifidum
GUT_GENOME142534	s__Bifidobacterium bifidum
GUT_GENOME142535	s__Bifidobacterium bifidum
GUT_GENOME142536	s__Bifidobacterium bifidum
GUT_GENOME142537	s__Bifidobacterium bifidum
GUT_GENOME142538	s__Bifidobacterium bifidum
GUT_GENOME142539	s__Bifidobacterium bifidum
GUT_GENOME142547	s__Bifidobacterium breve
GUT_GENOME142548	s__Bifidobacterium breve
GUT_GENOME142549	s__Bifidobacterium breve
GUT_GENOME142550	s__Bifidobacterium breve
GUT_GENOME142551	s__Bifidobacterium breve
GUT_GENOME142552	s__Bifidobacterium breve
GUT_GENOME142553	s__Bifidobacterium breve
GUT_GENOME142554	s__Bifidobacterium breve
GUT_GENOME142555	s__Bifidobacterium breve
GUT_GENOME142556	s__Bifidobacterium breve
GUT_GENOME142557	s__Bifidobacterium breve
GUT_GENOME142558	s__Bifidobacterium breve
GUT_GENOME142559	s__Bifidobacterium breve

GUT_GENOME142560	s__Bifidobacterium breve
GUT_GENOME142561	s__Bifidobacterium breve
GUT_GENOME142562	s__Bifidobacterium breve
GUT_GENOME142563	s__Bifidobacterium breve
GUT_GENOME142564	s__Bifidobacterium breve
GUT_GENOME142577	s__Lacticaseibacillus rhamnosus
GUT_GENOME142584	s__Bacteroides ovatus
GUT_GENOME143130	s__Parabacteroides distasonis
GUT_GENOME143131	s__Parabacteroides distasonis
GUT_GENOME143133	s__Enterobacter hormaechei_A
GUT_GENOME143155	s__Faecalibacillus intestinalis
GUT_GENOME143157	s__Clostridium_Q fessum
GUT_GENOME143194	s__Akkermansia muciniphila
GUT_GENOME143195	s__Akkermansia muciniphila
GUT_GENOME143205	s__Akkermansia muciniphila
GUT_GENOME143206	s__Akkermansia muciniphila
GUT_GENOME143207	s__Akkermansia muciniphila
GUT_GENOME143208	s__Akkermansia muciniphila
GUT_GENOME143209	s__Akkermansia muciniphila
GUT_GENOME143210	s__Akkermansia muciniphila
GUT_GENOME143211	s__Akkermansia muciniphila
GUT_GENOME143212	s__Akkermansia muciniphila
GUT_GENOME143213	s__Akkermansia muciniphila
GUT_GENOME143214	s__Akkermansia muciniphila
GUT_GENOME143215	s__Akkermansia muciniphila
GUT_GENOME143216	s__Akkermansia muciniphila
GUT_GENOME143217	s__Akkermansia muciniphila
GUT_GENOME143218	s__Akkermansia muciniphila
GUT_GENOME143219	s__Akkermansia muciniphila
GUT_GENOME143222	s__Akkermansia muciniphila
GUT_GENOME143223	s__Akkermansia muciniphila
GUT_GENOME143224	s__Akkermansia muciniphila
GUT_GENOME143225	s__Akkermansia muciniphila
GUT_GENOME143226	s__Akkermansia muciniphila
GUT_GENOME143230	s__Anaerostipes hadrus
GUT_GENOME143231	s__Bacteroides cellulosilyticus
GUT_GENOME143232	s__Bacteroides cellulosilyticus
GUT_GENOME143233	s__Bacteroides fragilis
GUT_GENOME143336	s__Lacticaseibacillus paracasei
GUT_GENOME143337	s__Lacticaseibacillus paracasei
GUT_GENOME143338	s__Lacticaseibacillus paracasei

GUT_GENOME143339	s__Lacticaseibacillus paracasei
GUT_GENOME143355	s__Bacteroides ovatus
GUT_GENOME143356	s__Bacteroides ovatus
GUT_GENOME143357	s__Bacteroides ovatus
GUT_GENOME143358	s__Bacteroides ovatus
GUT_GENOME143415	s__Enterobacter hormaechei_A
GUT_GENOME143483	s__Bifidobacterium breve
GUT_GENOME143485	s__Bacteroides cellulosilyticus
GUT_GENOME143516	s__Agathobacter rectalis
GUT_GENOME143521	s__Ruminococcus_E bromii_B
GUT_GENOME143523	s__Ruminococcus_E bromii_B
GUT_GENOME143572	s__Bacteroides fragilis
GUT_GENOME143578	s__Blautia_A wexlerae
GUT_GENOME143591	s__Bacteroides stercoris
GUT_GENOME143592	s__Bacteroides thetaiotaomicron
GUT_GENOME143593	s__Bacteroides thetaiotaomicron
GUT_GENOME143635	s__Lacticaseibacillus paracasei
GUT_GENOME143636	s__Lacticaseibacillus rhamnosus
GUT_GENOME143637	s__Lacticaseibacillus rhamnosus
GUT_GENOME143638	s__Lacticaseibacillus rhamnosus
GUT_GENOME143639	s__Lacticaseibacillus rhamnosus
GUT_GENOME143640	s__Lacticaseibacillus rhamnosus
GUT_GENOME143641	s__Lacticaseibacillus rhamnosus
GUT_GENOME143642	s__Lacticaseibacillus rhamnosus
GUT_GENOME143643	s__Lacticaseibacillus rhamnosus
GUT_GENOME143644	s__Lacticaseibacillus rhamnosus
GUT_GENOME143645	s__Lacticaseibacillus rhamnosus
GUT_GENOME143646	s__Lacticaseibacillus rhamnosus
GUT_GENOME143647	s__Lacticaseibacillus rhamnosus
GUT_GENOME143648	s__Lacticaseibacillus rhamnosus
GUT_GENOME143649	s__Lacticaseibacillus rhamnosus
GUT_GENOME143650	s__Lacticaseibacillus rhamnosus
GUT_GENOME143651	s__Lacticaseibacillus rhamnosus
GUT_GENOME143652	s__Lacticaseibacillus rhamnosus
GUT_GENOME143653	s__Lacticaseibacillus rhamnosus
GUT_GENOME143654	s__Lacticaseibacillus rhamnosus
GUT_GENOME143655	s__Lacticaseibacillus rhamnosus
GUT_GENOME143656	s__Lacticaseibacillus rhamnosus
GUT_GENOME143659	s__Bifidobacterium bifidum
GUT_GENOME143661	s__Neisseria gonorrhoeae
GUT_GENOME143662	s__Neisseria gonorrhoeae

GUT_GENOME143663	s__Neisseria gonorrhoeae
GUT_GENOME143664	s__Neisseria gonorrhoeae
GUT_GENOME143665	s__Neisseria gonorrhoeae
GUT_GENOME143666	s__Neisseria gonorrhoeae
GUT_GENOME143667	s__Neisseria gonorrhoeae
GUT_GENOME143668	s__Neisseria gonorrhoeae
GUT_GENOME143669	s__Neisseria gonorrhoeae
GUT_GENOME143670	s__Neisseria gonorrhoeae
GUT_GENOME143671	s__Neisseria gonorrhoeae
GUT_GENOME143672	s__Neisseria gonorrhoeae
GUT_GENOME143673	s__Neisseria gonorrhoeae
GUT_GENOME143674	s__Neisseria gonorrhoeae
GUT_GENOME143675	s__Neisseria gonorrhoeae
GUT_GENOME143676	s__Neisseria gonorrhoeae
GUT_GENOME143677	s__Neisseria gonorrhoeae
GUT_GENOME143678	s__Neisseria gonorrhoeae
GUT_GENOME143679	s__Neisseria gonorrhoeae
GUT_GENOME143680	s__Neisseria gonorrhoeae
GUT_GENOME143681	s__Neisseria gonorrhoeae
GUT_GENOME143682	s__Neisseria gonorrhoeae
GUT_GENOME143683	s__Neisseria gonorrhoeae
GUT_GENOME143684	s__Neisseria gonorrhoeae
GUT_GENOME143685	s__Neisseria gonorrhoeae
GUT_GENOME143686	s__Neisseria gonorrhoeae
GUT_GENOME143687	s__Neisseria gonorrhoeae
GUT_GENOME143688	s__Neisseria gonorrhoeae
GUT_GENOME143689	s__Neisseria gonorrhoeae
GUT_GENOME143690	s__Neisseria gonorrhoeae
GUT_GENOME143691	s__Neisseria gonorrhoeae
GUT_GENOME143692	s__Neisseria gonorrhoeae
GUT_GENOME143693	s__Neisseria gonorrhoeae
GUT_GENOME143694	s__Neisseria gonorrhoeae
GUT_GENOME143695	s__Neisseria gonorrhoeae
GUT_GENOME143696	s__Neisseria gonorrhoeae
GUT_GENOME143697	s__Neisseria gonorrhoeae
GUT_GENOME143698	s__Neisseria gonorrhoeae
GUT_GENOME143699	s__Neisseria gonorrhoeae
GUT_GENOME143700	s__Neisseria gonorrhoeae
GUT_GENOME143701	s__Neisseria gonorrhoeae
GUT_GENOME143702	s__Neisseria gonorrhoeae
GUT_GENOME143703	s__Neisseria gonorrhoeae

GUT_GENOME143704	s__Neisseria gonorrhoeae
GUT_GENOME143705	s__Neisseria gonorrhoeae
GUT_GENOME143706	s__Neisseria gonorrhoeae
GUT_GENOME143707	s__Neisseria gonorrhoeae
GUT_GENOME143712	s__Agathobacter rectalis
GUT_GENOME143713	s__Agathobacter rectalis
GUT_GENOME143720	s__Lacticaseibacillus paracasei
GUT_GENOME143753	s__Enterobacter hormaechei_A
GUT_GENOME143754	s__Enterobacter hormaechei_A
GUT_GENOME143756	s__Enterobacter hormaechei_A
GUT_GENOME143757	s__Enterobacter hormaechei_A
GUT_GENOME143758	s__Enterobacter hormaechei_A
GUT_GENOME143760	s__Enterobacter hormaechei_A
GUT_GENOME143761	s__Longicatena caecimuris
GUT_GENOME143762	s__Bacteroides xylanisolvens
GUT_GENOME143767	s__Thomasclavelia ramosa
GUT_GENOME145579	s__Enterococcus faecalis
GUT_GENOME145995	s__Bifidobacterium adolescentis
GUT_GENOME147110	s__Yersinia enterocolitica
GUT_GENOME147111	s__Yersinia enterocolitica
GUT_GENOME147112	s__Yersinia enterocolitica
GUT_GENOME147113	s__Yersinia enterocolitica
GUT_GENOME147114	s__Yersinia enterocolitica
GUT_GENOME147116	s__Yersinia enterocolitica
GUT_GENOME147117	s__Yersinia enterocolitica
GUT_GENOME147118	s__Yersinia enterocolitica
GUT_GENOME147119	s__Yersinia enterocolitica
GUT_GENOME147120	s__Yersinia enterocolitica
GUT_GENOME147121	s__Yersinia enterocolitica
GUT_GENOME147122	s__Yersinia enterocolitica
GUT_GENOME147123	s__Yersinia enterocolitica
GUT_GENOME147124	s__Yersinia enterocolitica
GUT_GENOME147125	s__Yersinia enterocolitica
GUT_GENOME147126	s__Yersinia enterocolitica
GUT_GENOME147127	s__Yersinia enterocolitica
GUT_GENOME147128	s__Yersinia enterocolitica
GUT_GENOME147129	s__Yersinia enterocolitica
GUT_GENOME147135	s__Bacteroides thetaiotaomicron
GUT_GENOME147149	s__Anaerostipes hadrus
GUT_GENOME147150	s__Anaerostipes hadrus
GUT_GENOME147156	s__Bacteroides xylanisolvens

GUT_GENOME147157	s__Enterococcus faecalis
GUT_GENOME147158	s__Mediterraneibacter faecis
GUT_GENOME147162	s__Agathobacter rectalis
GUT_GENOME147163	s__Agathobacter rectalis
GUT_GENOME147165	s__Ruminococcus_E bromii_B
GUT_GENOME147170	s__Parabacteroides distasonis
GUT_GENOME147541	s__Bifidobacterium bifidum
GUT_GENOME147548	s__Enterobacter hormaechei_A
GUT_GENOME147603	s__Enterococcus faecalis
GUT_GENOME147606	s__Coprococcus eutactus_A
GUT_GENOME147631	s__Bacteroides fragilis
GUT_GENOME147632	s__Bacteroides fragilis
GUT_GENOME147633	s__Bacteroides fragilis
GUT_GENOME147634	s__Bacteroides fragilis
GUT_GENOME147635	s__Bacteroides fragilis
GUT_GENOME147636	s__Bacteroides fragilis
GUT_GENOME147637	s__Bacteroides fragilis
GUT_GENOME147638	s__Bacteroides thetaiotaomicron
GUT_GENOME147639	s__Bacteroides thetaiotaomicron
GUT_GENOME147642	s__Parabacteroides distasonis
GUT_GENOME147643	s__Parabacteroides distasonis
GUT_GENOME147659	s__Eggerthella lenta
GUT_GENOME147660	s__Eggerthella lenta
GUT_GENOME147661	s__Eggerthella lenta
GUT_GENOME147662	s__Eggerthella lenta
GUT_GENOME147663	s__Eggerthella lenta
GUT_GENOME147664	s__Eggerthella lenta
GUT_GENOME147665	s__Eggerthella lenta
GUT_GENOME147666	s__Eggerthella lenta
GUT_GENOME147667	s__Eggerthella lenta
GUT_GENOME147668	s__Eggerthella lenta
GUT_GENOME147669	s__Eggerthella lenta
GUT_GENOME147670	s__Eggerthella lenta
GUT_GENOME147671	s__Eggerthella lenta
GUT_GENOME147672	s__Eggerthella lenta
GUT_GENOME147673	s__Eggerthella lenta
GUT_GENOME147674	s__Eggerthella lenta
GUT_GENOME147675	s__Eggerthella lenta
GUT_GENOME147676	s__Eggerthella lenta
GUT_GENOME147784	s__Bifidobacterium breve
GUT_GENOME147854	s__Bacteroides caccae

GUT_GENOME147855	s__Bacteroides cellulosilyticus
GUT_GENOME147860	s__Bacteroides fragilis
GUT_GENOME147861	s__Bacteroides fragilis
GUT_GENOME147862	s__Bacteroides fragilis
GUT_GENOME147863	s__Bacteroides fragilis
GUT_GENOME147864	s__Bacteroides fragilis
GUT_GENOME147866	s__Bacteroides ovatus
GUT_GENOME147867	s__Bacteroides ovatus
GUT_GENOME147872	s__Parabacteroides distasonis
GUT_GENOME147873	s__Parabacteroides distasonis
GUT_GENOME147876	s__Parabacteroides merdae
GUT_GENOME147877	s__Parabacteroides merdae
GUT_GENOME239653	s__Ruminococcus_B gnavus
GUT_GENOME239656	s__Bacteroides stercoris
GUT_GENOME239662	s__Bacteroides stercoris
GUT_GENOME239667	s__Bifidobacterium bifidum
GUT_GENOME239668	s__Eggerthella lenta
GUT_GENOME239669	s__Longicatena caecimuris
GUT_GENOME239671	s__Bacteroides fragilis
GUT_GENOME239672	s__Bifidobacterium bifidum
GUT_GENOME239675	s__Bifidobacterium adolescentis
GUT_GENOME239677	s__Bifidobacterium bifidum
GUT_GENOME239678	s__Bifidobacterium bifidum
GUT_GENOME239679	s__Clostridium_Q fessum
GUT_GENOME239680	s__Blautia_A wexlerae
GUT_GENOME239681	s__Bifidobacterium adolescentis
GUT_GENOME239682	s__Bifidobacterium adolescentis
GUT_GENOME239691	s__Bifidobacterium bifidum
GUT_GENOME239707	s__Mediterraneibacter faecis
GUT_GENOME239708	s__Blautia_A obeum
GUT_GENOME239711	s__Collinsella sp003466125
GUT_GENOME239712	s__Bacteroides fragilis
GUT_GENOME239716	s__Dorea formicigenerans
GUT_GENOME239718	s__Parabacteroides distasonis
GUT_GENOME239720	s__Faecalibacillus intestinalis
GUT_GENOME239722	s__Mediterraneibacter faecis
GUT_GENOME239723	s__Fusicatenibacter saccharivorans
GUT_GENOME239724	s__Bacteroides xylanisolvens
GUT_GENOME239727	s__Parabacteroides distasonis
GUT_GENOME239729	s__Mediterraneibacter faecis
GUT_GENOME239730	s__Anaerostipes hadrus

GUT_GENOME239732	s__Dorea formicigenerans
GUT_GENOME239733	s__Clostridium_Q fessum
GUT_GENOME239734	s__Coprococcus eutactus_A
GUT_GENOME239735	s__Agathobacter rectalis
GUT_GENOME239736	s__Blautia_A wexlerae
GUT_GENOME239740	s__Parabacteroides distasonis
GUT_GENOME239741	s__Fusicatenibacter saccharivorans
GUT_GENOME239742	s__Longicatena caecimuris
GUT_GENOME239743	s__Ruminococcus_E bromii_B
GUT_GENOME239744	s__Dorea formicigenerans
GUT_GENOME239746	s__Ruminococcus_D bicirculans
GUT_GENOME239748	s__Fusicatenibacter saccharivorans
GUT_GENOME239752	s__Bifidobacterium bifidum
GUT_GENOME239753	s__Bifidobacterium bifidum
GUT_GENOME239756	s__Bifidobacterium bifidum
GUT_GENOME239758	s__Ruminococcus_B gnavus
GUT_GENOME239767	s__Bifidobacterium bifidum
GUT_GENOME239771	s__Collinsella sp003466125
GUT_GENOME239773	s__Dorea formicigenerans
GUT_GENOME239774	s__Bifidobacterium bifidum
GUT_GENOME239777	s__Bacteroides stercoris
GUT_GENOME239783	s__Bifidobacterium adolescentis
GUT_GENOME239784	s__Bifidobacterium adolescentis
GUT_GENOME239785	s__Bifidobacterium bifidum
GUT_GENOME239788	s__Bariatricus comes
GUT_GENOME239791	s__Bacteroides ovatus
GUT_GENOME239792	s__Ruminococcus_B gnavus
GUT_GENOME239794	s__Bifidobacterium bifidum
GUT_GENOME239796	s__Bacteroides fragilis
GUT_GENOME239802	s__Blautia_A obeum
GUT_GENOME239803	s__Bacteroides xylanisolvens
GUT_GENOME239804	s__Dorea formicigenerans
GUT_GENOME239805	s__Ruminococcus_E bromii_B
GUT_GENOME239806	s__Dorea formicigenerans
GUT_GENOME239807	s__Blautia_A wexlerae
GUT_GENOME239808	s__Ruminococcus_E bromii_B
GUT_GENOME239809	s__Faecalibacillus intestinalis
GUT_GENOME239810	s__Agathobaculum butyriciproducens
GUT_GENOME239812	s__Anaerobutyricum hallii
GUT_GENOME239814	s__Ruminococcus_E bromii_B
GUT_GENOME239815	s__Collinsella sp003466125

GUT_GENOME239816	s__Fusicatenibacter saccharivorans
GUT_GENOME239817	s__Agathobacter rectalis
GUT_GENOME239819	s__Blautia_A sp003471165

A.1.2 Varkit

```
                G C G T G T T T G T C A C G T + C =
              1001101110111111110110100011011 « 2 =
            100110111011111110110100011011 00
          & 001111111111111111111111111111111 =
            000110111011111110110100011011 00 & 01
              011011101111111011010001101101
               C G T G T T T G T C A C G T C
```
(A.1)

A.1.3 Protal

Table A.6: All genomes that are part of the MSSS200R dataset and their GTDB r214 species annotations generated with GTDB-tk 2.32. Genomes were downloaded from MGnify Genomes v2.0 (ftp://ftp.ebi.ac.uk/pub/databases/metagenomics/mgnify_genomes/human-gut/v2.0/).

Genome	Species
MGYG000228797	s__Acutalibacter sp900759575
MGYG000001950	s__Gastranaerophilus phascolarctosicola
MGYG000003796	s__Peptoniphilus_B sp000478985
MGYG000002246	s__UBA6857 sp900555805
MGYG000003156	s__Eubacterium_R sp900544515
MGYG000000768	s__Collinsella sp900542905
MGYG000002978	s__Collinsella sp900548515
MGYG000003718	s__S5-A14a sp900553025
MGYG000002734	s__Collinsella sp900545165
MGYG000003432	s__UBA6857 sp900767015
MGYG000003419	s__RF16 sp900766775
MGYG000002829	s__Enterocloster sp900551225
MGYG000089933	s__CAG-313 sp900760745
MGYG000001602	s__Blautia sp900547685
MGYG000228316	s__HGM11417 sp900761895
MGYG000003748	s__Alterileibacterium massiliense
MGYG000030438	s__Kluyvera sp902363335
MGYG000289070	s__Mailhella sp900553065
MGYG000003512	s__UBA11524 sp900769075

MGYG000003062	s__Collinsella sp900550185
MGYG000067591	s__Collinsella sp900541065
MGYG000035843	s__Anaerosporomusa sp900542835
MGYG000002187	s__Collinsella sp000434535
MGYG000002891	s__Streptococcus sp900546335
MGYG000210301	s__Gemella sp900766305
MGYG000001644	s__UBA7173 sp900759895
MGYG000173278	s__Streptococcus mitis_D
MGYG000002764	s__Collinsella sp900555745
MGYG000003593	s__Enterocloster sp900770345
MGYG000001190	s__Collinsella sp900759435
MGYG000003398.1	s__HGM08974 sp900766555
MGYG000000785	s__Lancefieldella sp900555335
MGYG000002957	s__Ellagibacter sp900554945
MGYG000003533	s__SFTJ01 sp900769345
MGYG000000966	s__Blautia_A sp900553515
MGYG000003363	s__Bacteroides sp900766195
MGYG000001015	s__Gemmiger sp900556255
MGYG000000946	s__Collinsella sp900552685
MGYG000003523	s__CAG-475 sp900769205
MGYG000061704	s__Collinsella sp900552755
MGYG000003397.1	s__F0422 sp900766245
MGYG000003544	s__Alistipes sp900769525
MGYG000002911	s__Collinsella sp900541045
MGYG000003239	s__Coprococcus sp900761435
MGYG000003131	s__UMGS1260 sp900550105
MGYG000003779	s__Stomatobaculum sp002892395
MGYG000001124	s__Collinsella sp900757285
MGYG000209960	s__CAG-269 sp900762425
MGYG000002869	s__Collinsella sp900549535
MGYG000123327	s__Collinsella sp900551815
MGYG000000145	s__Lacrimispora sp902363735
MGYG000071947	s__Collinsella sp900762015
MGYG000003547	s__UMGS1976 sp900769535
MGYG000000555	s__Cellulosilyticum sp900556665
MGYG000003047	s__Collinsella sp900553705
MGYG000143125	s__HGM12957 sp900760695
MGYG000002927	s__Collinsella sp900554585
MGYG000002241	s__UMGS1901 sp900556135
MGYG000249426	s__KA00274 sp902373515
MGYG000000906	s__Phocaeicola sp900552075

MGYG000002915	s__Collinsella sp900548075
MGYG000002877	s__Prevotella sp900550365
MGYG000016339	s__Collinsella sp900542125
MGYG000186436	s__GCA-900066495 sp902362365
MGYG000002824	s__Collinsella sp900552705
MGYG000003345	s__Veillonella sp900765235
MGYG000218377	s__Collinsella sp900543605
MGYG000002830	s__Veillonella sp900552715
MGYG000001215	s__Collinsella sp900759045
MGYG000002752	s__Collinsella sp900556605
MGYG000212344	s__Pauljensenia sp001064145
MGYG000003646	s__UBA1259 sp900771345
MGYG000003803	s__Pantoea conspicua
MGYG000002984	s__UBA737 sp900554525
MGYG000001127	s__Collinsella sp900757385
MGYG000002988	s__Pauljensenia sp900556405
MGYG000155954	s__UMGS1370 sp900551135
MGYG000180154	s__V9D3004 sp900760345
MGYG000001082	s__Ruminococcus_E sp900755995
MGYG000003276	s__Collinsella sp900762345
MGYG000044303	s__Phascolarctobacterium_A sp900770955
MGYG000001121	s__Collinsella sp900757205
MGYG000001139	s__Collinsella sp900757595
MGYG000048288	s__Collinsella sp900545995
MGYG000003549	s__UBA1259 sp900769565
MGYG000096229	s__UMGS1601 sp900553335
MGYG000002709	s__UMGS1623 sp900553525
MGYG000035712	s__Collinsella sp900542965
MGYG000002132	s__Monoglobus sp900542675
MGYG000001000	s__UBA4716 sp900556575
MGYG000002923	s__Collinsella sp900547805
MGYG000003201	s__CAG-873 sp900759825
MGYG000003068	s__UMGS1388 sp900551345
MGYG000023952	s__Dialister sp900543455
MGYG000002875	s__CAG-873 sp900541865
MGYG000002847	s__Collinsella sp900554325
MGYG000003011	s__Collinsella sp900545605
MGYG000255664	s__CAG-267 sp900551865
MGYG000003274	s__Collinsella sp900762355
MGYG000003160	s__UMGS1858 sp900555705
MGYG000236063	s__Collinsella sp900542305

MGYG000003555	s__SFWF01 sp900769775
MGYG000261263	s__UMGS1783 sp900555065
MGYG000000127	s__Catenibacillus sp902363555
MGYG000003515	s__Prevotella sp900769275
MGYG000002767	s__Collinsella sp900557455
MGYG000106591	s__Paramuribaculum sp900759835
MGYG000193619	s__Veillonella sp900549805
MGYG000003132	s__Dysgonomonas sp900556485
MGYG000003202	s__UBA3263 sp900759865
MGYG000000943	s__Eubacterium_R sp900555015
MGYG000241852	s__Blautia_A sp900540785
MGYG000003472	s__Alistipes sp900768045
MGYG000003542	s__Alistipes sp900769445
MGYG000126202	s__Succinivibrio sp900770725
MGYG000003209	s__Collinsella sp900760215
MGYG000003260	s__Collinsella sp900761945
MGYG000003513	s__Prevotella sp900769055
MGYG000003157	s__Collinsella sp900556515
MGYG000002137	s__Collinsella sp900556205
MGYG000259799	s__Anaerococcus sp900551095
MGYG000000115	s__Clostridium_J sp902363375
MGYG000002883	s__UMGS1477 sp900553845
MGYG000171916	s__Dysosmobacter sp900546705
MGYG000038090	s__UMGS124 sp900539345
MGYG000003027	s__Collinsella sp900550595
MGYG000001274	s__Streptococcus sp902373455
MGYG000003019	s__Alistipes sp900553175
MGYG000238610	s__Streptococcus sp900755085
MGYG000003572	s__Prevotella sp900770025
MGYG000054386	s__Veillonella_A sp900545795
MGYG000003229	s__Collinsella sp900761035
MGYG000004457	s__CAG-485 sp900767075
MGYG000000116	s__Exiguobacterium_A sp902363455
MGYG000002949	s__UMGS1251 sp900549995
MGYG000000940	s__Collinsella sp900548935
MGYG000003035	s__CAG-485 sp900542185
MGYG000003231	s__Collinsella sp900761145
MGYG000003249	s__Victivallis sp900761715
MGYG000003751	s__Peptoniphilus_A raoultii
MGYG000004731	s__Bifidobacterium italicum
MGYG000148734	s__Ruminococcus sp900752785

MGYG000003200	s__CAG-873 sp900759845
MGYG000006226	s__Ruminococcus_D sp900539835
MGYG000001663	s__Alistipes sp900760675
MGYG000003199	s__HGM05190 sp900759815
MGYG000003765	s__Prevotella bergensis
MGYG000003775	s__Ezakiella massiliensis
MGYG000000152	s__Lacrimispora sp902363835
MGYG000003606	s__HGM10611 sp900770645
MGYG000003026	s__Collinsella sp900549195
MGYG000003500	s__Butyrivibrio_A sp900768755
MGYG000004084	s__HGM10766 sp900757295
MGYG000003587	s__Ruminococcus_D sp900770285
MGYG000101484	s__UMGS693 sp900544555
MGYG000003632	s__Acinetobacter sp900771065
MGYG000075875	s__Zag111 sp900551965
MGYG000003494	s__Ruminiclostridium_E sp900768735
MGYG000000102.1	s__Terrisporobacter sp902363255
MGYG000252678	s__CAG-485 sp900760885
MGYG000003462	s__Collinsella sp900767675
MGYG000004975	s__Collinsella sp900557505
MGYG000003194	s__Paramuribaculum sp900546365
MGYG000088509	s__Collinsella sp900546105
MGYG000003443	s__UBA1067 sp900767325
MGYG000001130	s__Collinsella sp900757615
MGYG000257190	s__HGM13862 sp900760825
MGYG000003497	s__RF16 sp900768725
MGYG000005033	s__Collinsella sp900547345
MGYG000121486	s__Ruminococcus sp900761275
MGYG000001267	s__Lancefieldella sp902373375
MGYG000218052	s__Collinsella sp900556415
MGYG000017976	s__Collinsella sp900768265
MGYG000003625	s__UBA2913 sp900770895
MGYG000003203	s__Butyricimonas sp900759925
MGYG000003180	s__Collinsella sp900544065
MGYG000003241	s__Eubacterium_R sp900761545
MGYG000277149	s__Dialister sp900545785
MGYG000003234	s__Collinsella sp900761155
MGYG000001134	s__Collinsella sp900757495
MGYG000249906	s__Desulfovibrio sp900547595
MGYG000003468	s__UMGS1883 sp900768005
MGYG000003038	s__Collinsella sp900554465

MGYG000003023	s__Parasutterella sp900554375
MGYG000003436	s__HGM20899 sp900767005
MGYG000005372	s__UMGS1601 sp900545345
MGYG000248022	s__UBA7185 sp900756555
MGYG000026963	s__UMGS856 sp900760305
MGYG000003413	s__Akkermansia sp900766865
MGYG000200135	s__Collinsella sp900544425
MGYG000003827	s__51-20 sp900539605
MGYG000031695	s__Succinivibrio sp900767695
MGYG000003609	s__UBA1259 sp900770685
MGYG000241933	s__Fusobacterium_B sp900545035
MGYG000048231	s__Caproiciproducens sp900546895
MGYG000133322	s__Collinsella sp900553165
MGYG000145548	s__HGM12998 sp900756495
MGYG000003651	s__RF16 sp900767595
MGYG000003173	s__Collinsella sp900541185
MGYG000000913	s__Blautia sp900556555
MGYG000003421	s__CAG-115 sp900766795
MGYG000148963	s__Dialister sp900547785
MGYG000003218	s__HGM13862 sp900760825
MGYG000002192	s__Acutalibacter sp900543305
MGYG000043704	s__Massilistercora sp902406105
MGYG000003601	s__HGM12713 sp900770605
MGYG000003496	s__CAG-115 sp900768705
MGYG000278902	s__HGM04593 sp900770665
MGYG000003560	s__UMGS1820 sp900769795
MGYG000003183	s__Collinsella sp900541195
MGYG000001628	s__Collinsella sp900549345
MGYG000003040	s__Collinsella sp900556705
MGYG000001087	s__Anaerostipes sp900756035
MGYG000158189	s__Blautia_A sp900551465
MGYG000168959	s__Collinsella sp900541135
MGYG000283354	s__Butyricimonas sp900759925
MGYG000115019	s__Evtepia sp900758955
MGYG000077418	s__Lacrimispora sp902363735
MGYG000003589	s__Dysosmobacter sp900770295
MGYG000234251	s__Zag111 sp900551965
MGYG000243949	s__Dialister sp900543455
MGYG000002887	s__Anaerococcus sp900550345
MGYG000034979	s__Gastranaerophilus phascolarctosicola
MGYG000003034	s__Collinsella sp900554495

MGYG000002956	s__Lactobacillus kalixensis
MGYG000282934	s__HGM11788 sp900760465
MGYG000003020	s__Collinsella sp900554985
MGYG000107112	s__Peptoniphilus_B sp000478985
MGYG000001605	s__Enorma sp900538305
MGYG000002968	s__Collinsella sp900546455
MGYG000033599	s__Collinsella sp900554255
MGYG000237437	s__Acutalibacter sp900759575
MGYG000201340	s__Treponema_D sp900769325
MGYG000003559	s__HGM04593 sp900769765
MGYG000003721	s__Peptoniphilus_C urinimassiliensis
MGYG000003261	s__Collinsella sp900761995
MGYG000001123	s__Collinsella sp900757265
MGYG000041593	s__Collinsella sp900542825
MGYG000000563	s__Collinsella sp900546105
MGYG000003471	s__UBA737 sp900768035
MGYG000148170	s__Collinsella sp900556415
MGYG000003377	s__Phytobacter sp002377245
MGYG000213982	s__Collinsella sp900541065
MGYG000071663	s__Collinsella sp900541715
MGYG000075643	s__CAG-485 sp900761855
MGYG000087858	s__Paramuribaculum sp900551515
MGYG000018490	s__Bilophila sp900553145
MGYG000007854	s__UBA10677 sp900760475
MGYG000269707	s__Lawsonibacter sp900763995
MGYG000231128	s__UMGS1783 sp900555065
MGYG000003277	s__Duncaniella sp900762315
MGYG000001135	s__Collinsella sp900757465
MGYG000105919	s__QAMM01 sp900762715
MGYG000003143	s__Clostridium_J sp900548455
MGYG000188244	s__Collinsella sp900544875
MGYG000000132	s__Beduini sp902363625
MGYG000178801	s__Clostridium_Q sp900547735
MGYG000003543.1	s__HGM04593 sp900769465
MGYG000058755	s__CAAGGB01 sp900769285
MGYG000238756	s__Collinsella sp900767675
MGYG000000868	s__Collinsella sp900544425
MGYG000003168	s__Collinsella sp900552295
MGYG000101694	s__Collinsella sp900759045
MGYG000002970	s__Collinsella sp900552755
MGYG000082196	s__Collinsella sp900556205

MGYG000003028	s__Collinsella sp900551655
MGYG000003198	s__Paramuribaculum sp900759835
MGYG000004538	s__CAG-313 sp900760745
MGYG000192963	s__UBA1829 sp900760615
MGYG000079547	s__UMGS1613 sp900553395
MGYG000003756	s__Varibaculum massiliense
MGYG000000923	s__Phocaeicola sp900552645
MGYG000003159	s__Collinsella sp900556285
MGYG000000783	s__Veillonella_A sp900545795
MGYG000001064	s__Streptococcus sp900755085
MGYG000121639	s__Collinsella sp900553165
MGYG000211017	s__Collinsella sp900541795
MGYG000002195	s__Emergencia sp900551775
MGYG000188468	s__HGM11808 sp900757025
MGYG000157150	s__Ruminococcus_D sp900539835
MGYG000216838	s__Collinsella sp900760325
MGYG000072093	s__UMGS693 sp900544555
MGYG000003174	s__Collinsella sp900542125
MGYG000003538	s__UBA737 sp900769375
MGYG000003558	s__UMGS1322 sp900769815
MGYG000003339	s__Collinsella sp900765115
MGYG000003347	s__Prevotella sp900765465
MGYG000001280	s__KA00274 sp902373515
MGYG000049493	s__UMGS1649 sp900553785
MGYG000043831	s__CAG-510 sp900551115
MGYG000003029	s__Collinsella sp900543615
MGYG000285993	s__Collinsella sp900547505
MGYG000188691	s__Collinsella sp900551365
MGYG000232223	s__Stenotrophomonas maltophilia_S
MGYG000003935	s__CAG-465 sp900554875
MGYG000000978	s__Streptococcus mitis_D
MGYG000228344	s__Collinsella sp900547765
MGYG000126710	s__UBA6857 sp900555805
MGYG000021465	s__CAG-267 sp900551865
MGYG000003437	s__Eubacterium_R sp900767025
MGYG000097077	s__Collinsella sp900545165
MGYG000000551	s__Mailhella sp900553065
MGYG000003364	s__Collinsella sp900766165
MGYG000001005	s__Faecalimonas sp900550235
MGYG000057269	s__CAG-485 sp900767075
MGYG000094638	s__Paramuribaculum sp900760855

MGYG000002917	s__Collinsella sp900544725
MGYG000001681	s__Ruminococcus sp900761275
MGYG000069350	s__Dysosmobacter sp900548505
MGYG000002952	s__Bulleidia sp900554555
MGYG000002733	s__Collinsella sp900549335
MGYG000003122	s__Collinsella sp900541235
MGYG000159707	s__Succinivibrio sp900767695
MGYG000000926	s__Veillonella sp900549845
MGYG000067131	s__Angelakisella sp003453215
MGYG000036406	s__Acetatifactor sp900771995
MGYG000197282	s__Collinsella sp900549185
MGYG000172030	s__CAG-272 sp900556615
MGYG000212243	s__Collinsella sp900556365
MGYG000206548	s__Porphyromonas uenonis_A
MGYG000003054	s__Eubacterium_R sp900547915
MGYG000133403	s__Collinsella sp900539735
MGYG000224544	s__Longibaculum sp900538465
MGYG000002043	s__Collinsella sp900757235
MGYG000003191	s__Collinsella sp900547285
MGYG000001212	s__Acutalibacter sp900759575
MGYG000001126	s__Collinsella sp900757505
MGYG000003018	s__Alistipes_A sp900549685
MGYG000003521	s__Parabacteroides sp900770835
MGYG000023461	s__Collinsella sp900548935
MGYG000207674	s__UMGS1623 sp900553525
MGYG000109557	s__Collinsella sp900542825
MGYG000011925	s__Collinsella sp900551365
MGYG000000536	s__Cetobacterium_A sp900766645
MGYG000003528	s__CAAGGB01 sp900769285
MGYG000000976	s__CAG-492 sp900557195
MGYG000003031	s__Collinsella sp900545995
MGYG000001616	s__Dysosmobacter sp900544955
MGYG000094422	s__Alistipes sp900761235
MGYG000120288	s__Collinsella sp900556415
MGYG000224160	s__UMGS1783 sp900555065
MGYG000003275	s__Collinsella sp900762325
MGYG000002871	s__Collinsella sp900555815
MGYG000058319	s__Pauljensenia sp900554605
MGYG000192428	s__CAG-977 sp900768845
MGYG000266338	s__Duodenibacillus sp900762555
MGYG000176561	s__Ezakiella sp900540185

MGYG000202305	s__Collinsella sp900548565
MGYG000133083	s__Enterocloster sp900555045
MGYG000072410	s__Fusobacterium_B sp900545035
MGYG000003517	s__UBA3766 sp900769175
MGYG000031271	s__CAG-267 sp900551865
MGYG000076247	s__HGM12957 sp900760695
MGYG000098151	s__Fusobacterium sp000235465
MGYG000252410	s__Eisenbergiella sp900548905
MGYG000002184	s__Collinsella sp900553165
MGYG000265493	s__Paramuribaculum sp900546365
MGYG000014463	s__Cellulosilyticum sp900556665
MGYG000003473	s__UBA6857 sp900768075
MGYG000001238	s__Evtepia sp900758955
MGYG000131888	s__CAG-485 sp900761855
MGYG000173721	s__Collinsella sp900761145
MGYG000002971	s__Collinsella sp900541715
MGYG000003213	s__Collinsella sp900760245
MGYG000003433	s__CAG-485 sp900766975
MGYG000145353	s__KA00274 sp902373515
MGYG000253035	s__Ruminococcus_D sp900539835
MGYG000002867	s__Alistipes sp900552955
MGYG000003012	s__Blautia_A sp900540785
MGYG000057517	s__Collinsella sp900555745
MGYG000280200	s__Enorma sp900538305
MGYG000106198	s__V9D3004 sp900760345
MGYG000239726	s__Acidaminococcus sp900554515
MGYG000000905	s__Collinsella sp900555555
MGYG000162316	s__Eubacterium_R sp900547915
MGYG000000769	s__Collinsella sp900544875
MGYG000004074	s__HGM12998 sp900756495
MGYG000079924	s__Paenibacillus_A sp900766135
MGYG000281238	s__Clostridium_J sp902363375
MGYG000003640	s__Butyrivibrio_A sp900771195
MGYG000128420	s__Butyricimonas sp900759925
MGYG000001631	s__Bilophila sp900553145
MGYG000001241	s__Collinsella sp900758885
MGYG000003341	s__Collinsella sp900765185
MGYG000003391	s__Gemella sp900766305
MGYG000003036	s__Collinsella sp900542555
MGYG000125852	s__Porphyromonas uenonis_A
MGYG000261359	s__Collinsella sp900541035

MGYG000023773	s__Collinsella sp900545055
MGYG000269333	s__Collinsella sp900541795
MGYG000147056	s__Streptococcus sp900755085
MGYG000003181	s__Collinsella sp900545905
MGYG000009732	s__CAG-485 sp900767075
MGYG000002924	s__UMGS124 sp900539345
MGYG000217268	s__HGM11514 sp900757255
MGYG000003024	s__Collinsella sp900545835
MGYG000087105	s__Collinsella sp900542305
MGYG000001660	s__HGM11788 sp900760465
MGYG000001118	s__HGM11808 sp900757025
MGYG000268084	s__Fusobacterium_B sp900542625
MGYG000003890	s__Zag111 sp900551965
MGYG000002983	s__Gemella sp900555985
MGYG000000959	s__Ruminococcus sp900752785
MGYG000003104	s__UMGS1601 sp900545345
MGYG000197391	s__Dialister sp900543455
MGYG000002744	s__Collinsella sp900544865
MGYG000003368	s__Finegoldia sp900766215
MGYG000001884	s__UBA1829 sp900760615
MGYG000003221	s__OM05-12 sp900760755
MGYG000000531	s__Collinsella sp900768265
MGYG000003729	s__Peptostreptococcus sp000758885
MGYG000003534	s__Treponema_D sp900769325
MGYG000003204	s__HGM05232 sp900759955
MGYG000175538	s__UBA7173 sp900759895
MGYG000091901	s__Coprococcus sp900761435
MGYG000183191	s__Paramuribaculum sp900760855
MGYG000183674	s__HGM12998 sp900756495
MGYG000004263	s__Massilistercora sp902406105
MGYG000000308	s__Pauljensenia sp001064145
MGYG000264226	s__Phocaeicola sp900552645
MGYG000145892	s__Treponema_D sp900769325
MGYG000003617	s__Helicobacter_D sp900770765
MGYG000007433	s__Alterileibacterium massiliense
MGYG000003253	s__HGM11417 sp900761895
MGYG000003590	s__Sodaliphilus sp900770215
MGYG000004623	s__Blautia_A sp900551465
MGYG000003577	s__Anaeroplasma sp900770055
MGYG000209800	s__Collinsella sp900549335
MGYG000004905	s__UBA7488 sp002477185

MGYG000003430	s__Eubacterium_R sp900766895
MGYG000022381	s__Acutalibacter sp900543305
MGYG000003228	s__Collinsella sp900761085
MGYG000187594	s__Collinsella sp900543025
MGYG000002991	s__Collinsella sp900542305
MGYG000164345	s__Stenotrophomonas maltophilia_S
MGYG000000984	s__Collinsella sp900549215
MGYG000004422	s__Collinsella sp900762015
MGYG000002777	s__Dialister sp900545785
MGYG000002918	s__Collinsella sp900547765
MGYG000165596	s__UBA10677 sp900760475
MGYG000254498	s__Collinsella sp900542555
MGYG000000110	s__Kluyvera sp902363335
MGYG000002928	s__Collinsella sp900554645
MGYG000000904	s__Collinsella sp900553415
MGYG000000921	s__Dysosmobacter sp900546705
MGYG000160954	s__Paramuribaculum sp900551515
MGYG000003359	s__Paenibacillus_A sp900766135
MGYG000152774	s__Collinsella sp900546455
MGYG000115908	s__Collinsella sp900547805
MGYG000003032	s__Collinsella sp900543605
MGYG000003415	s__Alistipes sp900766655
MGYG000225807	s__S5-A14a sp900553025
MGYG000172310	s__RF16 sp900767595
MGYG000002907	s__Anaerosporomusa sp900542835
MGYG000052345	s__Collinsella sp900557455
MGYG000266276	s__Collinsella sp900550185
MGYG000261816	s__UMGS1388 sp900551345
MGYG000018872	s__CAG-313 sp900760745
MGYG000182304	s__Collinsella sp900552875
MGYG000002117	s__Collinsella sp900556365
MGYG000255227	s__UMGS1601 sp900553335
MGYG000154772	s__51-20 sp900539605
MGYG000178849	s__Dysosmobacter sp900544955
MGYG000042980	s__Collinsella sp900539735
MGYG000191684	s__UMGS1490 sp900548185
MGYG000003252	s__Bacteroides sp900761785
MGYG000001083	s__CAG-873 sp900755985
MGYG000184757	s__Collinsella sp900768265
MGYG000001102	s__UBA7185 sp900756555
MGYG000002977	s__Bifidobacterium vaginale_F

MGYG000002934	s__Caproiciproducens sp900546895
MGYG000094493	s__Terrisporobacter sp902363255
MGYG000124683	s__CAG-485 sp900542185
MGYG000263667	s__Alistipes sp900761235
MGYG000002140	s__CAG-510 sp900551115
MGYG000001590	s__Desulfovibrio sp900547595
MGYG000002995	s__Enterocloster sp900555045
MGYG000002792	s__Dialister sp900547785
MGYG000160679	s__CAG-465 sp900554875
MGYG000145523	s__UBA2804 sp900768635
MGYG000019977	s__Ezakiella sp900540185
MGYG000003223	s__CAG-485 sp900760885
MGYG000003006	s__Collinsella sp900551365
MGYG000164789	s__UMGS1623 sp900553525
MGYG000001595	s__Fusobacterium_B sp900545035
MGYG000132539	s__Collinsella sp900545605
MGYG000000907	s__Collinsella sp900551665
MGYG000019218	s__Collinsella sp900547505
MGYG000002751	s__Collinsella sp900551815
MGYG000084238	s__Butyricimonas sp900759925
MGYG000133934	s__Gabonibacter sp900543425
MGYG000279910	s__Monoglobus sp900542675
MGYG000057888	s__Anaeroplasma sp900767915
MGYG000003025	s__Collinsella sp900548565
MGYG000130409	s__Collinsella sp900762015
MGYG000230032	s__Intestinimonas sp900540545
MGYG000001635	s__Longibaculum sp900538465
MGYG000242150	s__CAG-465 sp900554875
MGYG000154418	s__Zag111 sp900551965
MGYG000168458	s__Gemmiger sp900556255
MGYG000067306	s__Veillonella sp900549805
MGYG000238187	s__Ruminococcus_E sp900755995
MGYG000137104	s__Collinsella sp900554255
MGYG000066886	s__HGM05190 sp900759815
MGYG000034974	s__Collinsella sp900556605
MGYG000065652	s__S5-A14a sp900553025
MGYG000033988	s__CAG-873 sp900755985
MGYG000066988	s__CAG-485 sp900760885
MGYG000102899	s__Eisenbergiella sp900548905
MGYG000092577	s__Collinsella sp900542825
MGYG000003268	s__Prevotella sp900762125

MGYG000191558	s__Eubacterium_R sp900547915
MGYG000038402	s__HGM05232 sp900759955
MGYG000129050	s__HGM12957 sp900760695
MGYG000001172	s__Fusobacterium sp000235465
MGYG000131875	s__Collinsella sp900760325
MGYG000044730	s__Collinsella sp900549195
MGYG000098124	s__QAMM01 sp003150405
MGYG000213113	s__UMGS1901 sp900556135
MGYG000161034	s__Dialister sp900547785
MGYG000058630	s__UBA10677 sp900760475
MGYG000192274	s__Collinsella sp900541715
MGYG000003414	s__Acinetobacter sp900766635
MGYG000004089	s__UMGS1783 sp900555065
MGYG000150036	s__Emergencia sp900551775
MGYG000036209	s__Collinsella sp900542305
MGYG000209519	s__Paramuribaculum sp900551515
MGYG000002239	s__UMGS1613 sp900553395
MGYG000001612	s__Acidaminococcus sp900554515
MGYG000002247	s__Acetatifactor sp900771995
MGYG000259670	s__Acutalibacter sp900543305
MGYG000169476	s__UBA7488 sp002477185
MGYG000117956	s__Collinsella sp900541025
MGYG000023376	s__Collinsella sp900761085
MGYG000002936	s__Collinsella sp900547505
MGYG000001621	s__Pauljensenia sp900554605
MGYG000003838	s__QAMM01 sp003150405
MGYG000049659	s__Blautia sp900547685
MGYG000002916	s__Collinsella sp900545055
MGYG000001209	s__Porphyromonas uenonis_A
MGYG000002682	s__Dysosmobacter sp900548505
MGYG000276158	s__Collinsella sp900541035
MGYG000000565	s__Collinsella sp900545555
MGYG000144149	s__Ruminococcus_D sp900539835
MGYG000167754	s__Gabonibacter sp900543425
MGYG000003172	s__CAG-485 sp900555915
MGYG000025550	s__Dialister sp900543455
MGYG000000996	s__Collinsella sp900547345
MGYG000003280	s__CAG-269 sp900762425
MGYG000003491	s__UBA2804 sp900768635
MGYG000198535	s__Phascolarctobacterium_A sp900770955
MGYG000003309	s__Lawsonibacter sp900763995

MGYG000105688	s__Fusobacterium_B sp900542625
MGYG000244658	s__Peptostreptococcus sp000758885
MGYG000264072	s__Collinsella sp900541795
MGYG000068798	s__Collinsella sp900541135
MGYG000190975	s__Collinsella sp900544425
MGYG000164350	s__Alistipes sp900760675
MGYG000274460	s__Collinsella sp900542825
MGYG000207557	s__Dialister sp900545785
MGYG000004208	s__CAG-485 sp900759795
MGYG000000488	s__Anaeroplasma sp900767915
MGYG000117940	s__Collinsella sp900553165
MGYG000001011	s__Anaerococcus sp900551095
MGYG000003505	s__CAG-977 sp900768845
MGYG000018441	s__Blautia_A sp900540785
MGYG000160316	s__Collinsella sp900768265
MGYG000003220	s__HGM12957 sp900760695
MGYG000001108	s__Collinsella sp900756765
MGYG000001592	s__Intestinimonas sp900540545
MGYG000263477	s__Paramuribaculum sp900760855
MGYG000003657	s__Collinsella sp900552875
MGYG000003610	s__HGM04593 sp900770665
MGYG000049946	s__Exiguobacterium_A sp902363455
MGYG000288152	s__CAG-267 sp900551865
MGYG000028308	s__Collinsella sp900541025
MGYG000232463	s__Prevotella sp900557255
MGYG000001796	s__UBA7488 sp002477185
MGYG000105123	s__CAG-485 sp900542185
MGYG000012311	s__Collinsella sp900541795
MGYG000111643	s__Dialister sp900547785
MGYG000246119	s__Collinsella sp900541035
MGYG000001654	s__UMGS856 sp900760305
MGYG000144162	s__Veillonella sp900549805
MGYG000030268	s__Dialister sp900543455
MGYG000157737	s__Collinsella sp900544865
MGYG000092339	s__Collinsella sp900545055
MGYG000202205	s__HGM10766 sp900757295
MGYG000074998	s__GCA-900066495 sp902362365
MGYG000044105	s__UMGS693 sp900544555
MGYG000053684	s__Collinsella sp900762015
MGYG000150882	s__UBA10677 sp900760475
MGYG000098119	s__Stenotrophomonas maltophilia_S

MGYG000003986	s__CAG-485 sp900761855
MGYG000015662	s__Beduini sp902363625
MGYG000266524	s__Anaeroplasma sp900767915
MGYG000258586	s__V9D3004 sp900760345
MGYG000003612	s__Succinivibrio sp900770725
MGYG000058141	s__UMGS1623 sp900553525
MGYG000000689	s__Angelakisella sp003453215
MGYG000260268	s__UMGS124 sp900539345
MGYG000004241	s__Alistipes sp900761235
MGYG000238869	s__Dysosmobacter sp900548505
MGYG000217489	s__51-20 sp900539605
MGYG000001617	s__Clostridium_Q sp900547735
MGYG000003856	s__Paramuribaculum sp900551515
MGYG000004309	s__Fusobacterium_B sp900542625
MGYG000003911	s__QAMM01 sp900762715
MGYG000253674	s__Succinivibrio sp900767695
MGYG000043480	s__Collinsella sp900757235
MGYG000060106	s__HGM10766 sp900757295
MGYG000002747	s__Collinsella sp900543025
MGYG000192328	s__Anaerococcus sp900551095
MGYG000267159	s__Ezakiella sp900540185
MGYG000267455	s__S5-A14a sp900553025
MGYG000003154	s__UMGS1490 sp900548185
MGYG000002919	s__Collinsella sp900541065
MGYG000060639	s__HGM11514 sp900757255
MGYG000197972	s__Collinsella sp900552755
MGYG000285361	s__Collinsella sp900768265
MGYG000002237	s__UMGS1649 sp900553785
MGYG000073145	s__Collinsella sp900542825
MGYG000004244	s__UMGS1601 sp900553335
MGYG000002222	s__Dialister sp900543455
MGYG000164807	s__Collinsella sp900545605
MGYG000199732	s__Collinsella sp900761145
MGYG000033656	s__Prevotella sp900557255
MGYG000033135	s__Acutalibacter sp900543305
MGYG000268139	s__Collinsella sp900541035
MGYG000061438	s__Collinsella sp900543615
MGYG000141456	s__CAG-272 sp900556615
MGYG000137905	s__Duodenibacillus sp900762555
MGYG000095038	s__Collinsella sp900544725
MGYG000283832	s__UMGS693 sp900544555

MGYG000051353	s__CAG-485 sp900759795
MGYG000002757	s__Collinsella sp900550595
MGYG000203600	s__Porphyromonas uenonis_A
MGYG000230911	s__Collinsella sp900545555
MGYG000015667	s__HGM10766 sp900757295
MGYG000166960	s__Collinsella sp900539735
MGYG000219620	s__UBA6857 sp900555805
MGYG000261927	s__Streptococcus sp900546335
MGYG000204260	s__Cellulosilyticum sp900556665
MGYG000263284	s__Duodenibacillus sp900762555
MGYG000003404	s__Lactococcus hircilactis
MGYG000174914	s__Collinsella sp900760245
MGYG000001203	s__Collinsella sp900556415
MGYG000082687	s__S5-A14a sp900553025
MGYG000145232	s__Phascolarctobacterium_A sp900770955
MGYG000166132	s__Collinsella sp900767675
MGYG000146422	s__CAG-272 sp900556615
MGYG000120333	s__Prevotella sp900557255
MGYG000287438	s__Collinsella sp900542825
MGYG000199701	s__Collinsella sp900552705
MGYG000206602	s__Collinsella sp900762015
MGYG000193707	s__UMGS1601 sp900545345
MGYG000195933	s__CAG-485 sp900761855
MGYG000065129	s__Veillonella sp900549845
MGYG000184226	s__Anaerosporomusa sp900542835
MGYG000009177	s__Collinsella sp900541795
MGYG000152275	s__Parasutterella sp900554375
MGYG000152086	s__Ezakiella sp900540185
MGYG000265341	s__KA00274 sp902373515
MGYG000251010	s__UMGS1623 sp900553525
MGYG000164523	s__HGM11514 sp900757255
MGYG000129218	s__UBA2804 sp900768635
MGYG000143271	s__Paramuribaculum sp900551515
MGYG000249113	s__Collinsella sp900541025
MGYG000001615	s__Eisenbergiella sp900548905
MGYG000160931	s__Collinsella sp900760325
MGYG000217213	s__CAG-267 sp900551865
MGYG000148196	s__51-20 sp900539605
MGYG000038918	s__Collinsella sp900544425
MGYG000117795	s__Lactococcus hircilactis
MGYG000125919	s__Prevotella sp900557255

MGYG000245837	s__Collinsella sp900554255
MGYG000114423	s__Gabonibacter sp900543425
MGYG000213562	s__Collinsella sp900762015
MGYG000001682	s__Paramuribaculum sp900760855
MGYG000249855	s__V9D3004 sp900760345
MGYG000168158	s__UMGS693 sp900544555
MGYG000059613	s__Fusobacterium_B sp900545035
MGYG000068136	s__Stenotrophomonas maltophilia_S
MGYG000252181	s__Collinsella sp900541185
MGYG000067551	s__UBA1829 sp900760615
MGYG000001608	s__Gabonibacter sp900543425
MGYG000066213	s__Zag111 sp900551965
MGYG000224300	s__Paramuribaculum sp900760855
MGYG000173930	s__Ruminococcus_D sp900539835
MGYG000201925	s__Peptostreptococcus sp000758885
MGYG000133835	s__Collinsella sp900539735
MGYG000002958	s__Collinsella sp900541795
MGYG000257648	s__Dialister sp900547785
MGYG000188993	s__Collinsella sp900548515
MGYG000039947	s__UBA7488 sp002477185
MGYG000288209	s__Dialister sp900545785
MGYG000042657	s__Collinsella sp900542825
MGYG000000865	s__Collinsella sp900554255
MGYG000030853	s__Collinsella sp900541035
MGYG000044047	s__Stenotrophomonas maltophilia_S
MGYG000082380	s__CAG-313 sp900760745
MGYG000178125	s__Collinsella sp900552875
MGYG000090047	s__Acutalibacter sp900543305
MGYG000155608	s__UMGS1783 sp900555065
MGYG000139770	s__Collinsella sp900541795
MGYG000216110	s__HGM10766 sp900757295
MGYG000003044	s__Collinsella sp900541135
MGYG000251722	s__Phascolarctobacterium_A sp900770955
MGYG000028712	s__Fusobacterium sp000235465
MGYG000118662	s__Alistipes sp900761235
MGYG000003216	s__UBA10677 sp900760475
MGYG000077463	s__Blautia_A sp900540785
MGYG000075111	s__Collinsella sp900762015
MGYG000006619	s__HGM12957 sp900760695
MGYG000267118	s__KA00274 sp902373515
MGYG000175025	s__Stenotrophomonas maltophilia_S

MGYG000285636	s__CAG-485 sp900759795
MGYG000000955	s__Collinsella sp900557505
MGYG000081389	s__Collinsella sp900541795
MGYG000204885	s__Collinsella sp900542305
MGYG000256613	s__Collinsella sp900548515
MGYG000198340	s__Dialister sp900543455
MGYG000003463	s__Succinivibrio sp900767695
MGYG000001013	s__Ezakiella sp900540185
MGYG000010386	s__Collinsella sp900541715
MGYG000000001	s__GCA-900066495 sp902362365
MGYG000097218	s__HGM11514 sp900757255
MGYG000100817	s__Collinsella sp900768265
MGYG000138418	s__CAG-485 sp900759795
MGYG000037520	s__Ruminococcus sp900752785
MGYG000266542	s__Caproiciproducens sp900546895
MGYG000185315	s__Collinsella sp900541025
MGYG000044571	s__Eisenbergiella sp900548905
MGYG000026618	s__Dialister sp900543455
MGYG000001603	s__UMGS1370 sp900551135
MGYG000072140	s__Collinsella sp900759435
MGYG000144052	s__Collinsella sp900548515
MGYG000124085	s__Dialister sp900545785
MGYG000165962	s__Cetobacterium_A sp900766645
MGYG000159915	s__Stenotrophomonas maltophilia_S
MGYG000161662	s__Collinsella sp900542825
MGYG000003007	s__Collinsella sp900541025
MGYG000195419	s__Collinsella sp900541715
MGYG000095633	s__Ruminococcus_D sp900539835
MGYG000073441	s__Collinsella sp900541795
MGYG000121383	s__Collinsella sp900539735
MGYG000006391	s__HGM11808 sp900757025
MGYG000004157	s__HGM11514 sp900757255
MGYG000012762	s__Dialister sp900545785
MGYG000067560	s__Collinsella sp900541025
MGYG000283305	s__V9D3004 sp900760345
MGYG000000717	s__Duodenibacillus sp900762555
MGYG000111314	s__Collinsella sp900762015
MGYG000003033	s__Collinsella sp900549185
MGYG000003051	s__Collinsella sp900542965
MGYG000219429	s__Dialister sp900543455
MGYG000137402	s__Collinsella sp900768265

MGYG000144214	s__Collinsella sp900541025
MGYG000119895	s__Collinsella sp900549345
MGYG000257957	s__Stenotrophomonas maltophilia_S
MGYG000038512	s__51-20 sp900539605
MGYG000142929	s__Phascolarctobacterium_A sp900770955
MGYG000000328	s__CAG-272 sp900556615
MGYG000283395	s__Acutalibacter sp900543305
MGYG000283583	s__Dialister sp900545785
MGYG000000102	s__Terrisporobacter sp902363255
MGYG000276719	s__UMGS693 sp900544555
MGYG000002947	s__Collinsella sp900539735
MGYG000133416	s__Collinsella sp900762015
MGYG000167917	s__Veillonella sp900549805
MGYG000069737	s__Dialister sp900545785
MGYG000006162	s__V9D3004 sp900760345
MGYG000061136	s__UMGS1601 sp900553335
MGYG000130176	s__Collinsella sp900542825
MGYG000042144	s__Stenotrophomonas maltophilia_S
MGYG000089739	s__Collinsella sp900547505
MGYG000174408	s__Collinsella sp900762325
MGYG000221372	s__Collinsella sp900541035
MGYG000062122	s__CAG-267 sp900551865
MGYG000244345	s__Collinsella sp900542825
MGYG000215603	s__Collinsella sp900541025
MGYG000059401	s__Collinsella sp900544725
MGYG000082372	s__Collinsella sp900768265

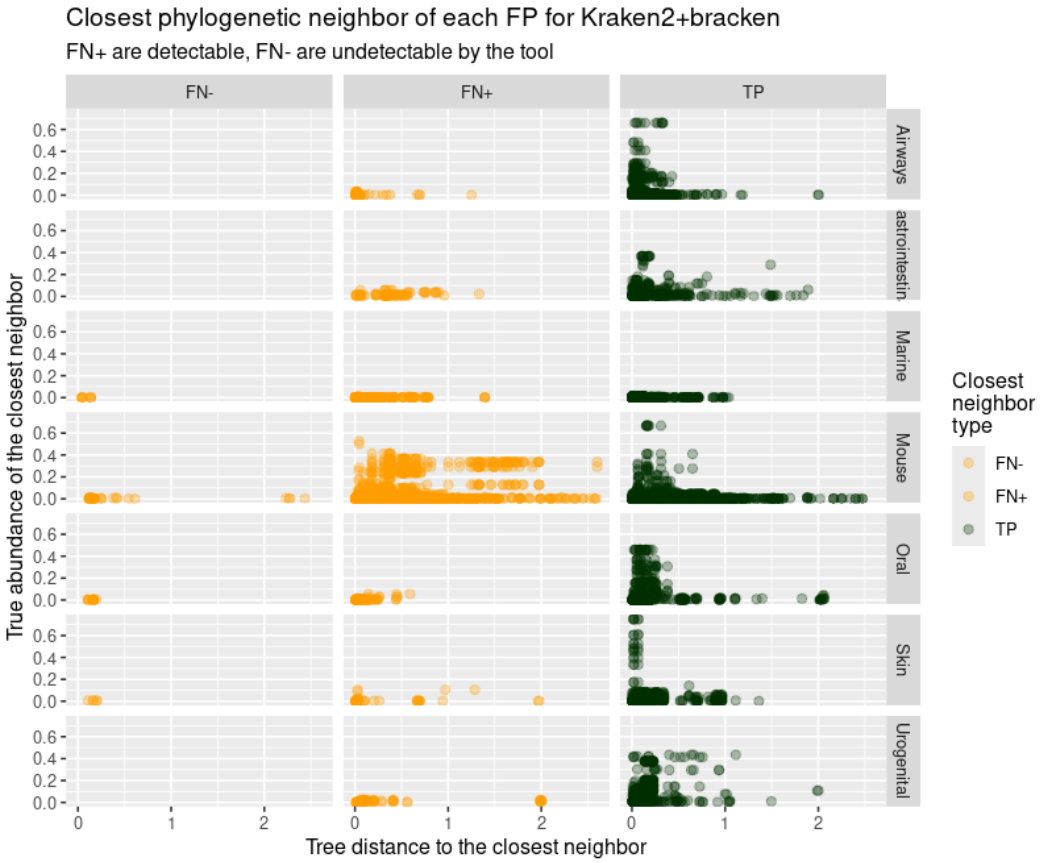

Figure A.1: Each point is a FP prediction in Kraken2+bracken, plotted with respect to the phylogenetically closest TP, FN- or FN+ in the same sample (horizontal panels). FN+ are false negatives that are contained in the taxonomic database of the tool. FN- are absent from the tool database. The x-axis shows the tree distance to the closest TP and the y-axis shows the true abundance of the TP, FN+, and FN-. TP, FP, and FN values are after re-evaluation. Vertical panels stratify datasets.

Table A.2: Species and their abbreviations used throughout the thesis.

Species	Abbreviation
s__Agathobacter_rectalis	A. rectalis
s__Agathobaculum_butyriciproducens	A. butyriciproducens
s__Akkermansia_muciniphila	A. muciniphila
s__Anaerobutyricum_hallii	A. hallii
s__Anaerostipes_hadrus	A. hadrus
s__Bacteroides_caccae	B. caccae
s__Bacteroides_cellulosilyticus	B. cellulosilyticus
s__Bacteroides_fragilis	B. fragilis
s__Bacteroides_ovatus	B. ovatus
s__Bacteroides_stercoris	B. stercoris
s__Bacteroides_thetaiotaomicron	B. thetaiotaomicron
s__Bacteroides_xylanisolvens	B. xylanisolvens
s__Bariatricus_comes	B. comes
s__Bifidobacterium_adolescentis	B. adolescentis
s__Bifidobacterium_bifidum	B. bifidum
s__Bifidobacterium_breve	B. breve
s__Blautia_A_obeum	B. A obeum
s__Blautia_A_sp003471165	B. A sp003471165
s__Blautia_A_wexlerae	B. A wexlerae
s__Clostridium_F_botulinum	C. F botulinum
s__Clostridium_Q_fessum	C. Q fessum
s__Collinsella_sp003466125	C. sp003466125
s__Coprococcus_eutactus_A	C. eutactus A
s__Dorea_formicigenerans	D. formicigenerans
s__Eggerthella_lenta	E. lenta
s__Enterobacter_hormaechei_A	E. hormaechei A
s__Enterococcus_faecalis	E. faecalis
s__Faecalibacillus_intestinalis	F. intestinalis
s__Fusicatenibacter_saccharivorans	F. saccharivorans
s__Lachnospira_eligens_A	L. eligens A
s__Lacticaseibacillus_paracasei	L. paracasei
s__Lacticaseibacillus_rhamnosus	L. rhamnosus
s__Lactiplantibacillus_plantarum	L. plantarum
s__Longicatena_caecimuris	L. caecimuris
s__Mediterraneibacter_faecis	M. faecis
s__Neisseria_gonorrhoeae	N. gonorrhoeae
s__Parabacteroides_distasonis	P. distasonis
s__Parabacteroides_merdae	P. merdae
s__Roseburia_inulinivorans	R. inulinivorans
s__RUG115_sp900066395	R. sp900066395
s__Ruminococcus_B_gnavus	R. B gnavus
s__Ruminococcus_D_bicirculans	R. D bicirculans
s__Ruminococcus_E_bromii_B	R. E bromii B
s__Sarcina_perfringens	S. perfringens
s__Thomasclavelia_ramosa	T. ramosa
s__Yersinia_enterocolitica	Y. enterocolitica

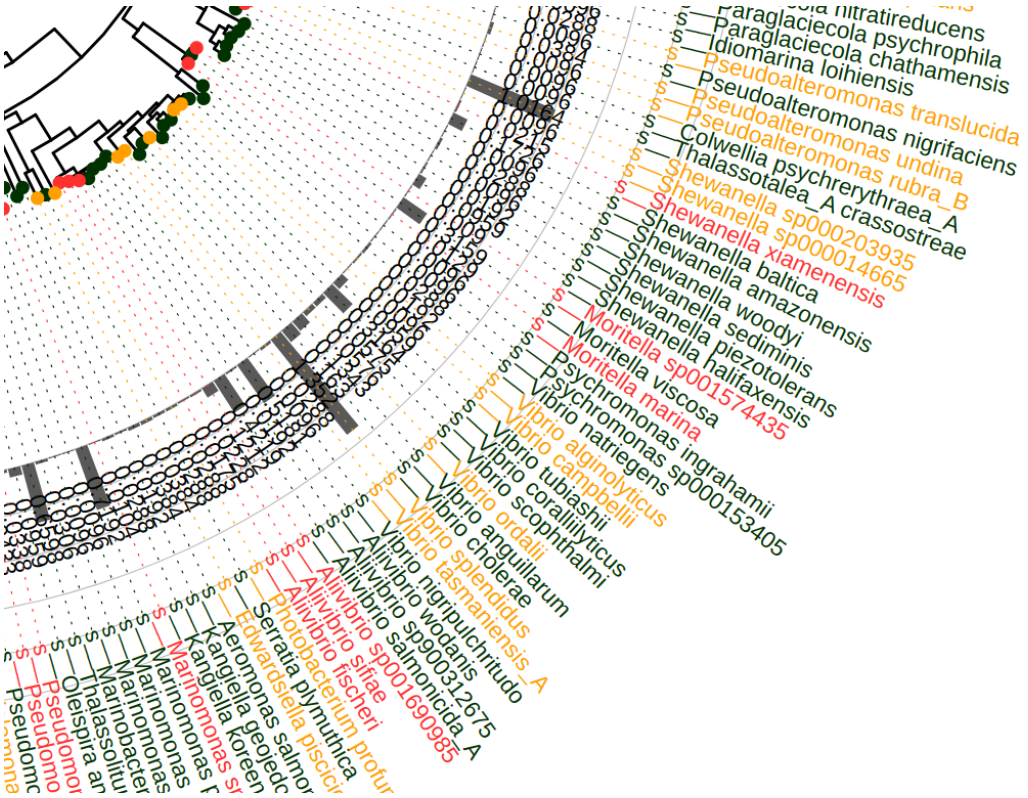

Figure A.2: Excerpt of phylogenetic tree as provided by GTDB r207, subset to TP, FP, and FN species as predicted by mOTUs3 GTDB on the dataset 3 of the CAMI Marine dataset. Red color indicates FPs, yellow indicates FN, and green indicates TPs.

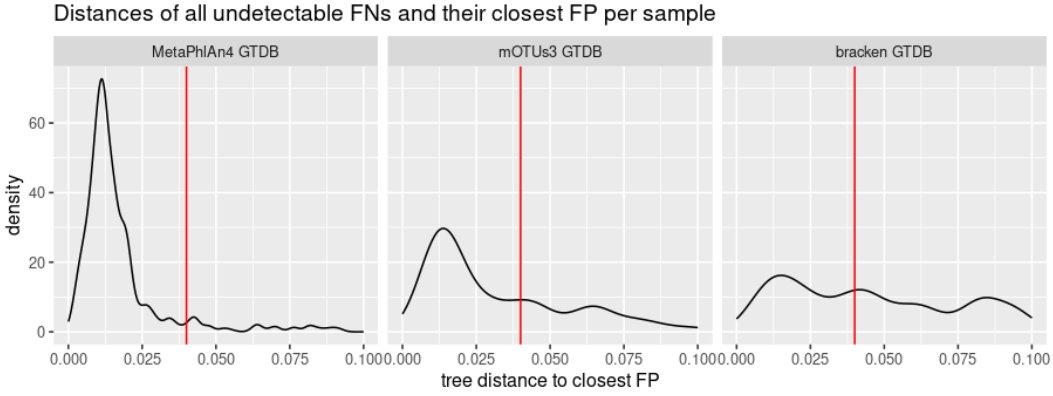

Figure A.3: Density plots showing the phylogenetic distances between undetectable false negative (FN) species—only those absent from the tool's database—and their closest false positive (FP) neighbours within the same CAMI sample (n = 107). Each panel represents a different profiling tool. The red vertical line marks the 0.04 cophenetic distance threshold used throughout this thesis to adjust binary classification metrics (F1 score, precision, sensitivity), accounting for missing taxa in tool databases. This threshold was selected to balance tolerance for taxonomic mismatches with avoiding overestimation of performance.

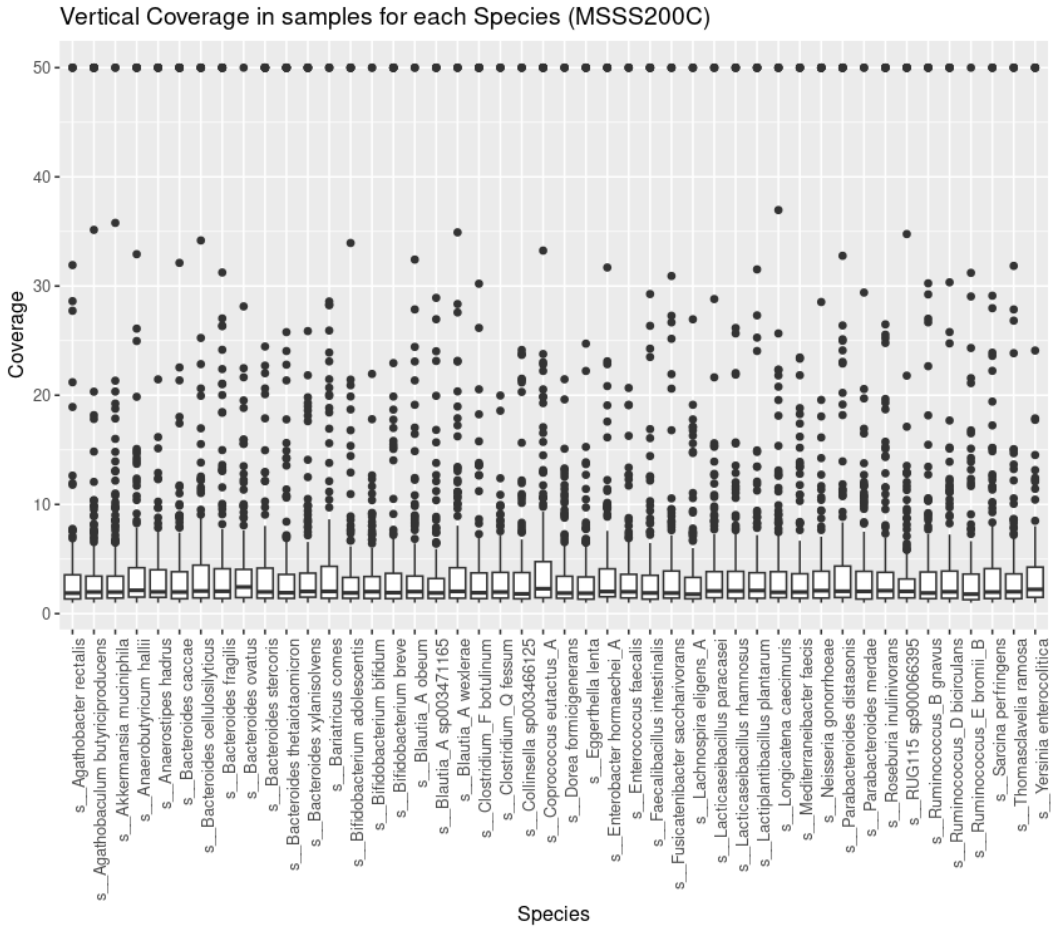

Figure A.4: Species in the MSSS200C Dataset on the x-axis and their mean vertical coverage with standard deviation across all 200 samples on the y-axis. Each species is represented in each sample with exactly one strain.

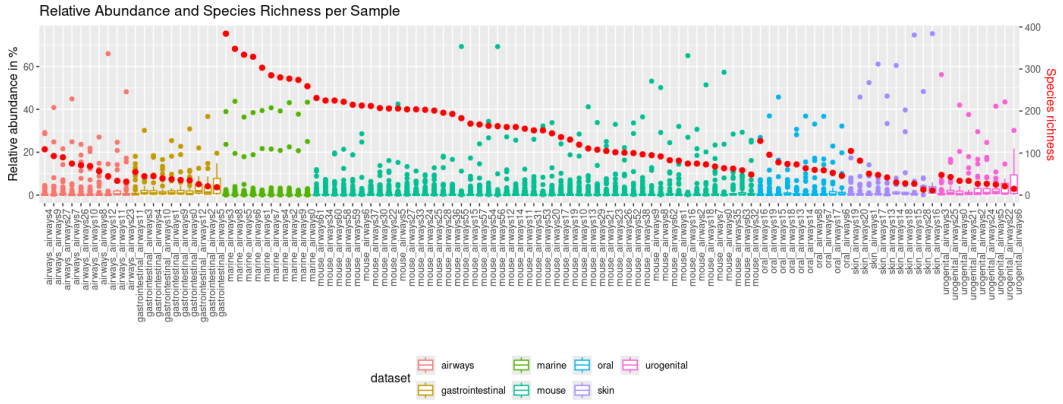

Figure A.5: Each sample from the CAMI datasets (Human, Mouse, Marine) on the x-axis with all relative abundances in % displayed in a boxplot on the left y-axis. The red dots show species richness on the right y-axis.

Table A.3: Species in the MSSS200C Dataset and their mean vertical coverage with standard deviation across all 200 samples. Each species is represented in each sample with exactly one strain.

Species	Mean	SD
s__Agathobacter rectalis	3.93	7.07
s__Agathobaculum butyriciproducens	5.08	9.94
s__Akkermansia muciniphila	4.14	6.44
s__Anaerobutyricum hallii	4.88	8.47
s__Anaerostipes hadrus	4.28	7.89
s__Bacteroides caccae	5.01	9.44
s__Bacteroides cellulosilyticus	4.95	8.55
s__Bacteroides fragilis	5.52	9.54
s__Bacteroides ovatus	4.13	5.34
s__Bacteroides stercoris	5.56	10.46
s__Bacteroides thetaiotaomicron	5.25	10.01
s__Bacteroides xylanisolvens	4.95	8.92
s__Bariatricus comes	4.85	7.85
s__Bifidobacterium adolescentis	4.41	8.38
s__Bifidobacterium bifidum	3.49	5.51
s__Bifidobacterium breve	4.66	8.80
s__Blautia_A obeum	4.07	6.57
s__Blautia_A sp003471165	4.26	7.83
s__Blautia_A wexlerae	4.78	7.66
s__Clostridium_F botulinum	4.70	8.94
s__Clostridium_Q fessum	3.74	6.34
s__Collinsella sp003466125	4.13	7.12
s__Coprococcus eutactus_A	5.69	9.36
s__Dorea formicigenerans	3.95	6.58
s__Eggerthella lenta	4.14	8.03
s__Enterobacter hormaechei_A	4.64	7.78
s__Enterococcus faecalis	3.84	6.53
s__Faecalibacillus intestinalis	4.55	8.43
s__Fusicatenibacter saccharivorans	4.17	7.24
s__Lachnospira eligens_A	3.98	6.88
s__Lacticaseibacillus paracasei	4.15	6.76
s__Lacticaseibacillus rhamnosus	4.38	7.66
s__Lactiplantibacillus plantarum	4.68	8.46
s__Longicatena caecimuris	4.75	8.14
s__Mediterraneibacter faecis	4.54	7.72
s__Neisseria gonorrhoeae	5.15	9.81
s__Parabacteroides distasonis	5.37	9.41
s__Parabacteroides merdae	3.98	6.08
s__RUG115 sp900066395	4.38	8.23
s__Roseburia inulinivorans	5.17	9.17
s__Ruminococcus_B gnavus	4.46	8.01
s__Ruminococcus_D bicirculans	4.56	7.78
s__Ruminococcus_E bromii_B	3.74	5.66
s__Sarcina perfringens	4.58	7.37
s__Thomasclavelia ramosa	4.20	7.19
s__Yersinia enterocolitica	4.29	7.30

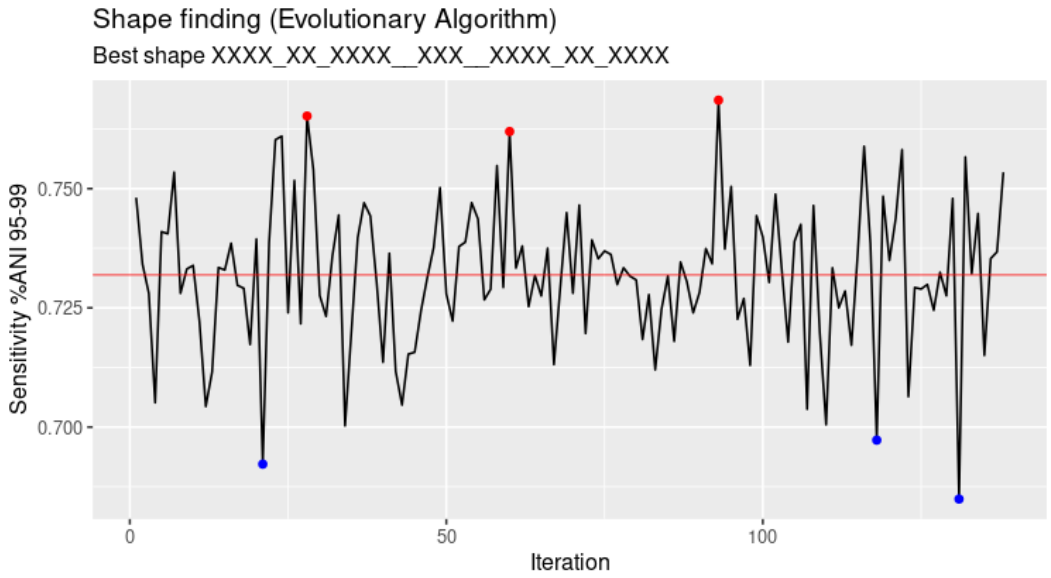

Figure A.6: In 138 iterations, each iteration tests the fitness of a new k-mer shape and then moves to the next (See 4.2.3). Fitness is defined as mean SNP sensitivity for ANIs between 95% and 99% (y-axis). The red dots are the three best performing shapes based on their fitness. From left to right these are 'X_XXXXX_XXX__XXXXX__XXX_XXXXX_X' (mean ANI of 0.7652134), 'XXXX_XX_XXXX__XXX__XXXX_XX_XXXX' (mean ANI of 0.7619734), and 'XXXX_XXX_XX_XX_X_XX_XX_XXX_XXXX' (mean ANI of 0.7685434). The blue dots mark the lowest scoring k-mer shapes and from left to right these are 'XXX____XXXXXXXXXXXXXXXXX____XXX' (mean ANI of 0.6922234),'XXXXX__XX__XXXXXXXXX__XX__XXXXX' (mean ANI of 0.6972566), and 'XXXX_X_X_X_XXXXXXXXX_X_X_X_XXXX' (mean ANI of 0.6849034).

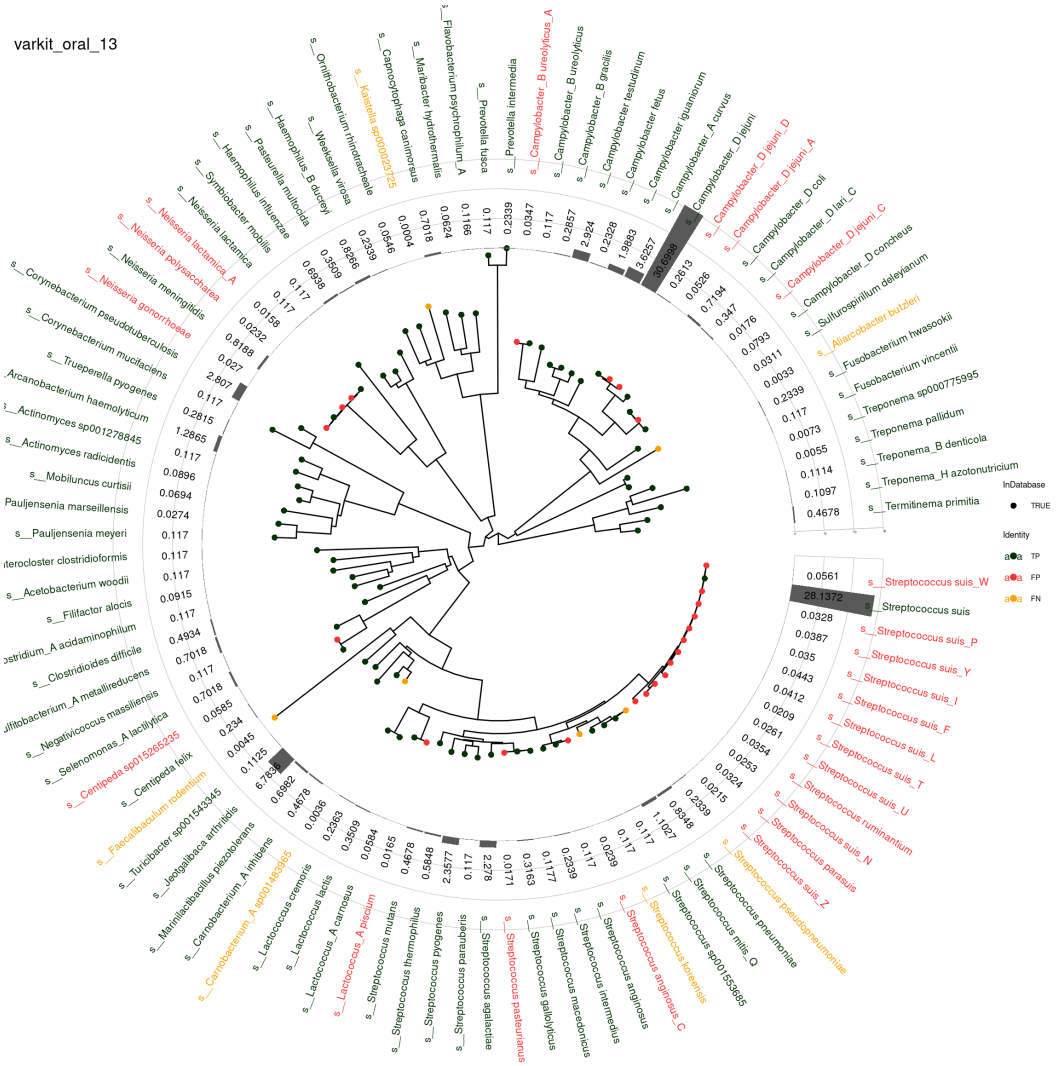

Figure A.7: Varkit species prediction profile for sample Oral 13 in the context of the phylogenetic tree of GTDB r207. Red tips are FPs, green tips are TPs, and yellow tips are FNs. The bottom right cluster of species under the genus *g__Streptococcus* are FPs likely wrong hits from the TP *s__Streptococcus suis*

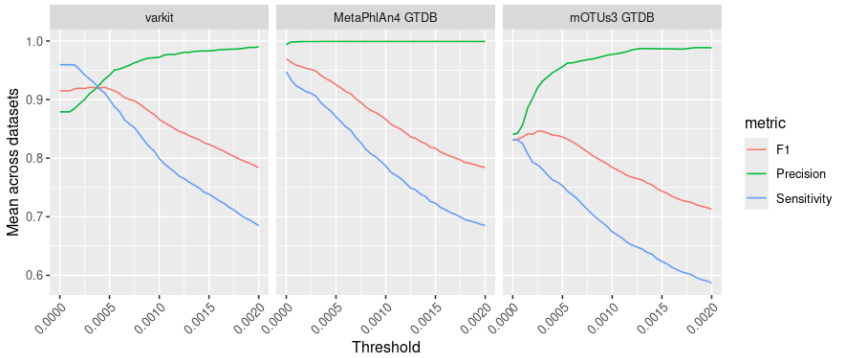

Figure A.8: Profiling performance with respect to different abundance threshold filtering.

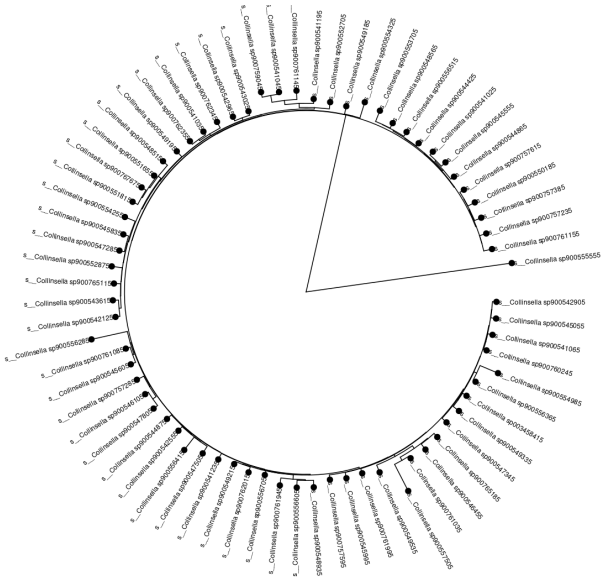

Figure A.9: Subtree of GTDB r214 species tree containing all *s__Collinsella* species from dataset MSSS200R_15.

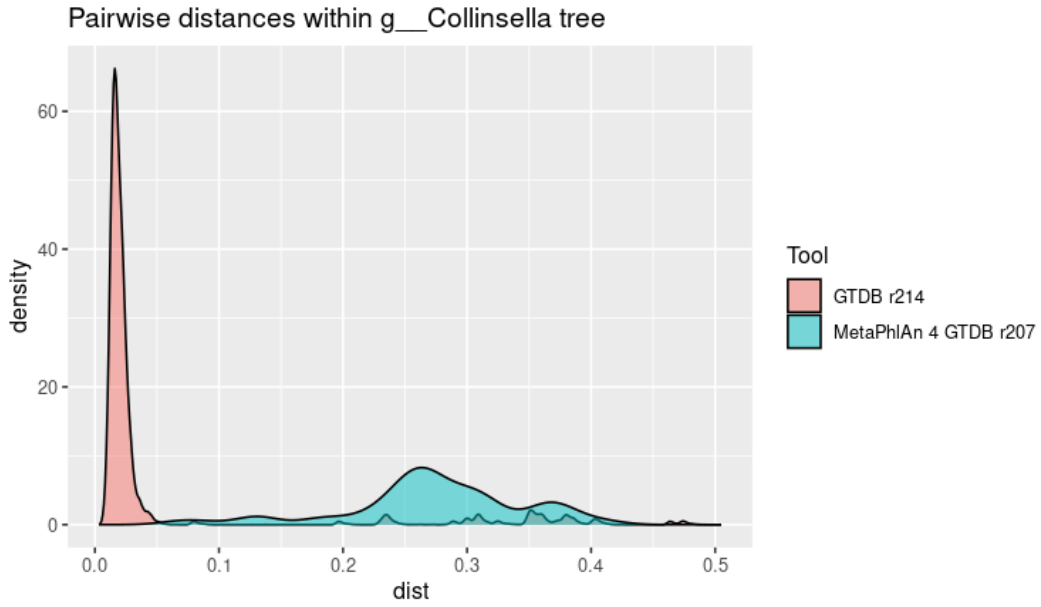

Figure A.10: Subtree of GTDB r214 species tree containing all *s__Collinsella* species within GTDB r214 and a subset of GTDB r207 containing only species covered by MetaPhlAn 4.

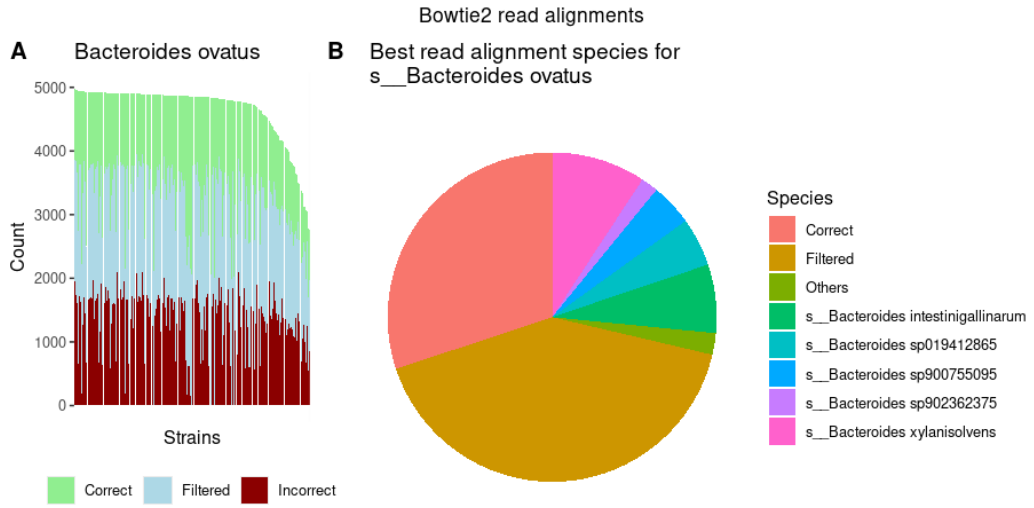

Figure A.11: Subtree of GTDB r214 species tree containing all *s__Collinsella* species within GTDB r214 and a subset of GTDB r207 containing only species covered by MetaPhlAn 4.

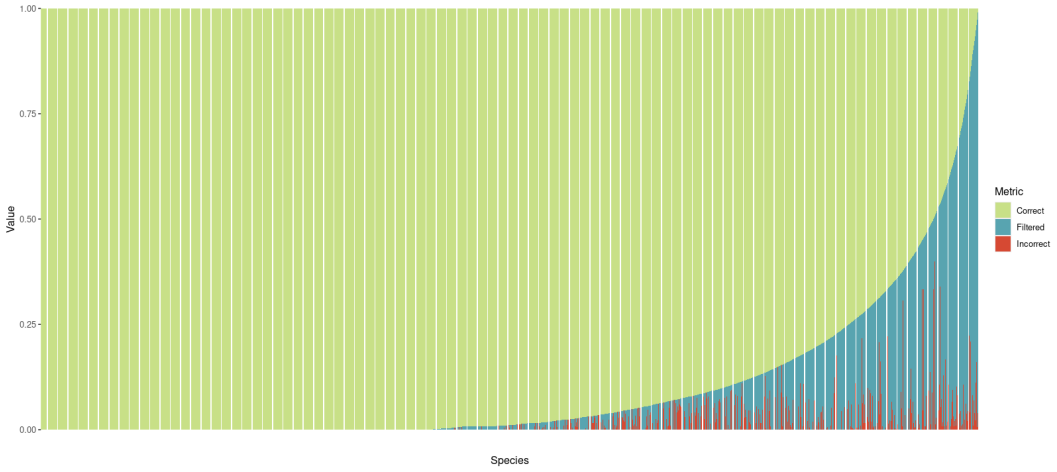

Figure A.12: Read alignments with protal for reads simulated from all marker genomes mapping outside of their original species cluster and filtering out alignments with MAPQ <4 (see Section 5.2.5 for details about data). All 80,789 are stratified on the x-axis and sorted based on the percentage of correct alignments.

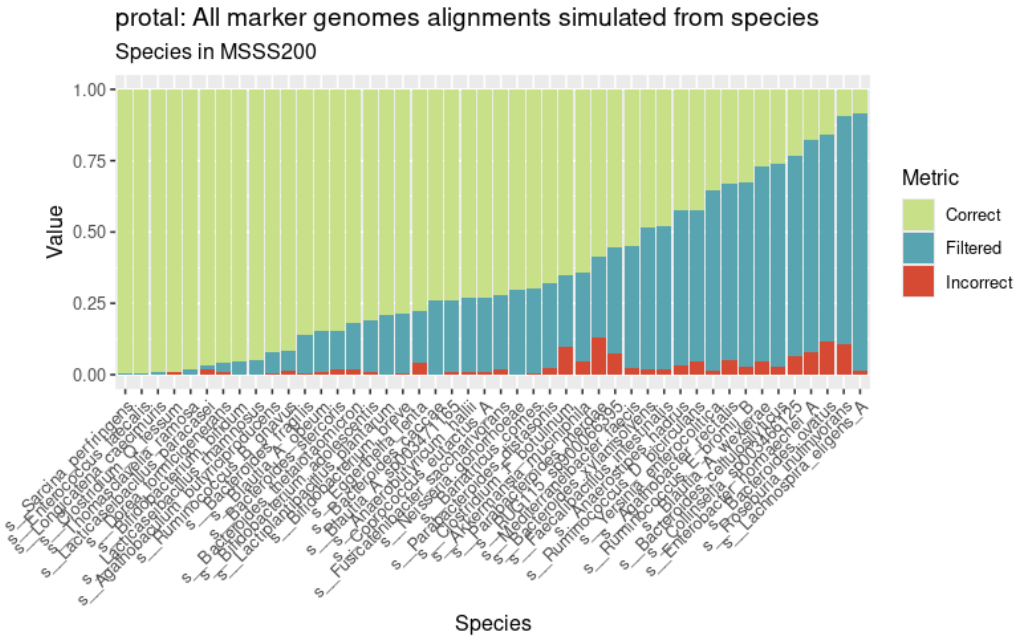

Figure A.13: Read alignments with protal for reads simulated from all marker genomes mapping outside of their original species cluster and filtering out alignments with MAPQ <4 (see Section 5.2.5 for details about data). Species contained in MSSS200 are stratified on the x-axis and (as opposed to Fig. A.14) incoming alignments are stratified based on whether they originate from the species, or not. Incorrect alignments for a species hence quantifies the amount of alignments from reads of other species.

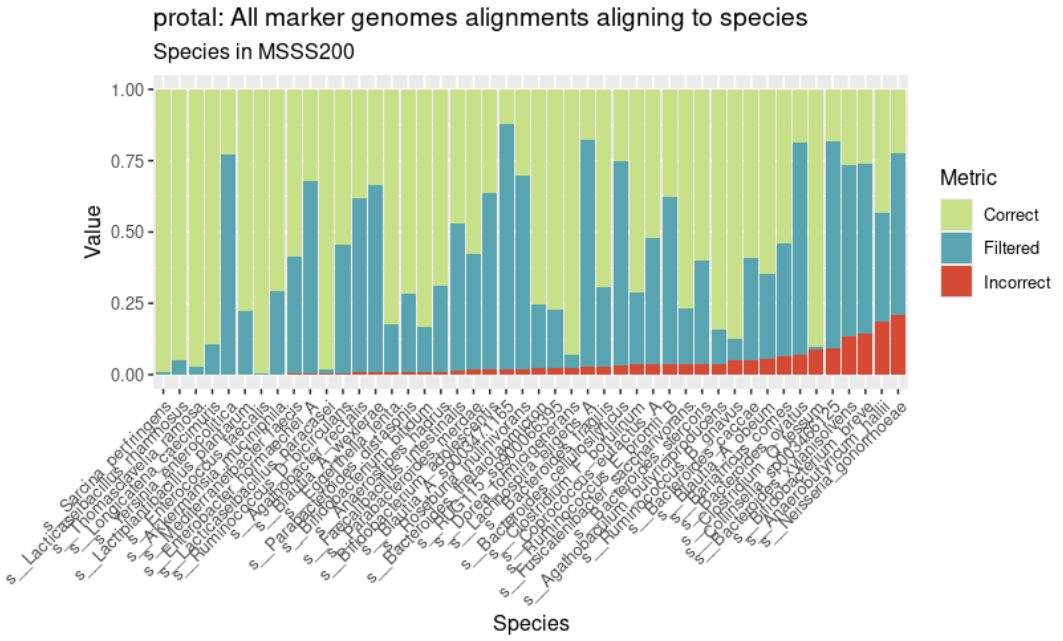

Figure A.14: Read alignments with protal for reads simulated from all marker genomes mapping outside of their original species cluster and filtering out alignments with MAPQ <4 (see Section 5.2.5 for details about data). Species contained in MSSS200 are stratified on the x-axis and sorted based on the percentage of correct alignments.

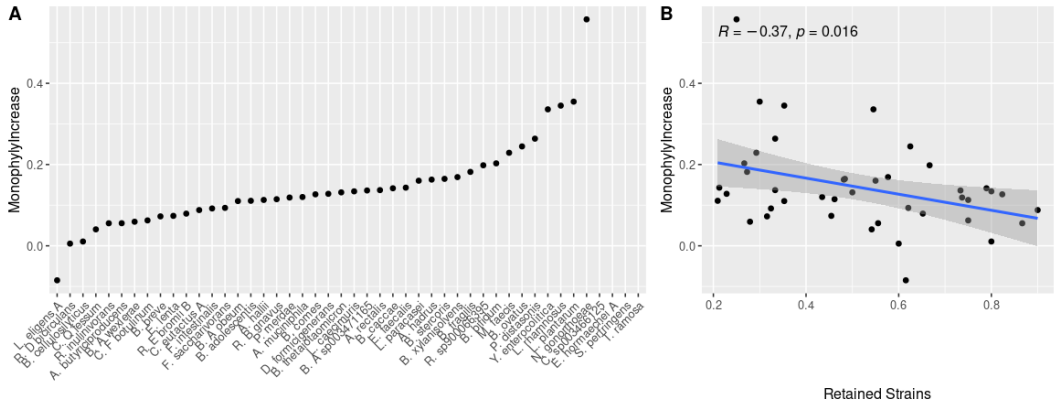

Figure A.15: A, Increase in monophyly score for protal from before to after filtering to the same samples as StrainPhlAn 4. B, Increase in monophyly score with respect to the fraction of retained strains from before to after filtering. There is a significant correlation between strain loss and increase in mean monophyly score (p=0.016, pearson correlation)

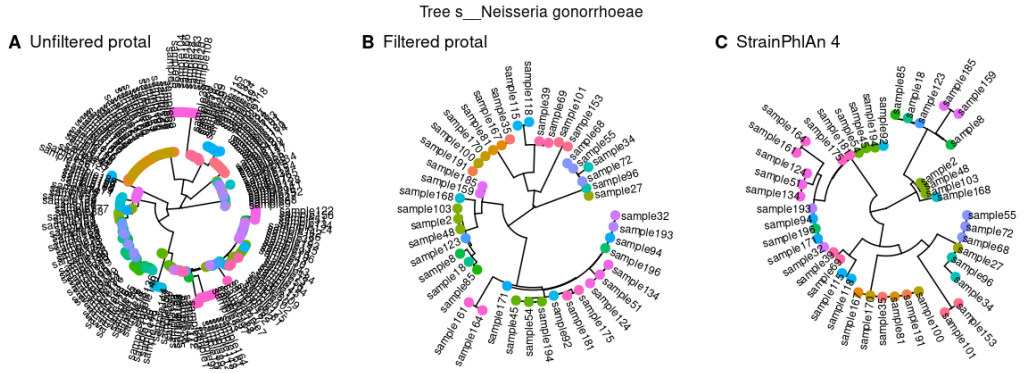

Figure A.16: Phylogenetic trees for *s__Neisseria gonorrhoeae*. A, phylogenetic tree constructed from protal's MSA. B, phylogenetic tree constructed from protal's MSA, subset to shared trees with StrainPhlAn 4. C, phylogenetic tree constructed from StrainPhlAn4's MSA.

Table A.5: Per sample mean vertical coverage (VC) and standard deviation (SD) for MSSS200R, as well as number of species.

Sample	Mean VC	SD VC	Species
sample0	1.77	6.31	200
sample1	1.68	6.28	200
sample10	1.41	5.72	200
sample11	1.67	6.17	200
sample12	1.46	5.89	200
sample13	1.56	5.91	200
sample14	1.33	5.60	200
sample15	1.29	5.53	200
sample16	1.40	5.61	200
sample17	1.63	6.09	200
sample18	1.59	6.17	200
sample19	1.71	6.43	200
sample2	1.30	5.65	200
sample3	1.31	5.64	200
sample4	1.27	5.60	200
sample5	1.50	5.95	200
sample6	1.32	5.68	200
sample7	1.38	5.60	200
sample8	1.64	6.04	200
sample9	1.64	6.20	200