Imperial College London
Faculty of Medicine
Department of Infectious Disease Epidemiology

Novel Computational Approaches for Pathogen Detection

Leonhardt Amadeus Hieronymus Unruh

Submitted in fulfilment of the requirements for the degree of
Doctor of Philosophy, October 2024

Copyright Declaration

The copyright of this thesis rests with the author. Unless otherwise indicated, its contents are licensed under a Creative Commons Attribution-Non Commercial 4.0 International license (CC BY-NC). Under this license, you may copy and redistribute the material in any medium or format. You may also create and distribute modified versions of the work. This is on the condition that: you credit the author and do not use it, or any derivative works, for a commercial purpose. When reusing or sharing this work, ensure you make the license terms clear to others by naming the license and linking to the license text. Where a work has been adapted, you should indicate that the work has been changed and describe those changes. Please seek permission from the copyright holder for uses of this work that are not included in this license or permitted under UK Copyright Law.

Statement of Originality

I hereby declare that this thesis is my own work and solely authored by me. This is to the exception on feedback and comments from my supervisor. To the best of my knowledge does not reproduce or plagiarize the work of others and no others copyright or proprietary ideas are violated within this thesis. References to others work and ideas are provided and results from joint research are indicated where applicable. Michael A. Crone and Paul S. Freemont collaborated on work on group testing and performed all in vitro parts of the research.

Abstract

Computational methods are an essential pillar in deriving information from biological data which can inform pathogen surveillance and diagnostics. Metagenomics has great potential to advance our understanding of bacteria and for surveillance and personalized medicine. However, high resolution taxonomic classification and the assignment of genes to bacterial strains still present major hurdles. For viruses, surveillance can be aided by group testing, which can greatly reduce testing resources required. Yet, no clear pathway from method development to clinical application exists and there is still much potential for theoretical developments improving the information gained from group testing. This thesis addresses the above points in two parts. First, following a benchmarking study of available tools, a novel metagenomic strain identification tool is developed. Results show state-of-the-art performance on simulated and mock metagenomic data. Additionally, accurate assignment of accessory genes to strains in metagenomes using the tool is possible. Secondly, a new group testing decoder is developed and evaluated within a novel validation framework for high-throughput group testing validation. This study shows the in vitro feasibility of group testing using a non-binary decoder. The presented decoder represents a first step towards designing a group testing decoder for multiplexed group testing, and future research avenues are discussed in detail. Finally, an exact expression for estimating prevalence from array testing data is derived. For the same number of tests, the array testing estimator proves to be more accurate than individual testing. Altogether, this thesis contributes valuable new methods for the analysis of metagenomics and group testing data. The metagenomics approach has the potential to provide new, interpretable insights into bacterial strains in microbiomes and the genes they carry. The work on group testing and prevalence estimation shows their usefulness for surveillance testing and highlights the potential of widespread implementation of group testing in public health applications.

Acknowledgements

First and foremost, I want to thank my supervisor Leonid Chindelevitch, without whom this project would not have been possible, for providing continuous support, advice, and expertise. I also want to express my gratitude towards my co-supervisor, Samir Bhatt. Michael A. Crone and Paul S. Freemont were outstanding collaborators during our group testing projects. They provided valuable advice on the intricacies of PCR testing and group testing implementation in the laboratory and conducted all of our in vitro experiments to an outstanding standard. I want to acknowledge Hooman Zabeti, Nick Dexter, Ivan Lau, and Ben Adcock from the Simon Fraser University for our fruitful discussions on group testing. Santiago Castillo Ramírez from the National Autonomous University of Mexico provided ongoing advice on the genomics of *Acinetobacter baumannii* and I want to thank him for the many useful discussions on developing models for strain-level classification of metagenomic data. My deepest gratitude goes to my parents for all their support and always believing in me. Without them, none of this would have been possible. Finally, I thank all of my friends and family for all their support throughout my PhD, academically and personally.

To my parents, Karl-Heinz and Berbel.

'The story so far:
In the beginning the Universe was created. This has made a lot of people very angry and been widely regarded as a bad move.'

– Douglas Adams

Contents

Copyright Declaration **3**

Statement of Originality **5**

Abstract **7**

Acknowledgements **9**

I Introduction **39**

Overview of Metagenomics 41

Bacterial Strains 44

Strain Identification Tools 48

Integer Linear Programming 52

Group Testing 62

Group Testing for Prevalence Estimation 69

Thesis Aims 71

II Benchmark of Strain-Level Resolution Taxonomic Classification Tools for Metagenomics **73**

Methods 75

Results 89

Discussion 103

III Core Genome based Strain Identification in Metagenomes **105**

Methods 107

Results 125

Discussion 128

IV Whole Genome based Strain Identification in Metagenomes **131**

Methods 132

Results 147

Discussion 159

V Array Testing for Prevalence Estimation **165**
Methods . . . 166
Results . . . 171
Discussion . . . 176

VI Preliminary Study on non-binary Group Testing for Respiratory Infections **179**
Methods . . . 180
Results . . . 193
Discussion and Future Work . . . 195

VII Discussion **201**
Synopsis of Results . . . 202
Metagenomics in Context . . . 203
Group Testing in Context . . . 209
Conclusion . . . 211

Bibliography **211**

A Supplementary Tables and Figures for Chapter II **237**

B Supplementary Text and Figures for Chapter IV **271**
Modifications of the Accessory-Proportion Model . . . 271
Supplementary Figures . . . 272

C Derivation of NCOMP False Positive Rate **277**

List of Tables

2.1 Overview of the evaluated strain identification tools. 75

2.2 Statistics for assemblies of the different species. Genome Length is the length of the genomes in nucleotides. N count is the number of ambiguous bases in the assemblies. # of Gaps is the number of gaps (-) in the assemblies. N50 is the shortest contig that needs to be included to cover over 50% of the genome length and N50 count the number of contigs that need to be included to reach the N50. # of Contigs is the number of contigs the assemblies consist of. # Outliers shows the number of assemblies that are more than three standard deviations away from the mean for each measure. The same assembly may be an outlier on multiple measures. Hence, some assemblies may be excluded on multiple grounds and the row totals do not add to the number of excluded assemblies for each species. 83

2.3 Parameters for metagenomic data generation. For each species 5 repeats were created for every combination of parameters in a row. See Appendix Table A1 for more details on the individual samples. 85

2.4 Results for NIST samples. True abundance indicates the percent of the sample (including all species) the strain of the species listed in the row makes up (e.g. the *A. baumannii* strain makes up a total of 2.6% of sample A). Note that percentages need not to sum to one within a sample, as more than the listed species are contained in the sample. # inferred strains column shows the number of strains for the species each tool inferred (one strain would be optimal). Mash difference indicates the difference between two Mash distances. The first Mash distance is the minimum of the distances between the inferred strains and the strain in the sample. The second is the minimum distance between the strain in the sample and all strains in the reference database. The first measures how close to the strain in the sample the estimated strain is. The second measures the Mash distance obtained by the best possible inference given the database. As such this column measures how close to the best possible result the inference is. Similarly, the WSD measures the same difference, however, this time for the WSD of the inferred population to the true strain and the best possible WSD given the database strains. In this manner, the number of strains inferred is taken into account. Bold fonts indicate the best performing model for a metric-sample-species combination. 102

3.1 Parameter settings used for model tuning. All different combinations of the listed parameter settings were used. $\lambda_s = 1$ (weight of the sparsity objective in Equations 3.3 and 3.12) and $\mathbf{K} = 3$ (maximum number of inferred strains) were used across models. Minimum z_k indicates that $z_k \geq 1$, $z_k \geq 0.9$, or $z_k \geq 0.5$ is added as a constraint. Hence, the inferred strains' allelic profile must match at reference strain to that percentage. As reminder, λ_p weighs the importance of minimizing the difference between observed and estimated nucleotide or read proportions (depending on whether a nucleotide- or allele-based model is utilized), λ_m weighs the total mismatch between reads and their assigned gene variant, and λ_d weighs the importance of the set of gene variants of the estimated strains being close to at least one reference strain. Finally, mismatch determination refers to the alignment process used prior to determination of the number of mismatches between reads and strain variants, with reads either being added to the SCCGs MSAs, or reads being aligned individually to each gene variant via pairwise alignment. 125

4.1 Parameter settings used for model tuning. $\lambda_s = 1$ (weight of the sparsity objetive) and $\gamma_{min} = 1$ (minimum nucleotide coverage) were used across models. λ_p is the weight for the objective minimizing the mismatch in observed and estimated nucleotide proportions for the SCCGs. λ_v weighs the objective of matching as many gene presence or absence observations as possible. λ_t is the weight that determines the importance of the deviation between observed and estimated proportions of accessory genes in the optimization. Finally, λ_t is the weight corresponding to the minimization of the number of genes estimated for the strains that are not carried by these strains in the database. 148

4.2 Number of core and accessory genes for the different analyzed species used in the final version of APM. 154

4.3 Results for NIST samples. True abundance indicates the percent of the sample (including all species) the strain of the species listed in the row makes up (e.g. the *A. baumannii* strain makes up a total of 2.6% of sample A). Note that percentages need not to sum to one within a sample, as more than the listed species are contained in the sample. # inferred strains column shows the number of strains for the species each tool inferred (one strain would be optimal). Mash difference indicates the difference between two Mash distances. The first Mash distance is the minimum of the distances between the inferred strains and the strain in the sample. The second is the minimum distance between the strain in the sample and all strains in the reference database. The first measures how close to the inferred strain the best inferred strain is. The second measures the Mash distance obtained by the best possible inference given the database. As such this column measures how close to the best possible result the inference is. Similarly, the WSD measures the same difference, however, this time for the WSD of the inferred population to the true strain and the best possible WSD given the database strains. In this manner, the number of strains inferred is taken into account. Bold fonts indicate the best performing model for a metric-sample-species combination. 160

A1 Details of the 840 created samples. Each row corresponds to four samples, one for *E. coli*, one for *A. baumannii*, one for *M. tuberculosis*, and one for *L. monocytogenes*, that were created with the same parameters. The sample reads column of each species indicates the total number of paired end reads in the sample. Coverage indicates the total coverage across all strains of the corresponding species in a sample. Relative abundance profile column indicates whether all strains in a sample had equal abundance (Equal) or whether one strain had 0.8 relative abundance with the remaining strains having equal parts of the remaining 0.2 relative abundance (One Dominant Strain). The same or different cluster cluster column describes whether all the strains in a sample were sampled from the same cluster at the 100 cluster level, or from different ones. The strains in reference database column indicates whether the strains in the sample were also in the reference database provided to the tools (Yes) or not (No). . . 245

List of Figures

1.1 Illustration of the simplex method for two variables x_1 and x_2, with $x_1, x_2 \geq 0$. **(A)** Constraints define an area of feasible values for x_1 and x_2. Here, each line represents a constraint and solid parts of the lines delineate the space where both x_1 and x_2 have a feasible value. **(B)** The space of feasible values stays convex when adding a new constraint. **(C)** The optimal objective value $\zeta = cx$ achievable with allowed variable values lies at a vertex. **(D)** The simplex method starting out at one vertex (lower right) and proceeding along the vertices until the optimum is reached (red dot). 54

1.2 Illustration of the branch-and-bound algorithm. 58

1.3 Illustration of the cutting plane method. Black dots represent integer valued solutions. **(A)** The optimal value of the LP relaxation (blue) is non-integer valued. **(B)** Introduction of a cutting plane (red line) making the solution to the LP relaxation infeasible, while keeping all integer valued solutions feasible. 59

1.4 Array testing illustration. Here, each cell represents one sample from an individual. Samples are grouped along rows and columns. The red cells represent positive samples, which cause their groups, marked with thick borders, to test positive. A decoding method that declares every sample positive for which all its tests are positive then identifies two true positives (TP) and two false positives (FP). 63

1.5 Illustration of d-disjunctness. On the left, the same samples as in Figure 1.4 are positive and inferred as such (labeled TP). However, we now also group samples along diagonals, increasing the disjunctness by one. This is shown for one example test, where the samples in that test are marked in blue. Similar groupings are made along the other diagonals (wrapping around such that seven individual samples are included in each test as also shown for the blue test). This diagonal test resolves the false positives (labeled FP in Figure 1.4) which are now labeled as true negatives (TN), because the diagonal test does not include any positive samples and therefore tests negative. However, we can again incur false positives. The disjunctness of the design with the diagonal tests is two. When we have three positive samples, as shown in the right illustration, we can again incur false positives. Here, the third positive sample (again inferred as a true positive and labeled TP) is contained in the diagonal test. The diagonal test will then test positive. The sample labeled FP will then also be in a positive row, a positive column, and a positive diagonal test and will be a false positive. 65

1.6 Example of PCR testing outcomes. Each curve represents a test. The Ct value of each sample is indicated by the dashed line of the same color, marking the point at which the fluorescence crosses the threshold. Units are arbitrary. 67

2.1 Country of collection for the assemblies of the different species. 'No Information' count contains all assemblies with the following labels: 'not determined', 'Not determined', 'unknown', 'Unknown', 'not collected', 'Not collected', 'missing', 'Missing', 'not applicable', 'Not applicable', 'Not Applicable', 'not available', 'not provided', 'nan'. 76

2.2 Counts of the identified MLST types for each species sorted by number of appearance. Tick labels on the x-axis indicate the strain type. Count is the number of assemblies with that strain type. Assemblies without an identified strain type are not shown. 80

2.3 Count of assemblies of *A. baumannii* assigned to each cluster at the three clustering levels (assemblies clustered into 10, 100, and 500 clusters, labeled as 10 Cluster Level, 100 Cluster Level, and 500 Cluster Level, respectively). Cluster numbers are arbitrary and numbering increases left to right on the x-axis (some x-axis ticks were left out for visibility). 82

2.4 Results for samples made up of strains contained in the reference database across tools. **(A)** F1-scores, JSDs, and WSD for the different tools for the samples made up of strains contained in the reference database. The WSD for KrakenUniq in *M. tuberculosis* was comparatively high. For visibility of the results the y-axis has been truncated for this figure, such that outliers are not shown for KrakenUniq. Similarly, the samples of *A. baumannii* for which StrainScan failed to run were assigned a WSD of 1.0 and are hence also not shown. Shaded areas of lineplots indicate 95% confidence interval. For the boxplots, line indicates median, edges of box the 25% and 75% quartile, and whiskers extend to include the datapoint furthest from the median that is within 1.5 times the interquartile range. Dots are outliers. Assemblies on x-axis indicates assembly level analysis (see Outcome Measures section). **(B)** Expected WSDs for *A. baumannii* samples from 10000 random choices of paired strain populations (see Outcome Measures section) compared to the observed WSDs across samples composed of strains contained in the reference databases for the four different tools. Line indicates median, edges of box the 25% and 75% quartile, and whiskers extend to include the datapoint furthest from the median that is within 1.5 times the interquartile range. Dots are outliers. 92

2.5 Results for samples made up of strains not contained in the reference database across tools. **(A)** F1-scores, JSDs, and WSD for the different tools for the samples made up of strains not contained in the reference database. For the boxplots, line indicates median, edges of box the 25% and 75% quartile, and whiskers extend to include the datapoint furthest from the median that is within 1.5 times the interquartile range. Dots are outliers. WSD for *A. baumannii* does not show WSDs for StrainScan for samples which failed to run. Assemblies on x-axis indicates assembly level analysis (see Outcome Measure section). The WSD for KrakenUniq in *M. tuberculosis* and *E. coli* was comparatively high. For visibility of the results the y-axis has been truncated for this figure. **(B)** The difference between the minimum possible distance any estimated strain has to the true strained contained in a sample and the minimum possible distance achievable given all strains present in the reference database (see text for details). Y-axes are truncated for *A. baumannii* and *E. coli* for visibility. Note the difference in scale for the different panels for higher visibility. 93

2.6 Number of strains estimated for StrainScan and StrainGE **(A)** across species for the samples made up of strains contained in the reference database (left columns) and the samples made up of strains not contained in the reference database (right column; size of circles represent fraction of samples). **(B)** Number of strains estimated for KrakenUniq and StrainEst cross species for samples with strains contained in the reference database (size of circles represent fraction of samples). 96

2.7 Figures show relationships between F1 scores, cluster sizes and Mash distance thresholds used for complete linkage hierarchical clustering across species for StrainScan and StrainGE. The first figure shows the F1 score depending on the Mash distance threshold used for clustering. Each Mash distance threshold results in a different number of clusters, which is shown in the second figure. The third figure then shows the F1 scores in relation to the number of clusters that were inferred. Note here that the same clustering threshold in two different species can lead to different numbers of clusters. Hence, the difference in the number of clusters results are plotted for in the bottom panel. 98

2.8 Results for samples made up of strains contained in the reference database combined across all species for StrainScan and StrainGE, stratified by number of strains in the samples as well as total coverage. Here, all samples irrespective of the species making up the sample are analyzed. Similarly, samples analyzed include both samples with strains originating in the same and in different clusters at the 100 cluster level. The coverage shown is the total coverage of the sample, with both samples analyzed that have equal relative abundance for all strains as well as the samples that have one dominant strain at 0.8 relative abundance with the remaining strains having equal shares of the remaining 0.2 relative abundance. The high performance for the 1-fold coverage samples can be explained by the fact that only single strain samples were simulated at 1-fold coverage, whereas higher coverage samples were simulated for samples containing 1, 2, 3, and 4 strains. 99

3.1 Histogram of number of strains genes were found in. 109

3.2 Illustration of how genes not shared between strains can skew the observed nucleotide proportions. The last six loci (here assumed to represent an accessory gene) are only present in strain B (actual nucleotides only illustrative). This leads to all of these loci being observed at a proportion of 1.0, even though strain B is only present at a relative abundance of 0.3. . . . 111

3.3 Performance metrics of all model variants. Each model variants' performance is averaged across samples with 1, 2, and 3 strains and displayed by a dot. Boxplots show model performances grouped by their model type (strain-mixture, nucleotide-mixture, cluster-allele, and strain-allele model; cluster-allele model not shown for WSD as no assembly level inference is provided). Lines show median and whiskers show 1.5 times the interquartile range. 126

3.4 Figure presents results on the 25 test datasets for the strain mixture model. **(A)** Average performance metrics of strain-mixture models with different parameter settings on the test dataset. $\lambda = \lambda_p$ and $c = \gamma_{min}$. **(B)** Expected WSD from random populations with 1 to 5 strains each and estimated WSD obtained from the best performing strain mixture model ($\lambda_p = 10, \gamma_{min} = 1$). **(C)** ANI between estimated and true strains across all test samples. Turquoise squares show all-to-all pairwise whole genome and SCCG ANI for all reference strains. Blue triangles indicate ANIs between each estimated strain and the true strain with the highest SCCG ANI to that estimated strain. Red dashed line indicates $x = y$. Histograms show the marginal distributions of the whole genome and SCCG ANIs of the estimated strains. 127

4.1 Number of assemblies/strains genes occur in for each species. Histogram use 100 bins each. Hence, the first and last bar do not necessarily represent the number of genes present in only one or all strains. A clear distinction between the distributions can be observed, with *M. tuberculosis* and *L. monocytogenes* showing a much higher number of genes shared across many or all strains and only few singletons compared to *A. baumannii* and *E. coli*. Total gene numbers were similar across all species besides *M. tuberculosis* for which fewer genes were identified (*A. baumannii*: 856; *E. coli*: 1056; *M. tuberculosis*: 571; *L. monocytogenes*: 933). . 136

4.2 Average Mash distances of strains from all other strains in the species for the four analyzed species. 138

4.3 Performance of the different tested models across different taxonomic resolutions (10, 100, and 500 cluster level as well as for the identification of individual assemblies) across all parameter settings. Assembly label on x-axis indicates the analysis where recovery of unique assembly genomes is considered. Each line shows the mean performance averaged across all parameter settings for the corresponding model and samples. Shaded area shows the 95% confidence interval across parameter settings (i.e. each datapoint corresponds to the mean performance of the model with one set of parameters across all 20 samples). 148

4.4 Performance characteristics of accessory-proportion models with $\lambda_s = 0.01$ and $\lambda_s = 0.1$. Shown are F1-scores **(A-D)**, and JSDs **(E-H)** at the different levels of clustering (10 clusters, 100 clusters, 500 clusters, and assemblies, i.e. no clustering). WSD at the assembly level is shown in **(J)**. For the boxplots, line indicates median, edges of box the 25% and 75% quartile, and whiskers extend to include the datapoint furthest from the median that is within 1.5 times the interquartile range. Dots are outliers. **(I)** shows the difference between the estimated number of strains versus the true number of strains in the samples. Size of diamonds indicates the count of how often the difference is observed. 149

4.5 Mean number of SCCGs for different number of randomly sampled assemblies. Each number of assemblies was sampled 100 times. 150

4.6 **(A)** F1-score, precision, and recall for inferred gene content across samples. Recall is almost perfect, resulting in low visibility of boxplot (line at 1). **(B)** Histogram of per nucleotide coverage for true positive and false positive gene calls. **(C)** Number of true and false positive genes inferred by number of strains in the sample. **(D)** F1-score by number of strains. 151

4.7 **(A)** Histogram for true and false positive gene calls by gene variability, defined as the mean of all-to-all pairwise hamming distances between strains. **(B)** Histogram of differences in gene variability from randomly permuted genes. Line indicates the observed mean difference between variability in true and false positive gene calls. 152

4.8 Processing times for different number of core genes in the reference database on a log base 2 scale. For the boxplots, line indicates median, edges of box the 25% and 75% quartile, and whiskers extend to include the datapoint furthest from the median that is within 1.5 times the interquartile range. Dots are outliers. The blue line indicates one hour and the red line ten hours. 153

4.9 F1-scores, JSDs, and WSD for the APM, StrainScan, and StrainGE for the samples made up of strains contained in the reference database. Shaded areas of lineplots indicate 95% confidence interval. For the boxplots, line indicates median, edges of box the 25% and 75% quartile, and whiskers extend to include the datapoint furthest from the median that is within 1.5 times the interquartile range. Dots are outliers. WSD for *A. baumannii* does not show WSDs for StrainScan for samples which failed to run. Assemblies label on x-axis indicates assembly level analysis. 155

4.10 Number of inferred strains versus true strains for StrainScan, StrainGE, and the APM on **(A-D)** the samples made up of strains contained in the reference database and **(E-H)** the samples made up of strains not contained in the reference database. 156

4.11 **(A)** F1-scores, JSDs, and WSD for the APM, StrainScan, and StrainGE for the samples made up of strains not contained in the reference database. Shaded areas of lineplots indicate 95% confidence interval. For the boxplots, line indicates median, edges of box the 25% and 75% quartile, and whiskers extend to include the datapoint furthest from the median that is within 1.5 times the interquartile range. Dots are outliers. WSD for *A. baumannii* is truncated on the y-axis to increase visibility. Assemblies on x-axis indicates assembly level analysis. **(B)** F1-scores for accessory gene assignment in samples made up of strains not contained in the reference database across species. 157

4.12 True and false positive gene calls as well as precision of gene calling depending on *e*-value used for BLAST search in Prokka for the isolate organism in the samples for *A. baumannii*, *E. coli*, and *L. monocytogenes* in NIST mock metagenomic data. Note the different in scales between the figures to allow for better visibility. 161

5.1 **(A-G)** Absolute error of estimated prevalence from true prevalence for each true prevalence for group size 9 array testing and individual testing with a total of 360 tests at varying sensitivities and specificities. Line indicates median, edges of box the 25% and 75% quartile, and whiskers extend to include the datapoint furthest from the median that is within 1.5 times the interquartile range. Dots are outliers. **(H)** Mean absolute error between true and estimated prevalence for array testing and individual testing aggregated across different sensitivities and specificities. Line shows the mean and shaded area the 95% confidence interval. 172

5.2 **(A-G)** Absolute error of estimated prevalence for group testing with wrong sensitivity and specificity estimate, as well as group testing and individual testing with correct sensitivity and specificity estimate. For the wrong estimates, estimates were $S_e = 0.95, S_p = 0.95$ for the true values $S_e = 0.99, S_p = 0.99$ and $S_e = 0.99, S_p = 0.99$ for $S_e = 0.95, S_p = 0.95$. Line indicates median, edges of box the 25% and 75% quartile, and whiskers extend to include the datapoint furthest from the median that is within 1.5 times the interquartile range. Dots are outliers. **(H)** Percent bias of group testing estimators with correct and incorrect sensitivity and specificity estimates for different underlying prevalence rates. Note that the maximum negative bias is 100% due to the lower limit of a prevalence of 0. Line shows the mean and shaded area the 95% confidence interval. 173

5.3 Group testing performance across different prevalence rates with different number of group testing rounds. Each group testing round was one array testing design with 18 tests and 81 samples. **(A)** Variance of the posterior distribution. **(B)** Absolute error from the true prevalence. In both plots line indicates median, edges of box the 25% and 75% quartile, and whiskers extend to include the data point furthest from the median that is within 1.5 times the interquartile range. Dots are outliers. 174

5.4 Comparison of prevalence estimation across different array testing designs of size 16, 64, and 256 with 320 tests in total. N indicates total number of samples tested. and T the number of tests. Also shown is $N = 81, T = 360, S_e = 0.95, S_p = 0.99$ for comparison of previous results. **(A)** Estimated versus true prevalence. **(B)** Absolute error between the estimated and true prevalence. In both plot line indicates median, edges of box the 25% and 75% quartile, and whiskers extend to include the datapoint furthest from the median that is within 1.5 times the interquartile range. Dots are outliers. 175

6.1 Illustration of a (49,7,4)-multipool. Each cell is one individual sample. Cells colored in one color indicate membership in the same group. Top row displays groupings along rows, with all samples in one row grouped into one group test (together, the row groupings correspond to the $g \times N$ submatrix of M for $j = 1$).Second row displays group test against the first diagonal and the third row group tests along the second diagonal. Note the action of the cyclic shift, which corresponds to the group test membership of the samples being shifted right, wrapping around to the left side of the array. The fourth row indicates grouping of samples into group tests along columns. Taking only row and column tests corresponds to array testing. Samples are grouped into one row, one column, and one of the diagonal tests each, illustrated in the example of overlaid groups. Here, each cell colored in the same color indicates the same test with multiple colors indicating membership in two or more of the illustrated tests. The other groups are created by shifting the shown groups horizontally (or vertically for the orange group), and cells wrapping around go to the first (left) column when they would pass beyond the last (right) column as illustrated in the first four rows of the figure. 182

6.2 **(A)** Illustration of a decoding iteration. In each iteration i a status vector ω_i indicating positivity of the individual samples is randomly changed in one position from positive (1) to negative (0), or vice versa. The resulting status vector ω_{i+1} is used as basis for a *maximum a posteriori* optimization of the individual Ct values (here, the parameters of are denoted in the general form Θ). Based on these individual Ct values, expected group test results are predicted and we obtain a mean squared error between these expected test results and the observed ones. Based on this mean squared errors' difference to the mean squared error in the previous iteration as well as a temperature that declines with every iteration we determine the probability of accepting the proposed status vector ω_{i+1} or rejecting it and continuing with the current status vector ω_i. The declining temperature at every iteration effectively causes the function of the acceptance probability to become steeper, such that we accept almost all proposals in the beginning, but are more strict in the end, accepting only improvements in the mean squared error (negative differences, for which the function is always equal to 1). The status vector after 500 iterations (with 10 iterations at each temperature) is then used to infer the positive samples. **(B)** Change in acceptance probabilities for different temperatures with temperature decreasing from left to right plots. Figure and caption taken from Unruh et al. [1]. 189

6.3 Illustration of the approach of setting each negatively inferred item to a minimum Ct value to check whether a low Ct value may be obscured by higher other Ct values. First, the simulated annealing process infers the positives from the potential positives that are in positive row and column tests (samples A, B, C, and D). In this example, sample A is inferred with an individual Ct value of 21 and sample D is inferred with an individual Ct value of 24 (marked in red). We also obtain predicted test outcomes $\hat{Y}_1$. In the next stage, the samples not inferred as positive (samples B and C) are individually set to a Ct value of 30. For example, in the left middle panel, sample B is set to a Ct value of 30. We again infer the predicted test outcomes $\hat{Y}_2$, now assuming sample B is positive with an individual Ct value of 30. Here, we observe that the new predicted test outcomes differ more than ϵ from the initially inferred ones (i.e. $|\hat{Y}_2 - \hat{Y}_1| > \epsilon$). Hence, sample B could not have been hidden by samples A and B and is inferred as negative. We again do the same for sample C, displayed in the right panel. Note that here sample B is again assumed to be negative. We again estimate the new test outcomes $\hat{Y}_3$ and observe that $|\hat{Y}_3 - \hat{Y}_1| \leq \epsilon$. Hence, we infer that if sample C had a low individual Ct value it could be hidden by the higher Ct values of A and B and we infer it as positive. Altogether, we then arrive at the final inference, where samples A, C, and D are inferred as positive, but sample B is inferred as negative. 191

6.4 Performance characteristics of COMP and gtMAP for in silico simulations for a 3-disjunct group testing design ($n = 81$) with different numbers of positive samples. PPV is the positive predictive value ($\frac{\text{true positives}}{\text{true positives+false positives}}$) and NPV the negative predictive value ($\frac{\text{true negatives}}{\text{true negatives+false negatives}}$). Accuracy is defined as $\frac{\text{true positives+true negatives}}{\text{positives+negatives}}$, i.e. the total number of positive identifications out of all samples. 194

6.5 **(A)** Sensitivity analysis showing number of false positives and false negatives per 100 positive and negative samples, respectively, for COMP and gtMAP depending on the number of positive samples in a group testing round. **(B)** ROC curves for gtMAP decoding using different decision thresholds (increasing from left to right) and different number of positives. Black '+' indicate a decision threshold of 0.5. 195

A1 Count of assemblies of *E. coli* assigned to each cluster at the three clustering levels (assemblies clustered into 10, 100, and 500 clusters, labeled as 10 Cluster Level, 100 Cluster Level, and 500 Cluster Level, respectively). Cluster numbers are arbitrary and numbering increases left to right on the x-axis (some x-axis ticks were left out for visibility). 246

A2 Count of assemblies of *M. tuberculosis* assigned to each cluster at the three clustering levels (assemblies clustered into 10, 100, and 500 clusters, labeled as 10 Cluster Level, 100 Cluster Level, and 500 Cluster Level, respectively). Cluster numbers are arbitrary and numbering increases left to right on the x-axis (some x-axis ticks were left out for visibility). 247

A3 Count of assemblies of *L. monocytogenes* assigned to each cluster at the three clustering levels (assemblies clustered into 10, 100, and 500 clusters, labeled as 10 Cluster Level, 100 Cluster Level, and 500 Cluster Level, respectively). Cluster numbers are arbitrary and numbering increases left to right on the x-axis (some x-axis ticks were left out for visibility). 248

A4 Assembly statistics for *A. baumannii*. Genome Length is the length of the genomes in nucleotides. N count is the number of ambiguous bases in the assemblies. Number of Gaps is the number of gaps (-) in the assemblies. N50 is the shortest contig that needs to be included to cover over 50% of the genome length and N50 count the number of contigs that need to be included to reach the N50. Number of Contigs is the number of contigs the assemblies consist of. 249

A5 Assembly statistics for *E. coli*. Genome Length is the length of the genomes in nucleotides. N count is the number of ambiguous bases in the assemblies. Number of Gaps is the number of gaps (-) in the assemblies. N50 is the shortest contig that needs to be included to cover over 50% of the genome length and N50 count the number of contigs that need to be included to reach the N50. Number of Contigs is the number of contigs the assemblies consist of. 250

A6 Assembly statistics for *M. tuberculosis*. Genome Length is the length of the genomes in nucleotides. N count is the number of ambiguous bases in the assemblies. Number of Gaps is the number of gaps (-) in the assemblies. N50 is the shortest contig that needs to be included to cover over 50% of the genome length and N50 count the number of contigs that need to be included to reach the N50. Number of Contigs is the number of contigs the assemblies consist of. 251

A7 Assembly statistics for *L. monocytogenes*. Genome Length is the length of the genomes in nucleotides. N count is the number of ambiguous bases in the assemblies. Number of Gaps is the number of gaps (-) in the assemblies. N50 is the shortest contig that needs to be included to cover over 50% of the genome length and N50 count the number of contigs that need to be included to reach the N50. Number of Contigs is the number of contigs the assemblies consist of. 252

A8 Inferred number of strains by KrakenUniq for samples made up of strains contained in the reference database. Size of circle represents fraction of samples. 253

A9 Inferred number of strains by StrainEst for samples made up of strains contained in the reference database. Size of circle represents fraction of samples. 254

A10 Results for samples made up of strains of *A. baumannii* contained in the reference database combined across all species for StrainScan, StrainGE, KrakenUniq, and StrainEst, stratified by number of strains in the samples. Samples analyzed include both samples with strains originating in the same and in different clusters at the 100 cluster level as well as samples with different coverages. Both samples that have equal relative abundance for all strains as well as the samples that have one dominant strain at 0.8 relative abundance with the remaining strains having equal shares of the remaining 0.2 relative abundance were analyzed in combination. Bar height shows mean performance and error bars the 95% confidence interval. 255

A11 Results for samples made up of strains of *E. coli* contained in the reference database combined across all species for StrainScan, StrainGE, KrakenUniq, and StrainEst, stratified by number of strains in the samples. Samples analyzed include both samples with strains originating in the same and in different clusters at the 100 cluster level as well as samples with different coverages. Both samples that have equal relative abundance for all strains as well as the samples that have one dominant strain at 0.8 relative abundance with the remaining strains having equal shares of the remaining 0.2 relative abundance were analyzed in combination. Bar height shows mean performance and error bars the 95% confidence interval. 256

A12 Results for samples made up of strains of *M. tuberculosis* contained in the reference database combined across all species for StrainScan, StrainGE, KrakenUniq, and StrainEst, stratified by number of strains in the samples. Samples analyzed include both samples with strains originating in the same and in different clusters at the 100 cluster level as well as samples with different coverages. Both samples that have equal relative abundance for all strains as well as the samples that have one dominant strain at 0.8 relative abundance with the remaining strains having equal shares of the remaining 0.2 relative abundance were analyzed in combination. Bar height shows mean performance and error bars the 95% confidence interval. 257

A13 Results for samples made up of strains of *L. monocytogenes* contained in the reference database combined across all species for StrainScan, StrainGE, KrakenUniq, and StrainEst, stratified by number of strains in the samples. Samples analyzed include both samples with strains originating in the same and in different clusters at the 100 cluster level as well as samples with different coverages. Both samples that have equal relative abundance for all strains as well as the samples that have one dominant strain at 0.8 relative abundance with the remaining strains having equal shares of the remaining 0.2 relative abundance were analyzed in combination. Bar height shows mean performance and error bars the 95% confidence interval. 258

A14 Results for samples made up of strains of *A. baumannii* contained in the reference database combined across all species for StrainScan, StrainGE, KrakenUniq, and StrainEst, stratified by the total coverage of the samples. Samples analyzed include both samples with strains originating in the same and in different clusters at the 100 cluster level as well as samples with different numbers of constituent strains. Both samples that have equal relative abundance for all strains as well as the samples that have one dominant strain at 0.8 relative abundance with the remaining strains having equal shares of the remaining 0.2 relative abundance were analyzed in combination. Bar height shows mean performance and error bars the 95% confidence interval. 259

A15 Results for samples made up of strains of *E. coli* contained in the reference database combined across all species for StrainScan, StrainGE, KrakenUniq, and StrainEst, stratified by the total coverage of the samples. Samples analyzed include both samples with strains originating in the same and in different clusters at the 100 cluster level as well as samples with different numbers of constituent strains. Both samples that have equal relative abundance for all strains as well as the samples that have one dominant strain at 0.8 relative abundance with the remaining strains having equal shares of the remaining 0.2 relative abundance were analyzed in combination. Bar height shows mean performance and error bars the 95% confidence interval. 260

A16 Results for samples made up of strains of *M. tuberculosis* contained in the reference database combined across all species for StrainScan, StrainGE, KrakenUniq, and StrainEst, stratified by the total coverage of the samples. Samples analyzed include both samples with strains originating in the same and in different clusters at the 100 cluster level as well as samples with different numbers of constituent strains. Both samples that have equal relative abundance for all strains as well as the samples that have one dominant strain at 0.8 relative abundance with the remaining strains having equal shares of the remaining 0.2 relative abundance were analyzed in combination. Bar height shows mean performance and error bars the 95% confidence interval. 261

A17 Results for samples made up of strains of *L. monocytogenes* contained in the reference database combined across all species for StrainScan, StrainGE, KrakenUniq, and StrainEst, stratified by the total coverage of the samples. Samples analyzed include both samples with strains originating in the same and in different clusters at the 100 cluster level as well as samples with different numbers of constituent strains. Both samples that have equal relative abundance for all strains as well as the samples that have one dominant strain at 0.8 relative abundance with the remaining strains having equal shares of the remaining 0.2 relative abundance were analyzed in combination. Bar height shows mean performance and error bars the 95% confidence interval. 262

A18 Results for samples made up of strains of *A. baumannii* contained in the reference database combined across all species for StrainScan, StrainGE, KrakenUniq, and StrainEst, stratified by the relative abundance profile. In the equal profile all strains have equal relative abundance. In the dominant strain samples one strain's relative abundance is 80%, with all other strains being equally distributed across the remaining 20%. Samples analyzed together include both samples with strains originating in the same and in different clusters at the 100 cluster level as well as samples with different numbers of constituent strains and coverages. Bar height shows mean performance and error bars the 95% confidence interval. 263

A19 Results for samples made up of strains of *E. coli* contained in the reference database combined across all species for StrainScan, StrainGE, KrakenUniq, and StrainEst, stratified by the relative abundance profile. In the equal profile all strains have equal relative abundance. In the dominant strain samples one strain's relative abundance is 80%, with all other strains being equally distributed across the remaining 20%. Samples analyzed together include both samples with strains originating in the same and in different clusters at the 100 cluster level as well as samples with different numbers of constituent strains and coverages. Bar height shows mean performance and error bars the 95% confidence interval. 264

A20 Results for samples made up of strains of *M. tuberculosis* contained in the reference database combined across all species for StrainScan, StrainGE, KrakenUniq, and StrainEst, stratified by the relative abundance profile. In the equal profile all strains have equal relative abundance. In the dominant strain samples one strain's relative abundance is 80%, with all other strains being equally distributed across the remaining 20%. Samples analyzed together include both samples with strains originating in the same and in different clusters at the 100 cluster level as well as samples with different numbers of constituent strains and coverages. Bar height shows mean performance and error bars the 95% confidence interval. 265

A21 Results for samples made up of strains of *L. monocytogenes* contained in the reference database combined across all species for StrainScan, StrainGE, KrakenUniq, and StrainEst, stratified by the relative abundance profile. In the equal profile all strains have equal relative abundance. In the dominant strain samples one strain's relative abundance is 80%, with all other strains being equally distributed across the remaining 20%. Samples analyzed together include both samples with strains originating in the same and in different clusters at the 100 cluster level as well as samples with different numbers of constituent strains and coverages. Bar height shows mean performance and error bars the 95% confidence interval. 266

A22 Results for samples made up of strains of *A. baumannii* contained in the reference database combined across all species for StrainScan, StrainGE, KrakenUniq, and StrainEst, stratified by the cluster of origin. Same cluster indicates that the strains in the sample are in the same cluster at the 100 cluster level, while different indicate they originate from different clusters. Samples analyzed together include both samples with different numbers of constituent strains and coverages. Both samples that have equal relative abundance for all strains as well as the samples that have one dominant strain at 0.8 relative abundance with the remaining strains having equal shares of the remaining 0.2 relative abundance were analyzed in combination. Bar height shows mean performance and error bars the 95% confidence interval. 267

A23 Results for samples made up of strains of *E. coli* contained in the reference database combined across all species for StrainScan, StrainGE, KrakenUniq, and StrainEst, stratified by the cluster of origin. Same cluster indicates that the strains in the sample are in the same cluster at the 100 cluster level, while different indicate they originate from different clusters. Samples analyzed together include both samples with different numbers of constituent strains and coverages. Both samples that have equal relative abundance for all strains as well as the samples that have one dominant strain at 0.8 relative abundance with the remaining strains having equal shares of the remaining 0.2 relative abundance were analyzed in combination. Bar height shows mean performance and error bars the 95% confidence interval. 268

A24 Results for samples made up of strains of *M. tuberculosis* contained in the reference database combined across all species for StrainScan, StrainGE, KrakenUniq, and StrainEst, stratified by the cluster of origin. Same cluster indicates that the strains in the sample are in the same cluster at the 100 cluster level, while different indicate they originate from different clusters. Samples analyzed together include both samples with different numbers of constituent strains and coverages. Both samples that have equal relative abundance for all strains as well as the samples that have one dominant strain at 0.8 relative abundance with the remaining strains having equal shares of the remaining 0.2 relative abundance were analyzed in combination. Bar height shows mean performance and error bars the 95% confidence interval. 269

A25 Results for samples made up of strains of *L. monocytogenes* contained in the reference database combined across all species for StrainScan, StrainGE, KrakenUniq, and StrainEst, stratified by the cluster of origin. Same cluster indicates that the strains in the sample are in the same cluster at the 100 cluster level, while different indicate they originate from different clusters. Samples analyzed together include both samples with different numbers of constituent strains and coverages. Both samples that have equal relative abundance for all strains as well as the samples that have one dominant strain at 0.8 relative abundance with the remaining strains having equal shares of the remaining 0.2 relative abundance were analyzed in combination. Bar height shows mean performance and error bars the 95% confidence interval. 270

B1 Performance of the APM for samples made up of strains contained in the reference database for all species (each column of figures corresponding to one species), stratified by the number of strains in the sample. Samples analyzed together include both samples with strains originating in the same and in different clusters at the 100 cluster level as well as samples with different coverages. Both samples that have equal relative abundance for all strains as well as the samples that have one dominant strain at 0.8 relative abundance with the remaining strains having equal shares of the remaining 0.2 relative abundance were analyzed in combination. Bar height shows mean performance and error bars the 95% confidence interval. 272

B2 Performance of the APM for samples made up of strains contained in the reference database for all species (each column of figures corresponding to one species), stratified by the total coverage the sample. Samples analyzed together include both samples with strains originating in the same and in different clusters at the 100 cluster level as well as samples containing different number of strains. Both samples that have equal relative abundance for all strains as well as the samples that have one dominant strain at 0.8 relative abundance with the remaining strains having equal shares of the remaining 0.2 relative abundance were analyzed in combination. Bar height shows mean performance and error bars the 95% confidence interval. Note the high performance for 1-fold coverage. This is likely due to the fact that only samples with one constituent strain were created at this coverage. 273

B3 Results of the APM for samples made up of strains contained in the reference database for all species (each column of figures corresponding to one species), stratified by by the relative abundance profile. In the equal profile all strains have equal relative abundance. In the dominant strain samples one strain's relative abundance is 80%, with all other strains being equally distributed across the remaining 20%. Samples analyzed together include both samples with strains originating in the same and in different clusters at the 100 cluster level as well as samples with different numbers of constituent strains and coverages. Bar height shows mean performance and error bars the 95% confidence interval. 274

B4 Results of the APM for samples made up of strains contained in the reference database for all species (each column of figures corresponding to one species), stratified by the cluster of origin. Same cluster indicates that the strains in the sample are in the same cluster at the 100 cluster level, while different indicate they originate from different clusters. Samples analyzed together include both samples with different numbers of constituent strains and coverages. Both samples that have equal relative abundance for all strains as well as the samples that have one dominant strain at 0.8 relative abundance with the remaining strains having equal shares of the remaining 0.2 relative abundance were analyzed in combination. Bar height shows mean performance and error bars the 95% confidence interval. 275

C1 The probability of a sample being inferred as false positive at different prevalence rates and different sensitivity and specificity values for NCOMP (lines shaded red) and COMP (lines shaded blue). 281

List of Publications

[1] Zabeti, H., Dexter, N., Lau, I., Unruh, L. A. H. , Adcock, B., Chindelevitch, L. (2021). Group testing large populations for SARS-CoV-2. medRxiv, 2021-06.

[2] Chindelevitch, L., Jauneikaite, E., Wheeler, N. E., Allel, K., Ansiri-Asafoakaa, B. Y., Awuah, W. A., et al. (2022). Applying data technologies to combat AMR: current status, challenges, and opportunities on the way forward. arXiv preprint arXiv:2208.04683.

[3] Unruh, L. A. H., Crone, M. A., Freemont, P. S., Chindelevitch, L. (2024). A standardised, high-throughput approach to diagnostic group testing method validation. medRxiv, 2024-04.

Chapter I

Introduction

'Nobody ever figures out what life is all about, and it doesn't matter. Explore the world. Nearly everything is really interesting if you go into it deeply enough.'
– Richard Feynman

Bacterial and viral pathogens present a major threat to public health and place a major burden on the healthcare system. In 2019, 13.7 million people were estimated to die due to bacterial infection [2] and antibiotic resistance is a growing threat, with 1.27 million people already estimated to die due to it each year [3]. Moreover, antibiotic resistance has a major economic impact [4]. As has been made abundantly clear by the Covid-19 pandemic, viruses present another central threat to human health. One major aim of public health has to be to curb the spread of infectious agents, especially novel, potentially more virulent, transmissible, or resistant variants. Underlying our ability to control infectious diseases is an understanding of the identity, spread, and characteristics of pathogens of interest. This understanding can only be gained by detailed surveillance. Surveillance capacity, methods, and strategies have improved substantially over time. Yet, challenges remain. Two central issues in employing effective surveillance are the implementation of surveillance in low resource settings and computational methods for the analysis of biological data to extract relevant information. During the Covid-19 pandemic, the WHO has repeatedly published guidelines, with widespread surveillance and testing being central pillars of their strategy [5]. Especially in low- and middle-income countries, these strategies cannot always be implemented to the extend that is required due to resource limitations, with one central limitation being the scarce availability of testing resources [6,7]. On the computational side, developments are regularly made and new tools published. However, it is often not clear when to use which of the potential multitude of applicable methods and which results we can trust. New methods also have a central place, either solving previously unsolved problems, improving on previous methods, or approaching issues from a

new angle. In this thesis, I address the above issues in two different areas relevant to public health. Firstly, I focus on bacterial pathogens, metagenomics, and the benchmarking and development of computational tools that enable bacterial identification in metagenomic data. Metagenomic surveillance of wastewater in particular has been frequently discussed in the literature[8,9,10,11], and I believe that improved identification of bacterial strains can provide novel insights in this area. Secondly, I present an investigation into group testing, the practice of pooling multiple samples into one group that is then tested, for respiratory viruses from a computational and theoretical point of view and address the transition from theory to practice via in vitro validation. Group testing has the potential to greatly reduce the number of tests required to test a population for an infectious agent, especially valuable in surveillance testing. As such, it shows great promise for improving surveillance in low resource settings.

In this chapter, I will give an overview of the background underlying this thesis. I start by briefly introducing metagenomics, before investigating the bacterial species and strain concepts, their application in metagenomics, and the current approaches for identifying bacterial strains in metagenomic datasets. After this excursion into the bacterial and metagenomic realm, I will address testing for respiratory viruses. I explain the concept of group testing for infectious diseases and polymerase chain reactions, the gold standard for determining the presence of a pathogen of interest in a sample. Lastly, the adjacent topic of prevalence estimation of infectious diseases via group testing is discussed briefly. I finally present the overall research aims of this thesis.

Overview of Metagenomics

Before diving into the details of evaluating published tools and developing new methods in metagenomics, it is first necessary to clearly state the nature of metagenomic data. In particular, I will focus on short-read shotgun metagenomics as opposed to other types, such as long-read metagenomics or 16s-RNA sequencing. In line with the focus on short-read shotgun metagenomics, whenever metagenomics is mentioned in the text, short-read shotgun metagenomics is meant.

The goal of metagenomics is to provide insights into the genetic structure of a community of organisms or cells, for example a microbiome consisting of bacteria, viruses, human cells, and others. Following sample collection, the first step in the generation of a metagenomic dataset is DNA extraction and library creation[12]. Note that prior to DNA extraction the samples may be filtered, for example by organism size, to enable enhanced sequencing of the organisms of interest[13]. Centrifugation is another method to further separate different organisms, often employed in viral metagenomics[14] or for the enrichment of mitochondrial DNA[15]. DNA probes can also be employed. These probes capture DNA segments of interest as well as related sequences through hybridization[16]. However, this technique requires a priori decisions about the sequences of interest, limiting its use in exploratory analyses of the broader diversity of DNA in the microbiome. Following DNA extraction and library creation, shotgun sequencing is performed. Shotgun sequencing refers to the splitting of the DNA in the sample into short, random fragments. Following this, the resulting fragments are sequenced, resulting in reads typically 100 to 200 basepairs (bp) in length. Various platforms are available, with one of the most widely used being the Illumina platform, which shows low error rates (0.1% to 1%) and returns hundreds of millions or even billions of reads[17].

Following sequencing, the raw reads need to be quality controlled and filtered. Filtering is important to remove contaminant sequences, such as human DNA[12]. Contaminants related to laboratory equipment/reagents can also present an issue[18] that should be accounted for where possible. Quality control is important to ensure that only high quality bases and reads are kept for further analysis. Sequencing platforms provide quality scores for each sampled base and usually a decline in quality is observed towards the end of reads[19]. Tools such as FastQC[20] provide further measures of sequence quality. The per-sequence GC content can be investigated, where peaks or multimodal distributions in sequencing of cultured organisms may hint to contaminants, though the latter is not necessarily true for metagenomics with its wide range of constituent organisms. Further, duplicated and over-represented sequences can point to over-sequencing or the presence of contaminants. FastQC can also point to the presence of adapter sequences that need to be removed, though the specific adapter sequences used in sequencing have to be kept in mind. Adapter sequences surround the

DNA insert and allow binding to the flow cell of the sequencing machine[21]. Failure to remove these sequences can greatly influence downstream processing[19]. Following quality control, trimming needs to be performed. Steps can be taken that remove beginnings and ends of reads with low quality, whole low quality reads, or that remove and trim reads in other ways. Trimming further includes the removal of adapter sequences. These steps can be performed with frequently used tools such as Trimmomatic[19] or BBDuk (part of BBMap; `https://sourceforge.net/projects/bbmap/`).

Many metagenomic pipelines include an assembly step[12,13]. In this step, reads are combined to create longer sequences (contigs). Longer contigs enable the more reliable calling of genes or other functional elements. Many tools for contig creation exist for metagenomic data, tackling the new challenges such as unknown diversity and abundance of organisms in different ways[22]. These tools, similar to single genome assembly, fall broadly into two categories, those using overlap-layout consensus methods and those utilizing de Bruijn graphs[22]. In the overlap-layout consensus approach, the overlap between all pairs of reads is determined. Based on these overlaps, a layout graph is constructed, with nodes representing reads and edges representing overlaps. The task of assembling a genome is then solved by finding the path through the graph with the highest overlap. A de Bruijn graph is constructed by taking all k-mers in a dataset (nucleotide sequences of length k) and connecting two kmers if the last $k-1$ nucleotides of one k-mer match the first $k-1$ nucleotides of another. Assembling a contig from this graph revolves around finding an optimal path through the graph, though repeats or sequencing errors make this process not straightforward[22]. Additionally, metagenomic data brings additional challenges not present in single genome assembly, such as unknown organism diversity and abundance[22]. Notably, contigs produced through Illumina sequencing show a high match to the true genetic sequences of the organisms of origin[23]. However, especially for short read sequencing, many reads will map to more than one genome, and hence, could be included in multiple contigs[23], creating chimeric contigs. This ambiguity in read assignment will be especially high with multiple organisms of high similarity[23]. The contigs resulting from assembly are regularly binned together to create metagenome assembled genomes[24] (MAGs). The binning of contigs likely also suffers from similar issues as contig assembly itself. Contigs from very similar organisms will be binned together given frequently used binning criteria such as those employed by MetaBAT[25]. For example, when considering one possible binning criterion, GC content (the proportion Guanine (G) and Cytosine (C) make up of the genome), different strains of the same species may be assigned to the same bin, as GC content is highly similar between organisms of a single species[26,27]. The hurdles in contig assembly will be reduced for higher coverages, as more reads will overlap. However, high enough coverage is rarely achievable for many genomes[13]. Due to these considerations, the presentation of MAGs as complete genetic sequences is prone to be erroneous in many cases and great care should be taken. Regarding

chimeric contigs derived from shotgun metagenomic data, tools such as DESMAN[28] have been developed to deconvolute MAGs into their constituent strains. However, DESMAN itself shows only high accuracy when considering many samples at the same time. Further, the method is based on high quality assembly and binning, which is often not a given. While reconstructing strains from contigs may be difficult, it may be worthwhile to use knowledge on the strain composition of a sample in the quality assessment or during the creation of MAGs. Altogether, quality assessment of MAGs is essential (and has been reviewed in detail in Yang et al.[29]). In summary, the limitations of MAGs will frequently necessitate alternative approaches to assess taxonomic content and functional assignment to taxonomies within the metagenome, which is the underlying goal of a part of the presented work.

Importantly, advances in sequencing approaches can alleviate some of the challenges present in short read metagenomics. Firstly, 3-Hi-C can assist in improving binning. 3-Hi-C is a technique that can provide information on the co-localization of DNA fragments in the same cell[30]. This additional information can be used to improve binning[30]. However, for many metagenomics datasets 3-Hi-C data is not available and the method requires additional preparation and sequencing steps[31], increasing the cost of application. Additionally, the presence of related organisms, such as closely related species or multiple strains, still presents a challenge for assembly and binning even in the 3-Hi-C context[32,33]. Nonetheless, future developments in this area may provide us with promising technologies and approaches going forward. Secondly, long read sequencing, enabled by platforms such as Oxford Nanopore or PacBio, is consistently improving. The longer sequences obtained, on the order of kilo-basepairs[34], can make assembly easier, though chimeric contigs still present a challenge[35]. Importantly, while the increased number of sequencing errors has been an issue for long-read sequencing in the past[34], the error rates are quickly approaching those of short-read sequencing[35]. Despite these exciting technological developments in sequencing, short read metagenomics stands out due to the wide availability of data and the higher sequencing depth compared to long-read technologies[36]. Similarly, already collected short read metagenomic data can be re-analyzed with novel approaches going forward, giving us the opportunity to gain new insights from old data. For these reasons, I focus on short read metagenomic data in this thesis.

Finally, metagenomic analysis often relies on the use of published analysis tools and packages. These need to be validated on data. However, for metagenomics we often run into the problem that the ground truth is not known. This can be tackled in three main ways. Firstly, there is the option of pure simulation from known sequences of various organisms through tools such as ART[37]. This to an extend can also simulate the presence of sequences not seen by a priori exclusion of these sequences from any database used. Notably, this might be different from the presence of sequences from unculturable organisms in real metagenomic

data, as these may display unknown features. Further, this approach might not always replicate all the specifics of real DNA extraction and sequencing completely. Secondly, metagenomic data can be created from sequencing experiments on mock communities, whose real composition is known as they are created in a laboratory. Multiple such mock communities can be found on public databases, with the some of the ones presented in Valencia et al.[38] used in this thesis. Again the presence of unculturable organisms cannot be replicated. Additionally, mock communities may not exhibit the same complexity or diversity as real metagenomic datasets. Finally, metagenomic samples can be spiked, meaning that sequences of known organisms are added to a real metagenomic sample via simulation[39]. This likely represents the closest match to real metagenomic data, but also comes with drawbacks, as for a large portion of the dataset the ground truth is again not known. As such the organisms or close relatives of theirs that the sample is spiked with may be present in the real metagenomic data, complicating the analysis of tool performance.

Bacterial Strains

Chapter II, Chapter III, and Chapter IV of this thesis are concerned with the identification of bacterial strains in short-read metagenomic data. Here, I will give some definitions and elaborate on the relevance of strain-level identification in short-read metagenomic sequencing, reviewing some potential applications. First of all, we need to define a bacterial species. The term species originates in the study of diploid, sexually reproducing organisms. Two organisms here are defined as the same species if they interbreed, though definitions can extend beyond this[40]. Bacteria do not engage in sexual reproduction. Nonetheless, classification of different organisms into distinct groups is useful when talking about them, and classification can be achieved in different ways. The advent of DNA sequencing has shifted the focus of considering what constitutes a bacterial species onto the genotypic stage. There are several ways to use genomics to classify bacteria into species and strains. Firstly, the bacterial ribosomal 16s rRNA gene is shared across all bacteria. Hence, it can be compared between all bacteria, and when the similarity in the investigated region of the gene is above a set threshold (e.g. 97% as determined by Stackebrandt and Goebel[41]) two organisms are considered to belong to the same species (or more conservatively to the same operational taxonomic unit or OTU). Notably, while such an approach can be a fairly reliable measure for species demarcation, the same can not be said for intraspecies typing[42]. An approach for identification of species from whole genome sequencing is the average nucleotide identity (ANI) of a genomic sequence to a reference genome. If this similarity is high enough (typically above 95%[43]) two species are considered the same. Both of the above approaches are applied frequently to delineate species in culture based approaches. The species concept is generally a useful

one. This is for example evidenced by the fact that certain species share important virulence factors, such as almost all members of *L. monocytogenes* carrying the conserved inlA-inlB locus and the Listeria Pathogenicity Island 1[44] or that plasmids preferably are shared among members of a species[45,46]. Turning our view to metagenomics and species classification I first want to briefly address 16s sequencing for metagenomics, which utilizes the 16s rRNA gene to provide a delineation of species in microbial communities. In this approach, part of the bacterial ribosomal 16s rRNA gene is amplified using polymerase chain reaction (PCR) and then sequenced. This approach has practical advantages such as quick turnaround time, and, similar to shotgun metagenomic sequencing, can provide a taxonomic overview of the diverse bacterial population in a sample. However, the degree to which taxonomic classification is achievable is not always clear and depends on which part of the 16s gene is targeted. Greay et al.[47] conducted a study where they attempted to identify bacterial pathogens in wastewater data. They showed that when using the V4 region only limited taxonomic resolution is achievable, making genera and species hard to identify. Greay et al.[47] showed that the individual regions of the 16s gene often cannot provide sufficient taxonomic resolution to identify pathogenic species in wastewater samples, highlighting the importance of this on the example of differentiating *E. coli* serotypes in wastewater. Notably, Ibal et al.[48] showed that higher resolution differentiation could be obtainable using the complete 16s gene. However, Greay et al.[47] correctly remark that shotgun metagenomic sequencing is better suited for providing the functional and taxonomic information required. Similarly, Jovel et al.[49] point out that 16s sequencing does not lend itself to in-depth functional analysis. However, species and sub-species level taxonomic classification, paired with functional analysis is an important endeavor and there is a lot of potential for strain identification in metagenomics. One potential area of application that I will discuss in some more detail to showcase the potential of sub-species level analysis is wastewater metagenomics. I will address both the potential for wastewater surveillance of antibiotic resistance and pathogen outbreak tracking.

Surveillance of antibiotic resistance genes (ARGs) is essential in informing strategies and interventions to curb the spread of antibiotic resistance. In line with the WHO's One Health approach, it is essential to integrate different environments in the surveillance of antibiotic resistance[50] (one health refers to the belief that different ecosystems such as humans, agriculture, and natural environments are strongly interconnected and a holistic understanding is required to advance ecological and public health[51]). There are however several hurdles to effective surveillance. Individual level surveillance is focused on hospital environments, expensive, and, especially in low and middle income countries, not standardized[8]. On the other hand, metagenomic surveillance of wastewater can be much cheaper; for Kenya, Aarestrup and Woolhouse[8] estimated a cost of 0.1% of the cost of isolate based testing. Additionally, surveillance of pathogens may be possible at the

same time as ARG surveillance[9] and wastewater surveillance potentially gives a view of ARG prevalence across a human population[8]. Multiple studies on antibiotic resistance genes found in wastewater have been published[52,10,53,54,11]. However, several open questions remain. How well ARGs found in wastewater reflect clinical resistance is a central open question, and different studies point to different conclusions. Pärnänen et al.[52] and Hutinel et al.[11] both showed that resistance levels in wastewater correlate with clinical resistance. However, neither of these studies used shotgun metagenomics, with Pärnänen et al.[52] utilising qPCR and Hutinel et al.[11] culturing *E. coli* isolates. On the other hand, Karkman et al.[53] investigated metagenomic data from wastewater and found only limited correlations between ARGs in wastewater samples and clinically observed frequencies. This may be due to multiple factors. One possible conclusion is that wastewater data may not represent clinical resistance well, after all. However, the lower correlation may also be due to the challenges presented by metagenomics. Importantly, Karkman et al.[53] were not able to assign ARGs to species. However, antibiotic resistance varies widely by species and drug class[55]. The ability of assigning ARGs to species may therefore provide the necessary resolution to improve the prediction of clinical resistance. Additionally, Hendriksen et al.[10] found that ARG prevalence is only explained to a limited degree by taxonomic composition. This indicates that relying on known ARG-species association (as was also done by Karkman et al.[53] for the *intI1* gene for *E. coli*, showing again a limited correlation to clinical resistance prevalence) may not be sufficient. Instead, ARG-species and ARG-strain associations need to be inferred directly from metagenomic data. If successful, this may provide the higher resolution required to answer the question of whether can conduct effective surveillance of clinical resistance via wastewater sampling and shotgun metagenomics. An interesting first step in this direction was taken by Munk et al.[54]. Munk et al.[54] performed metagenomic assembly and the resulting contigs were processed for the presence of ARGs. ARGs with less than 1kbp flanking regions were filtered out. The remaining ARGs were classified as either plasmid or non-plasmid related. While a significant proportion of ARGs were plasmid associated, many ARGs for which a host association was possible remained. Munk et al.[54] identified host genera via the flanking regions, finding ARGs that were both shared between different genera and ARGs unique to some genera. They further conducted an investigation that enabled them to identify which ARGs are likely highly transmissible, and which are more likely to be vertically transmitted (i.e. through bacterial replication). These results are highly valuable, but Munk et al.[54] do not present an association with clinical data. Conducting a regression analysis with the ARGs associated to specific genera or species, as done by Karkman et al.[53], would have been valuable to see if the predictability could be improved. Additionally, genus level assignment of ARGs still does not yield the same information as a species or even strain level assignment would have. In their study, Munk et al.[54] conducted a genus-ARG co-occurrence analysis following the assignment of the genes. One could however easily imagine reversing the analysis steps (i.e. first determining the strains in a sample)

and computing a co-occurrence network between clones or strains and ARGs to further elucidate the likely associations of ARGs. Additionally, combining the approach of Munk et al.[54] with the assembly of plasmids may be worthwhile. While assembly of plasmids is challenging[56], as is the assignment of plasmids to their hosts, potential lies in the fact that different plasmids show different levels of host specificity[46]. Furthermore, there is a high correlation between GC-contents in bacterial species and the plasmids they carry[57]. However, the results of such an analysis may or may not be informative and a combination of approaches may be valuable. Further investigation into this is clearly warranted. In summary, there is potential of the identification of specific strains and the assignment of specific genes to these strains. There is a clear need for further development of methods serving this purpose. This is not only true for metagenomic investigations of wastewater. Surveillance of other environments is also possible[58] and advancements in metagenomic methodologies, including assigning ARGs to species and strains can be beneficial here.

Beyond surveillance of wastewater for antibiotic resistance, pathogen surveillance is also possible using wastewater[9] and strain level analysis can support this endeavor. Hendriksen et al.[9] showed that pathogens can be detected in metagenomic data from an informal settlement in Kenya, where individual data is often not available. Notably, they identified *E. coli* as one of the pathogens, but did not discuss whether the found *E. coli* strains were pathogenic or potentially commensal or environmental. Such an analysis would have provided further detail of whether the presence of *E. coli* could be clinically relevant, pointing towards the importance of strain-level analysis. Hendriksen et al.[9] further discussed the potential of associating clinical or household reported cases of diarrhea and fever with the metagenomic data, however, the cases observed were to few to do so successfully. Nonetheless, such associations could provide insight into pathogens involved in potential outbreaks in informal settlements. Here, strain level analyses will again be beneficial, as it would enable the tracking of strains across samples, and thereby predict the burden of a pathogen associated with clinical cases. Beyond this fascinating study, strain level surveillance has long been fruitful outside of metagenomics. Whole genome sequencing studies on isolates obtained from outbreaks were able to identify strains and their reservoirs in outbreaks[59,60]. Douglas et al.[59] conducted a study of an outbreak of vancomycin-resistant *Enterococcus faecium* and among 26 isolates from 24 patients found only minimal variation when analyzing single nucleotide polymorphisms (SNPs) in the core and accessory genome, with 17 of the isolates belonging to the same strain based on core genome analysis (the core genome being the set of genes shared between all members of a species). Similarly, Johnson et al.[60] found highly similar isolates during an outbreak of *Sphingomonas koreensis* investigating both Mash distances[61] (a measure of the genetic distance between two genomes) as well as SNPs in the core genome. Using whole genome sequencing they were able to establish sinks and water in the hospital environment as likely source of the *S. koreensis*

infections. Naturally, whole genome sequencing of cultured isolates provides greater coverage of the genome and a better ability to identify SNPs than metagenomics. However, culture based approaches also require a clear target, whereas metagenomics can serve to surveil the microbiome in critical areas, such as sinks in a hospital environment, more broadly. By being able to identify both the core genome and estimate genetic distances between strains (such as the Mash distance) in metagenomic samples reservoirs of pathogens could be identified more readily, especially when metagenomic surveillance is employed routinely. Broad routine surveillance via culturing approaches is not as easily achievable considering that a clear target species for culturing is required, similar to clinical metagenomics where a differential diagnosis is required for culture-based analysis[62]. Yet, with metagenomic surveillance data readily available, outbreaks could be traced back after they are detected, utilizing previously collected data. Both metagenomic and culture based approaches could also be combined, with promising results already demonstrating the potential of reservoir identification[63]. Finally, multi-locus sequence typing (MLST) has long been a successful method for sub-species analysis based on a set of a few loci conserved in a species. MLST relies on the identification of the nucleotide sequence of select loci in a few housekeeping genes (typically 6 to 10)[64]. The different previously observed sequence variants of the genes (alleles), as well as the combinations of alleles making up different strain types, are deposited in an online database. MLST typing originally offered the advantage of being relatively easy to conduct. Hence, the loci chosen are fairly short and fairly few[64]. The need for housekeeping genes to be present in all members of a species, as well as the choice of these genes as ones that are under stabilizing selection, can make the MLST strain type a good proxy for studying the lineage of different isolates[64]. On the other hand, it is not always clear what information MLST provides about the phenotype of a strain type. While sometimes there are clear correlations[65,66], other times these are mostly absent[67,68]. Nonetheless, MLST analyses have proven useful. MLST has helped in outbreak tracking[69,70,71] and epidemiological surveillance[72,73] in the past, clearly highlighting the usefulness of sub-species level pathogen surveillance. The public database pubMLST[74] additionally enables comparison of strains from across the world. Extending sub-species analysis to metagenomics can provide similar insights in a much broader regime, addressing many species at the same time.

Strain Identification Tools

Conducting informative metagenomic strain level investigations firstly requires the ability to identify the strains present in a sample. Many different approaches are possible when investigating the sub-species level in metagenomics, and different tools have been developed. However, their explicit goals do vary. One

can roughly split the available tools into three categories. The first set of tools, including DESMAN[75], ConStrains[76], or metaPhlAn[77], is aimed at tracking strains across multiple samples. While tracking of strains across samples is essential, several practical and conceptual considerations have to be made. Firstly, using multiple datasets as input, while increasing the available data, has the disadvantage of the underlying assumption that strains across samples will be shared. If this is not the case, it will be fairly easy to draw the wrong conclusions, especially considering that tools such as DESMAN show only limited accuracy on single samples. Secondly, the published tools have their own limitations. For example, DESMAN and EVORha[78] rely on MAGs for strain inference. When these are not of high quality this may impact the results, as Quince et al.[75] report themselves. The second set of tools is aimed at identifying SNPs of the strains within a sample. There are various tools available for this purpose, which have been benchmarked by Andreu-Sánchez et al.[79]. Identifying SNPs is essential to track mutations and infer phylogenies. However, the calling of SNPs is often done is isolation from reference sequences. As such, only limited information can be gathered on the relation of the SNPs to known whole-genome sequencing derived phylogenies. Additionally, inferences on the gene content of specific strains are hard to make on this basis. Even when a reference sequence, or a small set of reference sequences, is used, choosing a high-quality reference isolate that is closely related to the sequences that need to be typed is essential and the inference of SNPs may be strongly dependent on which reference sequences are chosen[80]. Indeed, it has been shown that the choice of reference strain has strong impacts on the SNPs that are called as well as on downstream analyses, such as the construction of phylogenetic trees[81]. Nonetheless, calling of SNPs in metagenomes is of great importance, though strain recovery prior to SNP calling may provide clear benefits, as it will provide the best reference sequences relative to which to call SNPs. The last set of tools, and the one of most interest to us, aims to recover known strains, or known strains closely related to those in the samples, where strains correspond either to unique reference sequences or clusters thereof, depending on the tool. Importantly, this approach is suited for single sample analysis and can take into account population structure and our knowledge of SNPs and gene content, while not precluding further SNP-analysis downstream. In Chapter II, I present a benchmarking study of tools with this focus and in Chapter III and Chapter IV I will present the development of a novel tool in this category. Hence, I will briefly summarize the major tools previously published in this area. Tools that are aimed at tracking strains across multiple samples, and hence, are not designed for single sample analysis were not considered. Additionally, tools were excluded tools that might provide strain information (such as Kraken2[82]) but are not specifically designed for this purpose and instead focus on lower resolution taxonomic assignment. Tools aimed at reconstruction of MLST types (such as metaMLST[83]) were also excluded. Lastly, tools that only work using a single reference genome were excluded, as these do not take into account known population structure. I will briefly describe the tools considered here in order of publication date.

StrainEst StrainEst[84] aims to give a sparse set of known reference genomes explaining the observed reads in a metagenomic dataset. To this end, StrainEst first builds SNP profiles by selecting a subset of representative genomes from a species database (obtained by clustering of the genomes) and aligning these against the species reference. SNP profiles are then created using these alignments. Secondly, StrainEst selects a second set of genomes as mapping representatives, aligning them to the species reference and then aligning the metagenomic reads to this set of genomes. From this, nucleotide frequencies at the SNP positions determined previously are extracted. StrainEst finally uses a lasso-regression which estimates the relative abundance of each representative genome to predict the observed nucleotide frequencies. The conceptual simplicity and requirement for sparseness are strengths of StrainEst. On the other hand, there are some weaknesses. Firstly, the reliance on a set of representative genomes for each cluster naturally reduces the taxonomic resolution obtainable. The alignment of reference assemblies to a species representative performed to infer SNPs may also be sensitive to the representative chosen. Additionally, the total proportion of strains is not constrained to $[0, 1]$ during estimation (though notably the regression coefficients representing the relative abundances are scaled to unit norm after estimation). Moreover, assuming uniform coverage of a given strains' genome in a metagenomic dataset, it is clear that if this strain is only present at a relative abundance of less than one, but is also the only strain in the sample carrying a certain gene, that the observed frequencies will be skewed, potentially impacting the estimation process (see Figure 3.2 for an illustration). Nonetheless, StrainEst shows good performance on the datasets presented by Albanese and Donati[84].

KrakenUniq Both Kraken[85] and Kraken2[82] represent widely used tools for taxonomic classification of metagenomic data. Both tools work by counting k-mers shared between the metagenomic reads and a database. Each read is then assigned to the lowest common ancestor of all database entries that share the same k-mer count. This conservative approach makes the tools not ideal for strain identification. KrakenUniq[86] aims to address this issue and provide higher level resolution by using an efficient algorithm to estimate the count the unique k-mers for each database taxon. Notably, one needs to choose a cutoff for the number of unique k-mers assigned to a taxon for it to be inferred as present to control the false positive rate[87]. This cutoff varies both with the taxonomic level of interest and the sequencing depth. This will make it hard for researchers to choose appropriate cutoffs for their datasets. On the other hand, KrakenUniq has the benefit of being able to incorporate a large database, containing taxons from across bacteria, viruses, plasmids, humans, and archea. By default this is either a pre-constructed database or constructed from available genomes on NCBI. This database is further customizable using individual sequences.

MIST MIST[88] takes yet another approach to strain identification. Similar to StrainEst a set of genomes representative of genome clusters is generated via complete-linkage hierarchical clustering at different ANI thresholds. However, in contrast to StrainEst, MIST then aligns each read to the whole set of reference genomes, determining the number of nucleotide mismatches between each read and each reference genome. MIST then proceeds to select the minimum mismatch among all reference genomes among a cluster to obtain a mismatch for each cluster at each ANI threshold. These mismatches are then fed into a probabilistic model, where maximum likelihood estimation of cluster frequencies are obtained. Even though the methods differ, the strengths and weaknesses are similar to StrainEst. However, MIST conceptually improves on StrainEst in two areas. Firstly, the probabilistic nature of the model is a more natural approach to the estimation of proportions. Secondly, MIST does not rely on the mapping to a single representative reference genome per cluster, but instead performs a full alignment to all members of a cluster. In this way, the intra-cluster variation is still somewhat taken into account and the reliance on a reference genome is reduced.

StrainGE StrainGST, part of the StrainGE toolkit[89] (I will refer to the toolkit from hereon), works on a k-mer basis to identify strains. For the database creation a k-mer profile is created for each reference strain. Then, the reference strains are clustered with single-linkage hierarchical clustering using the Jaccard distances of the k-mer profiles. Lastly, genomes are removed that share over 99% of k-mers with another strain. Having established the database, the metagenomic sample is also k-merized. StrainGE then reports present strains iteratively. In each iteration a score is calculated for each strain based on the fraction of k-mers in the reference strain present in the sample, the count of k-mers in the sample as a fraction of the count of k-mers in the reference strain, and a measure for the evenness of coverage of k-mers matching to the reference strain across the genome. The strain with the highest score in each iteration is reported as present. The process terminates either after 5 iterations (by default) or when no more strain has a score above a set threshold. The StrainGE toolbox includes a second tool, strainGR. This tool allows for further analysis of the identified strains by identifying SNPs and large deletions. This is achieved through alignments of reads to the concatenated reference genome of the reference strains found by StrainGE. Notably, SNPs are only called in regions that are present in a single reference strain. This is done by filtering out reads that align ambiguously. However, this severely limits the scope, as even a read aligning ambiguously to two reference strains could still bear a SNP compared to a third reference strain present. Large gaps and possible deletions are identified as those regions above a given length (default 5000bp, but scaled according to the estimated coverage of the reference strain) to which no read aligns. This can identify potential deletions in a statistically sound manner. Insertions cannot be identified though. However, the identification of these is essential in practice as it can be relevant for the initial strain estimation and can present information on

acquired genes.

Sigma Sigma[90] is an alignment based tool in which the metagenomic reads are aligned to a custom reference database. A probabilistic model utilizing maximum likelihood estimation is applied. This model estimates the probability of each reference genome, utilizing the probability of obtaining the observed metagenomic reads (with the observed number of mismatches) given by the alignment. A distinguishing feature of Sigma is that the method enables the estimation of both P-values and confidence intervals, which can be highly valuable for researchers wishing to scrutinize their results. None of the other methods presented here have this capability.

StrainScan The most recent tool published is StrainScan[91]. StrainScan works on a k-mer basis in combination with a hierarchical clustering approach applied to the reference strains. The clustering is performed via Dashing[92] using single-linkage hierarchical clustering based on a Jaccard distance matrix. In the search tree constructed, each node is assigned a set of unique k-mers. To identify leaf nodes (strains) present in the sample, representing individual reference strains, a breadth-first search is conducted. Each node is assigned a score based on the fraction of k-mers in the sample matching the unique k-mers of the node and the relative fraction of k-mer counts among the matching k-mers, similar to StrainGE. Individual strains are reported via a threshold. The analysis of the performance of StrainScan is the most thorough among the reported tools, including multiple number of strains, various species, and strains contained and not contained among the reference strains. Here, Liao et al.[91] report that StrainScan outperformed all other tools analyzed in most categories. Lastly, they also performed an analysis of two mock data sets from the Human Microbiome Project[93]. Notably, StrainScan, while still performing well and returning strains closely related to the ones contained in the samples, did not outperform all of the other tools here, similar to simulated data for strains not among the reference strains. This points out that it is not clear which tool is best suited in which situation, and further comparisons are necessary.

Integer Linear Programming

So far, I was mainly concerned with the biological basis of this thesis. Now, I will present one of the major mathematical tools underlying the work presented in Chapter III and Chapter IV, mixed integer linear programmes (MILPs). MILPs are very useful for efficiently solving problems arising in genomics. They are frequently used in research for purposes such as multiple sequence alignment, assembly of sequences, or

protein folding[94]. Their usability comes both from the fact that a wide variety of problems can be translated into MILP formulations with relative ease and that highly efficient methods have been developed to solve these MILP problems. I will give a short overview of the basic theory behind them here, while more details can be found in the books by Vanderbei[95] and Michele Conforti[96]. To understand MILPs, I will first lay-out linear programmes (LPs). An LP maximizes a linear objective function $\zeta = cx$, where c is a row vector of fixed weights and x is a column vector of n decision variables we want to estimate to optimize ζ. Additionally, an LP contains linear constraints of the form $Ax \leq b$ for some $m \times n$ matrix A and a column vector b of m constraints. These constraints limit the range of values that the decision variables can have. A full LP can then simply be written as

$$\begin{aligned} \max \quad & cx \\ \text{subject to} \quad & Ax \leq b \\ & x \geq 0. \end{aligned}$$

Note the additional constraint of $x \geq 0$. Efficient methods for finding optimal solutions to LPs have been developed and enable us to solve even very large LPs in a reasonable amount of time. Before diving into the complexities introduced by extending LPs to mixed integer linear programmes (MILPs), I will briefly address one basic approach to solving an LP that illustrates the process well.

Solving LPs

Over time, many different approaches to solving LPs have been proposed. Some of these guarantee polynomial runtime, however, I will focus here on the Simplex method that performs well in most cases and is used across a wide range of problems. Here, I give an intuitive outline of the simplex method, while Vanderbei[95] can be consulted for a more formal treatment. The simplex method relies on a simple realization about the space of feasible variable and objective function values. Firstly, take the parameter space $\mathbb{R}^n$ of all values for the n variables in our problem with each of the n dimensions representing the value of one of the variables. Now the jth added constraint $\sum_{i=1}^{n} a_{j,i}x_i \leq b_j$ (where $a_{j,i}$ are the elements of A) partitions this space into two sections, one where the variables violate the constraint and one where they do not (Figure 1.1A). Note that the linearity of our constraints guarantees that the section with feasible variable values is convex (the constraints bound a convex polytope). Intuitively this means that any two points within the section can be connected by a straight line which completely lies inside the section as well. This holds true whenever a new constraint is added (Figure 1.1B). Additionally, note that any objective value $\zeta = cx$ for x with some set

values also describes a linear hyperplane and that for any given hyperplane $\zeta = cx$ all parallel hyperplanes to one side will have a greater value and all the ones to the other side a smaller value. From this, it can easily be seen that if ζ does not pass through one of the vertices of the convex polytope defined by the constraints we can pick a value for ζ with a more extreme value. Part of this hyperplane still lies within our feasible parameter space, and hence, has values for x that fulfill the constraints. If the hyperplane passes through at least one vertex, however, we have an extremal solutions (also called a basic feasible solution). Hence, one of the vertices must be the optimal value for our problem (Figure 1.1C).

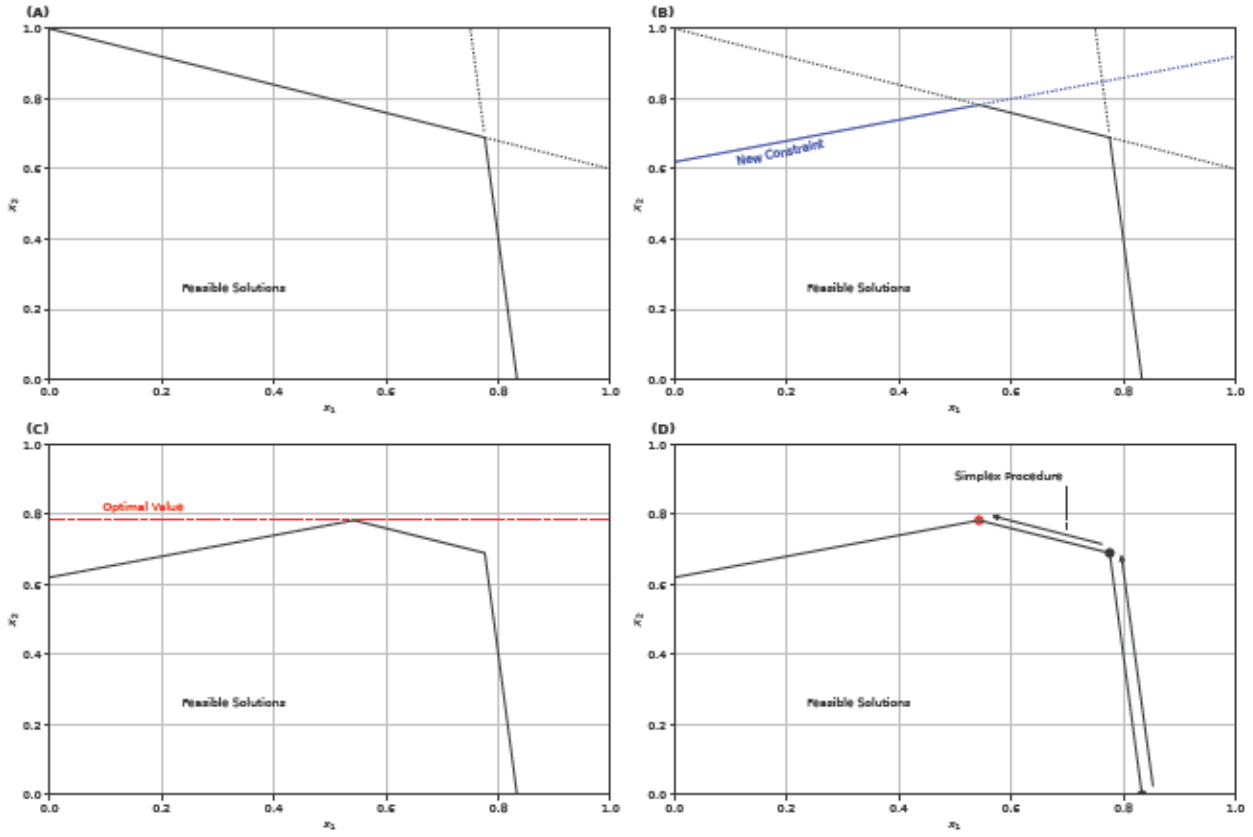


Figure 1.1: Illustration of the simplex method for two variables x_1 and x_2, with $x_1, x_2 \geq 0$. **(A)** Constraints define an area of feasible values for x_1 and x_2. Here, each line represents a constraint and solid parts of the lines delineate the space where both x_1 and x_2 have a feasible value. **(B)** The space of feasible values stays convex when adding a new constraint. **(C)** The optimal objective value $\zeta = cx$ achievable with allowed variable values lies at a vertex. **(D)** The simplex method starting out at one vertex (lower right) and proceeding along the vertices until the optimum is reached (red dot).

The simplex method (illustrated in Figure 1.1D) works by first introducing slack variables which are added to the constraints such that the inequalities turn into equalities ($\sum_{i=0}^{n} a_{j,i}x_i \leq b_j$ is converted to $s_j + \sum_{i=0}^{n} a_{j,i}x_i = b_j$). Then we proceed to find any basic feasible solution, which is often easily achieved by setting all variables x_i to 0. In cases where this leads to non-negative slack variables, a different approach is needed, as described for example in Vanderbei[95]. The simplex algorithm then proceeds to neighboring

vertices until all vertices reachable from the current vertex have lower objective values. This is done by taking a variable which would increase the objective function if it was changed (it has a positive coefficient in the objective function) and fixing another variable to 0. The vertex at which we cannot choose a variable whose change would increase the objective function is then the optimal one.

An important concept is the dual linear programme defined as

$$\begin{aligned} \min \quad & by \\ \text{subject to} \quad & yA \geq c \\ & y \geq 0. \end{aligned}$$

The minimization here is achieved by simply maximizing the negative of the objective function and adapting the constraints accordingly. Importantly, the dual of the dual is the original linear programme, the primal. The dual linear programme has the property that its optimal value is an upper bound on the primal linear programme (called the weak duality theorem). Hence, if we solve both the primal and the dual programme and obtain the same objective value we know that the solution to the primal programme is optimal. It has been proven that if the primal programme has a solution, so has the dual programme and the optimal values of the solutions are the same. This is known as the strong duality theorem. This also implies that we can solve the dual problem using the simplex method (called the dual simplex method) and obtain an optimal feasible solution for the primal LP. This is useful in many cases, as it leads to an easier approach for solving the primal. For example, if the primal LP has no feasible solution with all $x = 0$, i.e. if we have a constraint of the form

$$\sum_i a_{i,j} x_i \leq -b, \tag{1.1}$$

where $b > 0$, we might struggle to find an initial basic feasible solution. However, we may observe that the dual problem does have a feasible solution in the origin ($y = 0$) and by solving the dual we arrive at our desired optimal solution. The application of the simplex method contains a few more intricacies, the details of which would go beyond the scope of this outline, but can be found in Vanderbei[95]. Lastly, it is worth mentioning that while the simplex method is generally very efficient, it has been shown that in the worst case the runtime can increase exponentially with the problem size. However, in general, first through the ellipsoid and later through interior-point methods, it has been shown that LPs are solvable in polynomial time.

Mixed integer linear programming

A mixed integer linear programme is simply a linear programme in which some of the variables are fixed to be integers, i.e.

$$\begin{aligned} \max \quad & cx + dy \\ \text{subject to} \quad & Ax + By \leq b \\ & y \geq 0 \\ & x \geq 0 \\ & x \in \mathbb{N}_0, \end{aligned}$$

where $\mathbb{N}_0$ denotes the natural numbers including zero. Upon first inspection, one might be tempted to assume that solutions to such an MILP are also easy to find. However, it turns out that in general this is not the case, and an MILP (with pure integer linear problems (ILPs) in particular) is NP-complete as can be easily seen by the reformulation of the minimum vertex cover problem into an ILP[97].

$$\begin{aligned} \min \quad & \sum_{v \in V} x_v \\ \text{subject to} \quad & x_v + x_u \geq 1, (u, v) \in E \\ & x \in \{0, 1\}, v \in V, \end{aligned}$$

where V are the vertices of the graph in question and E are the edges. This example also serves to clarify the notation used throughout this thesis. Instead of specifying two full constraint matrices A and B as above constraints are separated and given for the variables whose contribution to the constraint is non-zero. The variables are indexed and the specification of the set defining those indices is given after each constraint. Similarly, I will also separate the objective function into interpretable parts.

Notwithstanding the complexities of solving MILPs in a short amount of time in general, much progress has been made, making many MILPs efficiently solvable such that they become a powerful tool. There are several ways of generating a solution for an MILP. One of the central methods is branch-and-bound, while another is to use cutting planes. Both approaches work by performing a relaxation of the MILP to an LP and solving this LP to obtain a tighter estimate for the solution of the MILP. I will describe both approaches in turn below. These and other ingenious approaches have been implemented many times in open source and commercially available solvers and the different approaches can be used together by these solvers to find an

optimal solution. One of the fastest solvers currently available is CPLEX by IBM, which is used throughout this thesis. While being a commercial product, a free academic license of CPLEX is available for researchers, making it still an accessible tool within the research community while retaining the improved performance over non-commercial and open source solvers.

Branch-and-Bound

The branch-and-bound method works by creating an LP-relaxation of the MILP, solving this relaxed LP, and then either further constraining the MILP to exclude some of the non-integer decision variable values or accepting the relaxed solution if all it's integrality constraints are fulfilled. Specifically, an LP-relaxation of an MILP is of the form

$$\begin{aligned} \max \quad & cx + dy \\ \text{subject to} \quad & Ax + By \leq b \\ & y \geq 0 \\ & x \geq 0, \end{aligned}$$

i.e. we drop the constraints on integrality. The solution to this LP will be easy to determine, for example using the simplex method. However, not all decision variables that need to be integers are guaranteed to be. Nonetheless, we know that a solution to the relaxation will have an objective value at least as big as the solution to the MILP. Hence, we can now set an upper bound to the objective value of the MILP. If on the other hand the solution of the relaxation is integral, then we have also found a solution for the MILP. It is when the solution is not integral that branching comes into play. We pick one x_i that is not integral, optimized to a value of z, and add a new constraint to the model that will constrain it such that solutions with $x_i = z$ are not feasible. We have two options to succeed with this using integer constraints, adding either $x_i \geq \lceil z \rceil$ or $x_i \leq \lfloor z \rfloor$ ($\lceil z \rceil$ and $\lfloor z \rfloor$ denote the smallest integer not smaller than x and the largest integer not bigger than z, respectively). Including either of these two constraints leads us to two new MILPs for both of which the previously found solution will not be feasible. Hence, we branch our search for an optimal solution into two parts, as illustrated in Figure 1.2. For each of these new nodes, we again create an LP-relaxation, solve it, and repeat the process, creating a search tree. The search is completed when we find an integral solution to the LP-relaxation that is optimal. This is where bounding the search tree comes into play. In total there are four possible scenarios following the optimization of the LP-relaxation. Firstly, the LP may be infeasible. In this case the current branch of the search tree gets bound and we do not continue our search

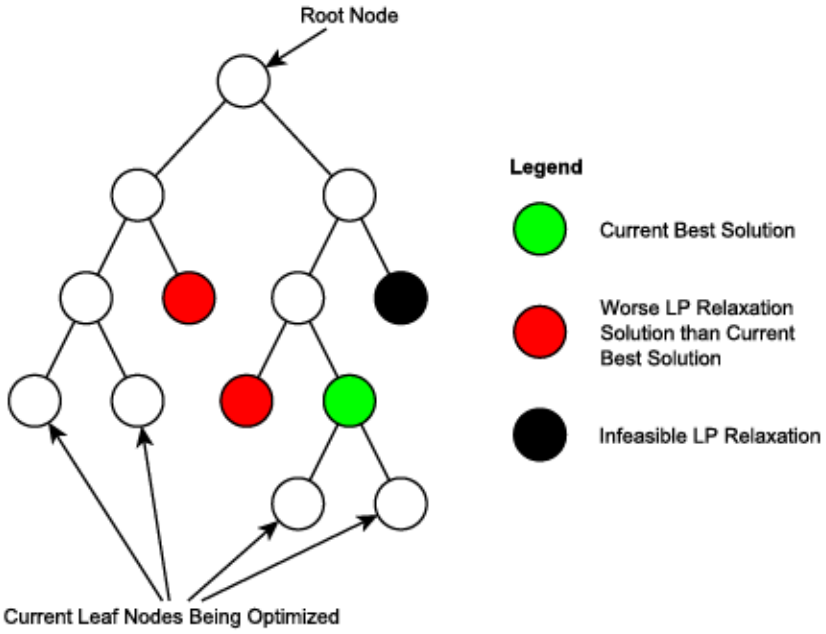


Figure 1.2: Illustration of the branch-and-bound algorithm.

along it. Secondly, the solution to the LP may not be integral, in which case we add the new constraints and expand the search tree. As a third option, the LP may give us a solution that is integral. This solution will have some objective value lower than or equal to our upper bound. Additionally, it can be lower, equal, or higher than the integer solutions we have found so far. If the objective value is lower than the highest objective value we have found so far for an integer solution along our search tree we can again bound the current branch we are on, as the other solution is better. Lastly, we can decrease our upper bound to the lowest objective value of all the LP relaxations explored at the moment. We know that we have found an optimal solution to the MILP when the current upper bound matches the current lower bound. This is also referred to as closing the gap, where the gap is defined as

$$\frac{|\text{upper bound} - \text{lower bound}|}{|\text{lower bound}|}.$$

In practice, solutions with a low gap may give optimal answers, as the only reason they are not provably optimal is that we have not gotten our final upper bound through the LP relaxations. Hence, sometime it can be useful to terminate the MILP optimization at a non-zero gap. This may be hard to determine, however, one can experiment with this technique, allowing solutions with non-zero gaps and investigating their quality.

Cutting Planes

The second common approach is that of cutting planes. Several methods for creating cutting planes exist. In general, upon generating a solution to the LP relaxation of an MILP a cutting plane method will add a linear constraint that is not fulfilled by the solution to the LP relaxation but is fulfilled by all feasible integer solutions. This then leads to a new programme that we can solve again using its LP relaxation. One common version of cuts are Gomory fractional cuts. We observe that these cuts lead to negative constraints, hence, we make use of the dual simplex method to arrive at a solution to the new LP relaxation. I will not describe Gomory fractional cuts in detail, as for a clear presentation several more concepts would need to be introduced. However, a simple intuition is given in Figure 1.3 and by the following example. We first find a solution to the LP relaxation (Figure 1.3A). We then find a fractional constraint that is fulfilled by the solution to the LP relaxation of the form $\sum_i a_{i,j} x_i \leq f$, with f fractional (Figure 1.3B). When we find such a constraint, for which the $a_{i,j}$ are integer, then we know the optimal integer solution must also fulfill $\sum_i a_{i,j} x_i \leq \lfloor f \rfloor$, which can be added to the problem constraints, while the original LP will not fulfill this constraint by its construction (Figure 1.3B). This then provides us with an improved ILP formulation we can solve. For details on the procedure in general the reader can refer to the textbook by Michele Conforti [96].

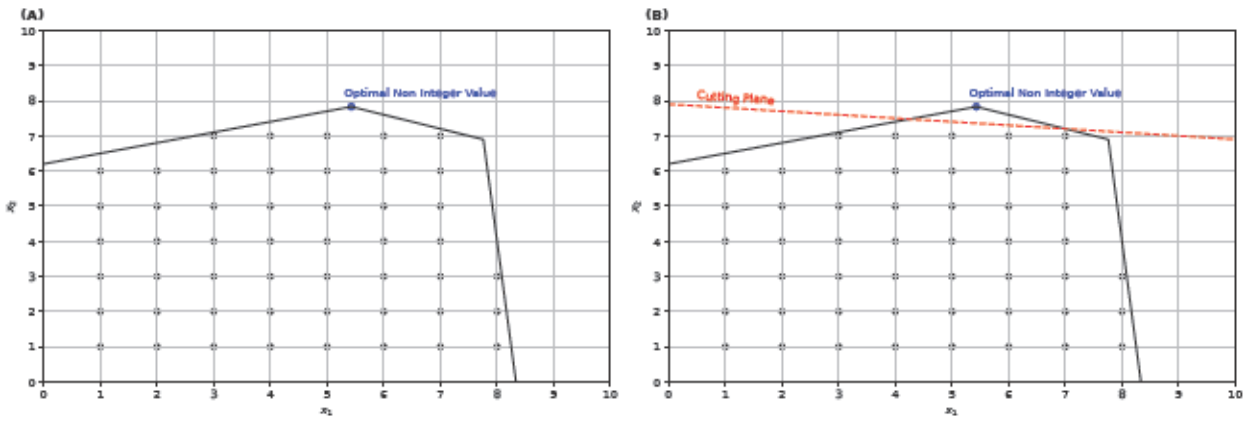


Figure 1.3: Illustration of the cutting plane method. Black dots represent integer valued solutions. **(A)** The optimal value of the LP relaxation (blue) is non-integer valued. **(B)** Introduction of a cutting plane (red line) making the solution to the LP relaxation infeasible, while keeping all integer valued solutions feasible.

Techniques for setting up ILP formulations

When translating a problem into an MILP formulation several situations will be encountered repeatedly, hence, while explained in the method sections, I will give a brief overview here of a few basic concepts. Firstly, we can make use of so called Big-M constraints [98]. Say we have a constraint of the form $\sum_{i=0}^{m} a_{i,j} x_i \geq b_j$. In

this case, when we initialize all x_i to 0 we cannot get an initial feasible solution, making the simplex method hard to apply. We can remedy this by introducing a variable d_j, such that $d_j - s_j + \sum_{i=0}^{m} a_{i,j}x_i = b_j$, where s_j is the corresponding slack variable. Now, we include d_j in the objective function with a large enough constant M. $x = 0$ will now be an initially feasible solution and the optimal solution will have $d_j = 0$, if M is set big enough. Notably, the use of the Big M method for initialization has caused some concern in the research community[99], but introducing a big constant M also has other uses. Say we want to constrain all but one x_i to be 0. If we introduce M and binary variables z we can obtain the constraints

$$\begin{aligned} x_i &\leq Mz_i, i \in \mathcal{I} \\ x_i &\geq -Mz_i, i \in \mathcal{I} \\ \sum_{i \in \mathcal{I}} z_i &= 1, \end{aligned}$$

with $\mathcal{I}$ some set of indices. Then x_i is constraint to be 0 for all but one instance in which it can vary from $-Mz_i$ to Mz_i and so is practically unconstrained for a large enough M. For binary variables x_i we can further drop M and the second constraint above. This idea will be of importance in Chapter III.

Secondly, logical constraints, such as boolean AND and OR relations, are often important[100]. For example, one might require that a binary variable takes on the value $X = 1$ if at least one of the variables Y and Z is equal to 1. This can be achieved by introducing constraints

$$\begin{aligned} X &\geq Y \\ X &\geq Z \\ X &\leq Y + Z, \end{aligned}$$

which can be extended freely to more variables than Y and Z. Similarly, if we want $X = 1$ if $Y = 1$ and $Z = 1$, we can add the constraints

$$\begin{aligned} X &\geq Y + Z - 1 \\ X &\leq Y \\ X &\leq Z. \end{aligned}$$

Note that for both AND and OR constraints when X appears in the objective function of a maximization we can drop either the greater-than constraints (if X is included with a positive coefficient in the objective

function), or the smaller-than constraints (if the coefficient in the objective function is negative).

Furthermore, while conceptually we might distinguish between different solutions, these may be the same, or symmetric, to the MILP. For example, if we had two variables X_1 and X_2 indicating the relative abundances of arbitrarily indexed members of a microbial community two valid solutions could be $X_1 = 0.7$, $X_2 = 0.3$ or $X_1 = 0.3$, $X_2 = 0.7$. We can apply symmetry breaking to exclude one of these possibilities a priori by introducing the constraint $X_1 \geq X_2$. While these additional constraints increase the model size, they also decrease the size of the feasible region and may as such overall reduce runtime. Above, a simple symmetry breaking example was considered, however, we may encounter more complicated ones. Nonetheless, the basic idea remains the same. We aim to exclude all but one of the possible solutions that give essentially the same result. Similarly, we can enforce uniqueness in the MILPs. For example, we might estimate a set of basepair sequences. To ensure that these are all distinct we can constrain the MILP such that

$$d(s_1, s_2) \geq 1, \tag{1.2}$$

where $d(s_1, s_2)$ is a linear distance between the two sequences counting the number of differing nucleotides. Again, the approach will reduce the number of valid solutions and make it easier to find the optimal one. In practice, many solvers have automated symmetry breaking. However, while often helpful, observation showed that many times solving proceeds faster if symmetry breaking constraints are implemented a priori.

CPLEX

In the present work, CPLEX 22.1 was used to generate solutions for all MILP models. CPLEX is a software developed by IBM that provides methods for solving MILPs and a range of other optimization problems, including the simplex method, cutting planes methods, and branch-and-bound, among others. CPLEX is implemented in C++, and easily usable through a Python interface where users can specify an optimization problem by providing the objective function and constraints. CPLEX status as one of the two fastest MILP solvers[101,102] makes it an ideal choice as solver for the MILPs presented in this thesis. Importantly, there are various options available when running a solver for an MILP. Determining the ideal settings from theoretical considerations is hard, and the same processing parameters may not work well for different situations. In this thesis, for most parameters the default setting of CPLEX 22.1 were chosen. Parameter settings were tested over the course of model development and the following settings performed well empirically.

- `parameters.mip.tolerances.mipgap` = `0.05`: The maximum gap a solution can have.

- `parameters.mip.strategy.lbheur` = `1`: Applies local branching heuristic. When a new best integer solution is found, similar solutions are explored.
- `context.cplex_parameters.threads` = `8`: Number of threads used in branch and bound process.
- `parameters.preprocessing.symmetry` = `0`: Turns off automated symmetry breaking.
- `parameters.parallel` = `-1`: Uses opportunistic parallel search. Computations are multi-threaded were possible.

Group Testing

While the previous sections have focused on bacteria and their identification in metagenomic data, viruses represent yet another major group of organisms of vital interest to public health efforts. This has been made abundantly clear by the recent SARS-CoV-2 pandemic. In the wake of the pandemic, my collaborators[1] and I decided to tackle the identification of SARS-CoV-2 and other viruses in swab samples using PCR and group testing. In particular, the problem can be posed as follows: How do we test as many samples of potential carries of SARS-CoV-2 using as few tests and time as possible? The answer to this can be found in group testing methods. Group testing was first developed in 1943 by Robert Dorfman to enable increased syphilis testing of soldiers during the second world war[103] and has since been applied in a variety of fields, ranging from infectious disease testing[104,105,106] to DNA sequencing[107] and data forensics[108]. In general, group testing aims to reduce the number of tests needed to test a population of people compared to individual testing. To achieve this, samples from multiple people are pooled together in groups and the groups are tested as a whole. If a group tests positive we know that at least one positive sample must be contained in it. By cleverly distributing samples across groups we can recover which samples are positives. Group testing divides into two categories, adaptive and non-adaptive[109]. In adaptive group testing, as initially developed by Dorfman[103], multiple rounds of testing are conducted, with the design of later rounds depending on the results of earlier ones. On the other side we have non-adaptive testing, which our work is mainly concerned with. In a non-adaptive group testing design all tests need to be specified a priori and can then be conducted in parallel. A simple explanatory example is array testing[110,111,106], illustrated in Figure 1.4. Here, each cell represents one sample and groups are constructed across rows and columns, leading to $2\sqrt{n}$ total tests, where n is the number of tested samples. Then, all those samples are inferred as positive that are in both a positive row and column test. If there is only one positive sample it can be identified unambiguously. However, if there

[1] Leonid Chindelevitch, Michael A. Crone, and Paul S. Freemont

are more positive samples, some false positives may be inferred, as shown in Figure 1.4. Non-adaptive testing has the advantage of being less time consuming than adaptive group testing. This benefit is significant, as for some infectious diseases such as SARS-CoV-2 a testing round can take up to several hours. Additionally, the logistical overhead of keeping track of samples across rounds is reduced. Both of these factors led us to focus on non-adaptive group testing in our research.

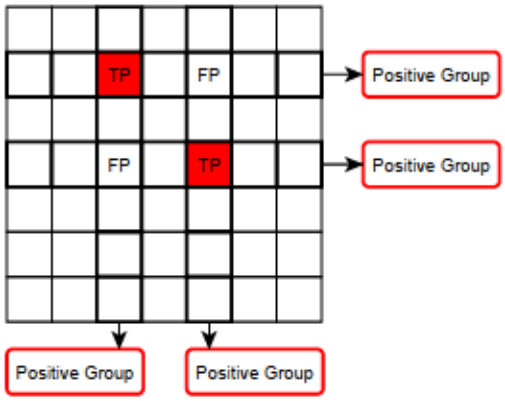


Figure 1.4: Array testing illustration. Here, each cell represents one sample from an individual. Samples are grouped along rows and columns. The red cells represent positive samples, which cause their groups, marked with thick borders, to test positive. A decoding method that declares every sample positive for which all its tests are positive then identifies two true positives (TP) and two false positives (FP).

When one considers using group testing there are both theoretical and practical constraints to keep in mind. On the theoretical side, group testing can introduce false positives at prevalence rates (probability of any one sample being positive, i.i.d.) higher than the design allows (as for example for two samples as shown in Figure 1.4). In fact, there is a maximum prevalence for which adaptive group testing can provide a higher rate (information provided per test) than individual testing. This maximum prevalence is $(3-\sqrt{5})/2$[112,113]. Note here, that this is only true in the small-error regime, in which there is a non-zero (but possibly asymptotically vanishing) probability of obtaining false positives or false negatives. Furthermore, no such rate can be achieved for non-adaptive group testing, and individual testing will always be optimal, as non-adaptive group testing incurs an error independent of the number of samples tested that is bounded below[114]. Nonetheless, for a given low error probability non-adaptive group testing can still be useful and potentially achieve the desired error rate or lower with less tests than individual testing[109]. Importantly, these results assume perfect test

characteristics. These results are also only proven for binary test outcomes, i.e. tests that are either positive or negative. When considering non-binary (noisy) test outcomes lower errors may be achievable, though it is unlikely that this would remove the bound on the error completely. Using non-binary test outcomes could improve detection because of the additional information provided by each test and because the tests are not independent. In particular, tests containing the same positive sample will have correlated test outcomes, with the dependence determined by the amount of virus in the sample. On the practical side there are further constraints we have to adhere to. Notably, we cannot simply combine as many samples as we want into a group or split a sample into as many groups as we want, as the resulting groups would then likely test negative due to the low amount of viral particles in each group. However, previous research provides promising results. Yelin et al.[115] showed that they obtained no significant loss in PCR test sensitivity, even when pooling 15 negative with one positive sample. Even when pooling 32 samples, reliable detection was still possible at a sensitivity of 90%. Other studies[116,117] confirm these positive results. On the other hand, some studies suggest higher sensitivity losses[118]. Nonetheless, Bateman et al.[118] still show a retainment of over 90% sensitivity in group sizes up to 10. Hence, we consider group sizes of less than 10 as practical. This is further supported by studies addressing 2-stage adaptive group testing for SARS-CoV-2 and obtaining positive results[119,110]. Additionally, ease of conducting a test design within a clinical laboratory needs to be considered. Clinical laboratories often have fixed equipment and processes in place. Ideally, these laboratories would be able to switch to group testing easily. One particular constraint is the size of PCR-testing plates, where each plate usually contains 96 wells for testing in parallel. To maximize time efficiency any design should maximize the usage of these 96 wells. Lastly, any design that is implemented in practice should be easily extendable to adapt to varying prevalence rates.

With the practical constraints in mind we focused our work on the shifted-transversal design, which is a generalization of the array testing design shown in Figure 1.4. It has the advantage of being conceptually easy to understand and being adaptable to different prevalence rates via simple extension of the design. Different group sizes can also be chosen. Shifted transversal designs are classified by the property of d-disjunctness. This property guarantees that given a particular decoder (i.e. a method to identify positive samples given the group test results) and error-free tests, we can identify up to d positive samples without incurring false positives or negatives. A precise definition of d-disjunctness requires the introduction of some formalism of group testing. In group testing a set of n positive or negative samples X_j are given whose binary status is to be inferred. The samples are split and placed into t grouped tests specified by a binary design matrix $M \in \{0,1\}^{t \times n}$. Sample X_j is placed into test i if $M_{i,j} = 1$. The ith entry of the result vector Y is

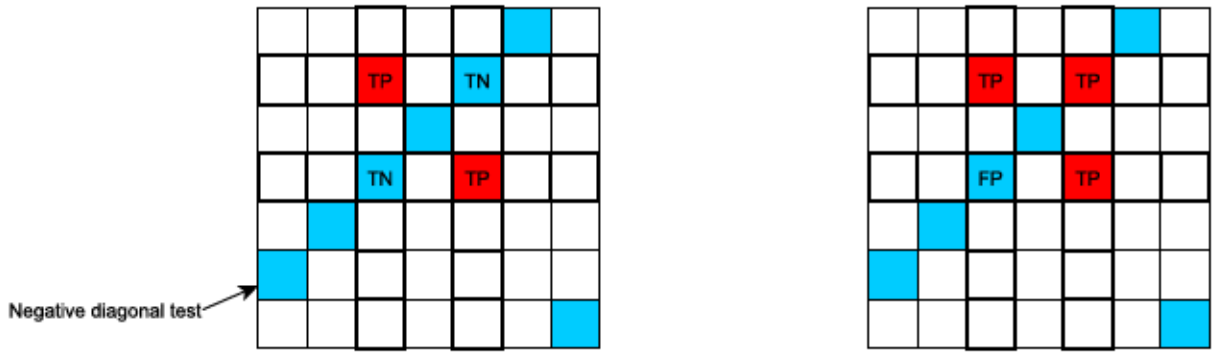


Figure 1.5: Illustration of d-disjunctness. On the left, the same samples as in Figure 1.4 are positive and inferred as such (labeled TP). However, we now also group samples along diagonals, increasing the disjunctness by one. This is shown for one example test, where the samples in that test are marked in blue. Similar groupings are made along the other diagonals (wrapping around such that seven individual samples are included in each test as also shown for the blue test). This diagonal test resolves the false positives (labeled FP in Figure 1.4) which are now labeled as true negatives (TN), because the diagonal test does not include any positive samples and therefore tests negative. However, we can again incur false positives. The disjunctness of the design with the diagonal tests is two. When we have three positive samples, as shown in the right illustration, we can again incur false positives. Here, the third positive sample (again inferred as a true positive and labeled TP) is contained in the diagonal test. The diagonal test will then test positive. The sample labeled FP will then also be in a positive row, a positive column, and a positive diagonal test and will be a false positive.

obtained via

$$Y_i = \bigvee_{j \in [1,n]} M_{i,j} \wedge X_j,$$

where $\wedge$ and $\vee$ are the boolean AND and OR operators. This means that Y_i will test positive if any sample j is positive and contained in test i. The status of each sample is then obtained via a decoder, given only M and Y. It is possible to specify a range of different group testing design matrices. However, for the development of our method, while it can be applied to any design, we chose the d-disjunct shifted transversal design[120] (STD). A design matrix M is considered d-disjunct if the union of any d columns of M does not contain any other column. This then guarantees no false positives or negatives (assuming perfect tests) when the number of positive samples is not more than d. Intuitively, this can be seen by considering that if d or fewer samples are positive, then the union of any $d-1$ tests does not contain any other column and therefore if the $d-1$ tests would create an ambiguity about the status of a sample (as seen in Figure 1.4) then the final test will resolve that ambiguity, as only one of the potential positives will be contained in that test. In the array testing example in Figure 1.4 we can see that the 1-disjunct array testing design can return two false positives when two positive samples are present. However, we can increase the disjunctness of the test design by also grouping samples across the diagonals (Figure 1.5). The group test including the two false positives will then turn out negative, resolving the ambiguity and no false positives will be returned. However, by including a third positive sample, we again could incur false positives as laid out in Figure 1.5.

Notably, group testing as described above only considers binary test outcomes. A classical approach to decoding, i.e. inferring positive samples, in this binary test outcome regime is combinatorial orthogonal matching pursuit (COMP)[121]. When applying COMP to the STD design, a sample is declared definite negative if at least one of the tests the sample is in tests negative. The remaining samples are then inferred as positive. As such, in a d-disjunct design with perfect testing COMP will exactly identify the true positive samples. However, when more than d positive samples are included in the group testing design, COMP is likely to return false positives. Two recent papers[122,123], as ourselves, have realized the potential benefit in using non-binary test outcomes that are proportional to the number of positive samples in a test. This should in theory enable the detection of more positive samples while retaining a low error rate. In particular, the false positive rate could be reduced compared to COMP. Before I describe the approaches, PCR testing needs to be defined, as PCR test results are exactly the non-binary test results returned from infectious disease testing.

PCR testing

PCR testing, in our case quantitative reverse-transcriptase PCR, enables the identification of even tiny amounts of target DNA in a sample. As such it represents the gold standard of clinical infectious disease testing. For PCR testing it is important to know the DNA sequence of interest. Based on this sequence, primers can be designed that are complementary to the 3' and 5' ends of the target sequence. The primers attach to the matching, previously denaturated, single-stranded DNA in the sample. Using DNA polymerase the target DNA fragments can then be duplicated. By repeating this process, the amount of DNA fragments of interest is duplicated at each step (called cycle), greatly amplifying the amount of available target sequence. Reverse transcriptase refers to an enzyme that enables the conversion of RNA into DNA, enabling the application of the above process to RNA sequences, such as for RNA viruses. The last step in PCR is the measurement of whether the amplified target DNA sequence is actually present in the sample. In quantitative PCR, this is achieved through measurement of fluorescent light emitted either from intercalating fluorescent dye or 5' nuclease probes. Quantitative PCR enables the quantification of the amount of target sequence after each amplification cycle. A sample is declared as positive for the target sequence if the measured output is above a given threshold (Figure 1.6) within a maximum number of cycles. The specific number of cycles needed to cross the threshold is the cycle threshold (Ct) value. The Ct value is not necessarily an integer, as a function (such as the one shown in Figure 1.6) can be fit to the fluorescence values across the cycles and we can determine when this function crosses the threshold (Michael Crone; personal communication). For our work it is important to note that the higher the amount of original viral RNA copies per μl in the sample,

called the viral load (VL), the lower the Ct value. Furthermore, the Ct value is logarithmically related to original amount of target sequence in the sample, due to the doubling of the number of sequences in each cycle. Samples with a Ct value below 37 are generally considered positive. However, for group testing it can be reasonable to increase the maximum Ct value to 40 to account for sample dilution, due to the pooling of multiple samples into the same amount of fluid used for testing. A strong relationship between Ct values and dilution has been demonstrated previously [124]. Multiplexed quantitative PCR works in much the same way, but with multiple different fluorescent dyes for different target sequences, such as those of different infectious diseases, used at the same time. More details on PCR testing can be found in the book by Clark and Pazdernik [125].

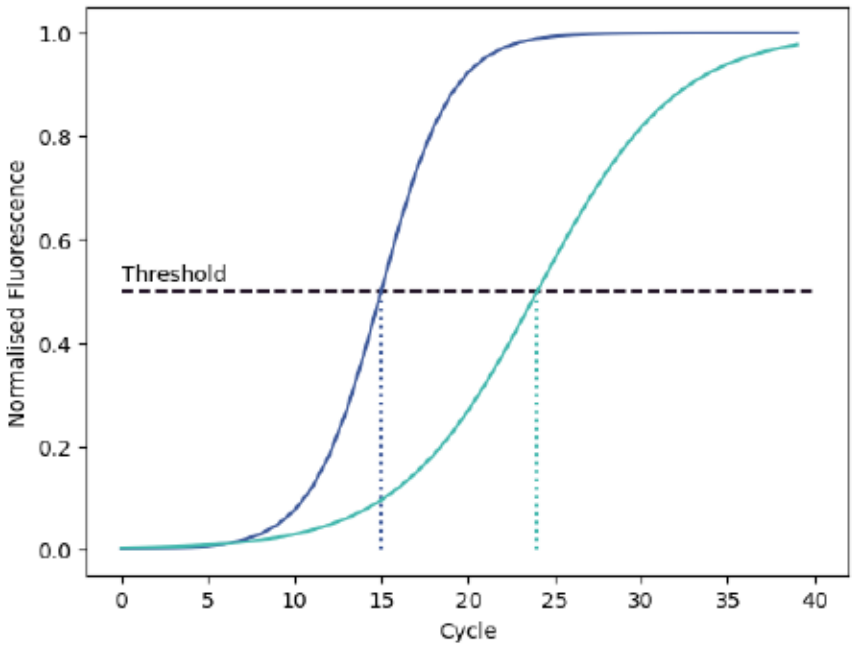


Figure 1.6: Example of PCR testing outcomes. Each curve represents a test. The Ct value of each sample is indicated by the dashed line of the same color, marking the point at which the fluorescence crosses the threshold. Units are arbitrary.

Ct value characteristics

In this work, Ct values are utilized to expand group testing beyond its binary test outcome regime. Hence, we must discuss some characteristics of Ct values. As any physical measurement, Ct values have some variance, even when two samples with the same VL are tested. In group testing, we split up samples into equal parts, combine different samples together, and test the groups. Firstly, the splitting process itself may lead to different VLs in different parts of the same sample. Secondly, even if the split is perfect, there is still stochasticity in the amplification process. Vogels et al. [124] tested multiple PCR tests. Their results indicate a maximum variability of 4.01 cycles in Ct values. My collaborators Michael Crone and Paul S. Freemont conducted experiments with artificial viral particles that indicate a maximum variability of 1.1

cycles[1]. Notably, my collaborators' testing protocol is likely more sensitive, as it generates lower Ct values for the same viral concentration and in general is able to detect lower viral concentration. Based on this, I will generally assume a Ct value sampled from a normal distribution with the expected Ct value as mean and 0.5 as standard deviation. This way 95% of Ct values will be within a interval of 2 around the expected Ct value. In practice, for each testing kit used, the variability should be determined prior to application of methods affected by Ct value variability.

Group Testing with PCR test outcomes

I will briefly describe the two methods previously developed for quantitative group testing, i.e. the usage of quantitative test outcomes. Firstly, Tapestry[126] applies a compressed sensing approach for decoding of group testing results. Specifically, group testing can be viewed as a special case of compressed sensing, which is an efficient method for finding sparse solutions to underdetermined linear systems of equations. Notably, Ghosh et al.[122] applied their novel approach to a set of real datasets. Here, it is shown that while generally performing favourably, the method still produces a notable number of false positive identifications (as well as some false negatives) even for a low number of positive samples. Tapestry also promises the reconstruction of viral loads. However, while Ct values do relate to disease outcome[127], viral loads are not straightforward to estimate even from individual tests. These estimates require careful calibration and normalization of the PCR test under consideration[128] and estimation of viral loads from Ct values remains controversial[129]. Secondly, Wang et al.[123] developed tropical group testing. In general, tropical group testing is an approach unsuitable for testing of infectious diseases. This is, as it requires the modification of tests during the testing process. However, Wang et al.[123] propose a decoder for tropical group testing that is also applicable to standard group testing as a special case. Notably, this decoder has not been validated in vitro.

Application of group testing

The application of group testing in real world settings has great potential. However, group testing is, with few exceptions[130,131,132,133,134,135,136], seldom applied in business as usual diagnostic and surveillance testing. In all of the cited cases simple adaptive testing is used, which can be considered a straightforward, but not necessarily optimal, group testing design. In general, there is little regulatory guidance on the application of group testing. Pandemic situations are an exception here to some degree[137,138], however, systematic regulatory guidance is essential. One major hurdle to more widespread application of group testing, and novel group testing designs and methods specifically, is the absence of clear validation workflows. Such a

validation workflow will enable the translation from design to practical application via in vitro validation and has been addressed in more detail together with my collaborators [1]. Furthermore, it is important to address the ethical issues surrounding group testing. One of the main potential issues in applying non-adaptive group testing is the return of false positive identifications. One can question whether it is appropriate to inform people of their status as testing positive for a disease while they are actually negative. While this routinely happens across medicine, as any test has a false positive rate, increasing the rate of false positives is not desirable. On the other hand, increasing testing capacities may outweigh the negatives, as we may be able to protect many more people from an infection for the cost of falsely identifying some. To weigh the benefits and costs of group testing lies in the hands of policy makers, however, in general, there is far too little discussion of this topic. Notably, STD and array testing and similar non-adaptive approaches still do offer the option of a second testing stage for the confirmation of the results, turning them into adaptive designs. Furthermore, d-disjunct STD testing has the guarantee that there are no false identifications if fewer than d positive samples are observed (assuming perfect tests). As such, retesting could be limited to cases when more than d potential positives are inferred. When the disjunctness is adapted appropriately to the expected prevalence, retesting should only rarely be required. The possibility of retesting should therefore be taken into consideration in designing effective policies. Importantly, even if individual identification of positive samples would be considered not ethically sound, group testing could still be employed for general surveillance when less emphasis is placed not inferring false positives. Representative sampling underlies genomic surveillance [139] and low income countries show a significant gap in testing to high income countries [140]. As such, group testing as a resource-saving method can be applied in this regime to great effect. Again, policy makers need to engage with group testing in this approach in detail to determine its potential.

Group Testing for Prevalence Estimation

The research I conducted in the field of group testing led to additional theoretical analyses which prompted the derivation of an analytical method for estimating prevalence from array testing. As for case identification, using group testing for prevalence estimation has the benefit of using fewer testing resources than individual testing. Here, I give a non-exhaustive overview of prevalence estimation using group testing, and array testing in particular, in the literature. In practice, it is likely not possible to derive a general solution to estimating the posterior distribution of the prevalence given group testing results and we need to either perform analyses for a particular design or employ more general approximation methods. There is a rich literature on estimating prevalence from Dorfman testing, i.e. each sample is contained in one group with groups of fixed size, and

retesting when a group tests positive[103]. Maximum likelihood estimates for the prevalence given Dorfman testing have been derived[141]. Dorfman testing has also been evaluated for different group sizes and test characteristics, namely sensitivity and specificity, which are not perfect, and may be lower in pooled testing compared to individual testing[142]. Additionally, Nguyen et al.[143] provide a method for estimating optimal group sizes in different scenarios. However, in clinical practice this may not be of much benefit, as quick adaptation of testing designs may constitute a potentially unjustifiable logistical overhead. On the other hand, an important consideration to make in the application of Dorfman testing to prevalence estimation is that of correlated samples. Most work assumes that sample positivity is identical and independently distributed (i.i.d.). A violation of this has been shown to potentially cause problems in group testing[144], prompting Hung and Swallow[144] to suggest the randomization of samples prior to testing.

Some analyses have been performed for array testing designs as illustrated in Figure 1.4. However, exact estimators have not been derived for array testing and only simulation based approximation methods are available. Martins et al.[145] used a previously developed simulation based approach[146,147] to compare Dorfman testing with only one stage and array testing. They show that for the same number of individuals, array testing shows a higher accuracy. Notably, for the same group size array testing requires twice as many tests as a single round of Dorfman testing by definition. This is due to the fact that in Dorfman testing, each sample is in one group only, whereas in array testing each sample is in a row and a column test. However, at sufficiently low prevalence rates it will also serve better to identify positive samples, without the need for retesting. Extending beyond 2-dimensional array testing, 3-dimensional array testing has been investigated and shown not to provide a benefit in prevalence estimation[148].

The estimation of multiple disease prevalence rates is also of interest. In general, any group testing method can be applied to multiplexed testing, but more specialized considerations can be made. Warasi et al.[149] provide an analysis optimizing group sizes for co-circulating diseases. This is expanded on by Warasi et al.[150] by presenting a Bayesian approach for inferring the prevalence rate in a multiplexing scenario using Markov chain Monte Carlo sampling. This allows for the estimation of multiple probabilities of infection, one for each possible combination of infection statuses, e.g. the probability of a sample being positive only for disease A, for disease A and disease B, and so on.

Thesis Aims

Firstly, I aim to advance the field of strain identification in metagenomics. The goals are twofold. In Chapter II, I present a benchmarking study of several tools identifying reference strains in metagenomic data, with the goal to identify the tools best suited for different research aims, in particular, investigations of different species, and to more broadly provide a perspective on the tools' performance. In Chapter III and Chapter IV, I present a novel tool for strain identification in metagenomes. Beyond simply providing competitive performance compared to the other available tools, interpretability and modularity was a central aim. One aspect that I aimed for to increase interpretability is to have the developed model explicitly work on the basis of identified coding sequences and to be split into core and accessory genome (the accessory genome consisting of all genes not found in all members of a species). The aim here was to build a model that can be adapted according to the species analyzed and that can be informed by particular research goals. The second aim in developing the new tool was to enable an estimation of acquired coding sequences or genes in the accessory genome of an identified bacterial strain that are not present in the reference strain's whole genome sequence. While StrainGE[89] does infer large deletions, large acquisitions of genetic material in bacterial strains are not, to the best of my knowledge, estimated by any available tool.

Secondly, issues in the field of non-adaptive group testing for surveillance testing were tackled. In Chapter V, I investigate the potential of deriving an explicit likelihood and posterior estimate the prevalence obtained from array testing results and compared the performance of array testing using the derived expression to individual testing. Beyond this, there is still potential in utilizing quantitative PCR test results for group testing. In Chapter VI, I present an initial investigation into a novel approach to incorporating quantitative PCR test results in the inference of positive samples. Part of the aim of developing this model is to be able to, in the future, decode multiplexed group testing data and to incorporate out prior beliefs on prevalence and expected Ct values. While the current model only aims at singleplexed testing, the current work underpins these future developments.

Chapter II

Benchmark of Strain-Level Resolution Taxonomic Classification Tools for Metagenomics

'The whole process resembles biological evolution. A problem is like an ecological niche, and a theory is like a gene or a species which is being tested for viability in that niche.'

– David Deutsch

Metagenomics has the potential to provide novel insights into the structure of microbiomes in many different environments. Additionally, there are a lot of areas of study that would benefit from the ability to identify strains within metagenomic data, such as the tracking of strains during outbreaks. Various tools are available that tackle the issue of inferring the strains in metagenomic samples, each with different aims. Here, I focus on those tools that aim at the recovery of one or more reference strains from metagenomic data. For any practical application, it is of vital importance to be aware of the different tools' performance characteristics, best use cases, and potential biases or shortcomings. However, while a recent study addresses SNP calling methods[79], there is still a need for a thorough benchmarking of reference strain identification tools. While the study presenting the tool StrainScan[91] has compared a number of tools on an array of datasets there are still several drawbacks with this evaluation. First and foremost, the study introduces the tool StrainScan which naturally takes the focus of the investigation onto this tool in comparison to the other available ones. Furthermore, the set up and details of running the different tools is not addressed sufficiently, owing again to the focus of the paper not being a benchmarking study. Moreover, several measures have not been addressed sufficiently. While F1-scores ($\frac{2\times\text{True Positives}}{2\times\text{True Positives}+\text{False Positives}+\text{False Negatives}}$, the harmonic mean of precision and recall) are reported mostly at strain level, more coarse grained clustered analysis will be beneficial. Strain

level resolution will often not be achievable, but identifying clades or clusters of closely related strains may be and these may still be highly informative. Furthermore, the focus is largely on F1-scores, while the Jenson-Shannon Divergence (JSD, symmetric distance between two probability distribution) is reported for the accuracy of the estimation of relative abundances. Relative abundances sum to unity and are at least 0. Hence, we can treat the relative abundances across all strains as a probability distribution and use the JSD as divergence measure between the actual relative abundances in a sample and the estimated ones. However, no combined measure, or investigation of the actual inferred number of strains, is conducted. Moreover, different relative abundance profiles have not been investigated. In practice, metagenomic data may contain multiple strains at similar frequencies. Another likely scenario is that one strain is dominant, making up most of the species' abundance in a sample, with other strains only making up a small proportion. Finally, the underlying reasons of varying performances across species were not addressed sufficiently.

Here, I present a benchmarking of the six tools summarized in Chapter I which serve either the direct identification of reference strains or clusters of closely related reference strains in a database. Reiterating, the tools included in this analysis are StrainEst[84], StrainScan[91], StrainGE[89] (a toolbox including strainGST for strain identification), KrakenUniq[86], Sigma[90], and MIST[88] (for ease of presentation the tools are summarized again in Table 2.1). These tools were evaluated in detail on simulated data of four bacterial species, *A. baumannii*, *E. coli*, *M. tuberculosis*, and *L. monocytogenes*. These four species were chosen due to data availability, clinical relevance, and their different population structures. On the one hand, *M. tuberculosis* is a highly clonal species, with fairly little variation between strains[151], while *E. coli* shows much higher variation[152,153]. *A. baumannii* is both highly relevant in the context of antibiotic resistance[154,155,156] and shows a high rate of recombination[157,154]. Lastly, *L. monocytogenes* is of special interest in food safety[44] and metagenomics may provide a promising avenue for food safety surveillance by enabling identification, tracking, and source attribution of *L. monocytogenes* strains. I created datasets both from strains that were provided to the programmes via their reference databases and strains that were excluded from the database, testing different relative abundance profiles, coverages, and varying the relatedness between strains. Furthermore, I evaluated the different tools' performance on several complex mock communities. This analysis delivers a much needed benchmarking of the different tools available for four highly relevant pathogenic bacterial species, showcasing the tools' strengths and weaknesses and thereby providing researchers with the information necessary to decide how to include strain-level analysis in their research.

Tool	Summary	Database	Notes
StrainEst	Reference strain cluster identification via penalized regression of nucleotide frequencies	Custom database requires custom clustering of genomes. Pre-processed database available for some species	
StrainScan	*k*-mer based, hierarchical identification of clusters down to exact reference sequences	Custom database or pre-processed for some species	
StrainGE	*k*-mer based reference strain cluster identification with strainGST	Custom database requires parameter setting. Pre-processed for some species.	SNP calling and detection of large deletions via strainGR
KrakenUniq	Designed for simultaneous analysis of whole NCBI taxonomy via *k*-mer counting	Can use custom or pre-processed database. Pre-processed database can be extended with custom sequences.	Enables read assignment. Requires separate installation of jellyfish 1.1.12
Sigma	Alignment based reference sequence identification via probabilistic model	Custom set of reference sequences	Provides confidence intervals via bootstrapping and variant calling
MIST	Alignment based reference strain cluster identification using probabilistic approach	Custom database clustered at user-specified average nucleotide identities. Uses precomputed or custom pan-genome. Preprocessed database links not functional.	Enables read assignment.

Table 2.1: Overview of the evaluated strain identification tools.

Methods

Here, I lay out the methods used to perform the benchmarking. Firstly, for each species I describe the method of obtaining the strains used. Secondly, I describe how a separate dataset for each species is simulated from the strain databases. Then, I describe the steps taken to create the databases for each tool, again with a separate database for each species. Finally, I describe the outcome measures used to assess performance of the different tools, as well as several further analyses that were conducted.

Strain Databases

For the present study the species *E. coli*, *M. tuberculosis*, *A. baumannii*, and *L. monocytogenes* were considered. For each a separate reference database was created, which is described for each species separately below together with a brief summary of some key species characteristics. These reference databases were based on assemblies. In general, assemblies correspond to a single or several long continuous sequences (contigs) created through the merging of shorter reads. In the NCBI database, different assembly levels exist. Firstly, the contig level assembly indicates that several gapless sequences exist, but that these are not necessarily connected. At the scaffold level, contigs are connected across gaps. For the chromosome level assembly, there is at least one chromosome, however, unlocalized sequences may still be present. Finally, there is the complete genome assembly level, indicating complete, gapless genomes with few ambiguous bases. Throughout, the term strain will be operationally defined to correspond to a single unique assembly. It is noteworthy that for the analyzed species metadata on the NCBI database is only partially available and often not unified.

Nonetheless, to provide some information on the origin of the isolates, where the information was available, histograms for the countries of collection of the assemblies of *A. baumannii*, *M. tuberculosis*, and *E. coli* are shown in Figure 2.1.

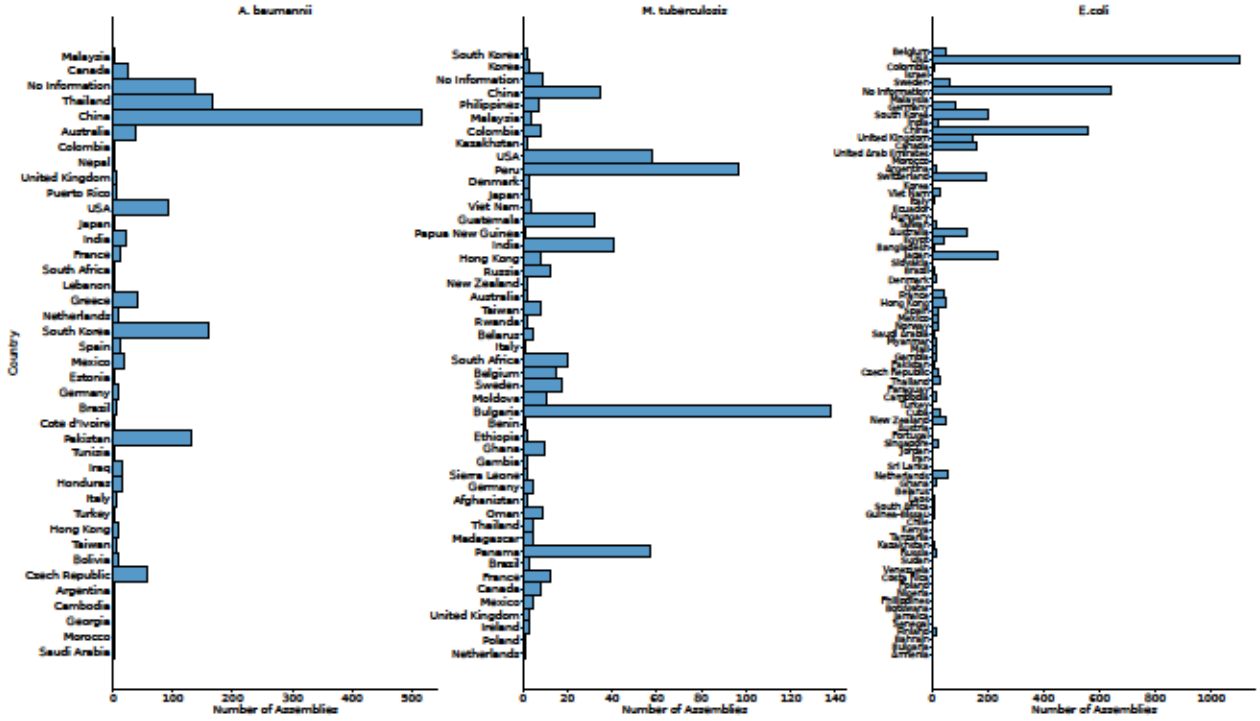


Figure 2.1: Country of collection for the assemblies of the different species. 'No Information' count contains all assemblies with the following labels: 'not determined', 'Not determined', 'unknown', 'Unknown', 'not collected', 'Not collected', 'missing', 'Missing', 'not applicable', 'Not applicable', 'Not Applicable', 'not available', 'not provided', 'nan'.

E. coli

Escherichia coli is one of the most widely studied bacterial species. Its study was an integral part both in early whole genome sequencing studies[158] and in the establishment of the pangenome concept[159]. *E. coli* is classified as gram-negative[153]. Importantly, *E. coli* divides into commensal and pathogenic variants[152]. Commensal *E. coli* variants can be found in the intestines of many mammals[152]. Pathogenic variants divide again into intestinal pathogenic *E. coli* (IPEC) and extra-intestinal pathogenic *E. coli* (ExPEC)[153]. Distinguishing these pathogenic from commensal variants is of great interest and especially distinguishing ExPEC from commensal variants presents major challenges[153]. In the past, *E. coli* was estimated to cause upwards of 2 million fatalities per year[152], while a more recent estimate places the number of deaths lower, though still above 500 thousand[2]. Furthermore, *E. coli* is frequently observed in environments such as soil and water[160]. The observed variety in types of *E. coli* is also represented in the pangenome. The pangenome

is composed of all genes found in a species, spanning core and accessory genomes. As genome sequencing progressed, studies included an increasing number of *E. coli* strains and the number of genes shared between all strains declined with an increase in the number of strains analyzed [161,162,163,164]. As such, and given the fact that some strains of *E. coli* only share 20% of their genome, the classification of *E. coli* as a singular species has been debated [153]. *E. coli* was chosen as the first species for this benchmarking study both due to its high intra-species variability and its ubiquity in the evaluation of strain identification tools performance. Indeed, StrainEst [84], StrainGE [89], MIST [88], Sigma [90], and StrainScan [91] all incorporated *E. coli* as a species to test their respective tools on. As such, successful analysis of *E. coli* can be considered an essential goal of these tools and should be evaluated and compared.

To establish a database of *E. coli* strains, assemblies were downloaded from the NCBI database, as would likely be done in everyday research practice. For *E. coli*, the NCBI database contains 4153 complete genome assemblies, which were downloaded to create the database. Chromosome, scaffold and contig level assemblies were not included, as in total, there are far more assemblies in the database than could sensibly be included in the database (252118 assemblies across all assembly levels). Hence, the focus lay on the most complete available assemblies. Next, a number of assembly statistics were calculated to exclude outliers and assemblies of comparatively low quality. The measures considered were total genome length, the count of undetermined bases, the number of gaps, the N50 (shortest contig that needs to be included to cover over 50% of the genome length), the number of contigs required to reach the N50, and the total number of contigs (Appendix Figure A5). Assemblies that deviated by more than three standard deviations from the mean of any measure were excluded (Table 2.2). This cutoff is somewhat arbitrary, however, it will ensure that nearly all assemblies (99.7% assuming a normal distribution) will lie within this range, hence only excluding extreme outliers. Details on mean, standard deviation, and number of excluded assemblies for each statistic can be found in Table 2.2. In total the analysis of assembly statistics led to the exclusion of 74 assemblies for *E. coli* (1.8%). Notably, all of the assemblies have an N50 count (number of contigs that need to be included to achieve the N50) of one, as well as very few ambiguous bases or gaps. This high quality in genomes is likely due to the fact that only Complete Genome level assemblies were included to begin with. *E. coli* also has the longest genome length among the species analyzed here with an average genome length of 5.1 million basepairs (STD $= 350000bp$). Regarding the country of collection (Figure 2.1), most *E. coli* assemblies were collected in the USA (1105 assemblies) and China (558 assemblies), with no information available for 645 assemblies. To obtain information about the sub-species level structure, MLST assignment of the assemblies was performed using the mlst tool [165], which uses Blastn [166] to perform alignment of the PubMLST database [74] of MLST alleles and the assemblies to obtain strain types. Strain types are only reported for perfect matches. Overall,

the dataset showed a wide variety of identified strain types (289 strain types in total), with strain types 2, 43, 628, and 262 making up the majority of assemblies for which a strain type could be reliably inferred (1175 assemblies could not be assigned a strain type unambiguously; Figure 2.2).

The final reference database was created by excluding 10% of assemblies at random. These 10% will serve to simulate data from assemblies not present in the reference database provided to the tools. The resulting reference database then consisted of 3671 assemblies. Both the *reference database* (which does not contain the 10% excluded strains, and was passed to the strain identification tools as database) and the *total database* (which includes the 10% excluded strains) were independently clustered for downstream data creation and analysis. For each dataset the procedure for clustering was as follows. First, all-to-all pairwise Mash[61] distances were computed between all assemblies. A mash distance is a measure of sequence similarity, estimating the mutation distance between two genomes[61]. The benefit of Mash distances is their fast compute time while preserving high accuracy, which becomes relevant when conducting millions of pairwise comparisons. Importantly, Mash distances correlate highly with average nucleotide identities (with ANI $\approx 1 -$ Mash distance). The resulting distance matrix was used as a basis for complete linkage hierarchical clustering. The goal was to create clusters of different resolutions based on genome similarity, achieved through identifying clusters at three different levels, each using a different number of clusters. The number of desired clusters was set to 10, 100, and 500 at the three levels, respectively. To achieve these clustering levels, within the complete linkage hierarchical clustering, a Mash distance threshold was chosen as the highest distance that results in not more than the specified (10, 100 , or 500, respectively) number of clusters, resulting in the final three clustering levels (see Appendix Figure A1 for a histogram of the cluster sizes at each level). For *E. coli*, a higher clustering level could also have been considered given the large number of assemblies. However, the other species have fewer assemblies available and for comparability the highest resolution chosen was 500 clusters. Throughout, I will refer to the different resolutions in clustering as the 10, 100, and 500 cluster levels, while discussions of results relating to individual assemblies are referred to as being on the assembly level.

M. tuberculosis

Mycobacterium tuberculosis is an acid-fast pathogenic bacterial species[167]. Tuberculosis (TB) caused 1.3 million of deaths in 2022[168], with *M. tuberculosis* being a major cause of TB. *M. tuberculosis* is transmitted host-to-host, and notably requires causing disease to be transmittable[169]. Adding to its high clinical burden is the emergence of antibiotic resistance within the species. Antibiotic resistance can arise quickly due to large

populations despite a low mutational rate and little horizontal gene transfer compared to other species[167]. The importance of resistance is highlighted by rifampicin-resistant *M. tuberculosis*' listing as critical on the WHO Bacterial Priority Pathogens List[156]. The population structure of *M. tuberculosis* shows comparatively little diversity, with individual strains only differing in around 2000 SNPs from each other[151]. Additionally, the different clades of *M. tuberculosis* are clearly divided geographically and are adapted to hosts from different origins[170]. Transcriptional regulation and epistatic interactions seem to play an important role in the diversity of virulence of *M. tuberculosis*[167]. The low genetic variability between members of *M. tuberculosis* is in contrast to a more variable species such as *E. coli*. Hence, including *M. tuberculosis* as species in the benchmark will provide insights into the tools performance in such a less variable species.

For *M. tuberculosis*, assemblies were obtained from NCBI, similar as was done for *E. coli*. However, due to fewer assemblies being available, both Complete Genome and Chromosome level assemblies were downloaded. This resulted in a total of 703 assemblies. Assemblies were assessed on the same measures as for *E. coli* and assemblies that deviated from the mean by more than three standard deviations were again excluded (Table 2.2, Appendix Figure A6). This led to the exclusion of 32 assemblies (4.6%). Finally, 10% of assemblies were again excluded from the reference database such that the final reference database provided to the tools contained 607 assemblies. Both the final reference database and the total database were again clustered using complete linkage hierarchical clustering at the 10, 100, and 500 cluster levels (Appendix Figure A2). MLST analysis of *M. tuberculosis* (performed as for *E. coli*) shows a markedly different distribution of strain types from *E. coli* (Figure 2.2), with strain type 506 making up the majority of identified strain types (26 assemblies were unidentified). Furthermore, only 30 strain types were identified in total. This points towards a much more conserved population structure in *M. tuberculosis*, which may translate to a more challenging identification of strains in metagenomic data. As such, this dataset provides a good contrast for benchmarking tools compared to *E. coli*. Assembly statistics (Table 2.2) show a similar quality to *E. coli*, again with a low N50 count of 1 for all assemblies. However, more ambiguous bases are observed (Mean = 21000, STD = 25). The average genome length is shorter at 4.4 million basepairs. The difference in ambiguous bases can potentially be explained by the inclusion of Chromosome level assemblies in addition to Complete Genome level assemblies. For *M. tuberculosis* most samples were collected in Bulgaria (138 assemblies) (Figure 2.1). Peru (97 assemblies), the USA (58 assemblies), and Panama (57 assemblies) were also the origin of many assemblies. No information was available for 9 assemblies.

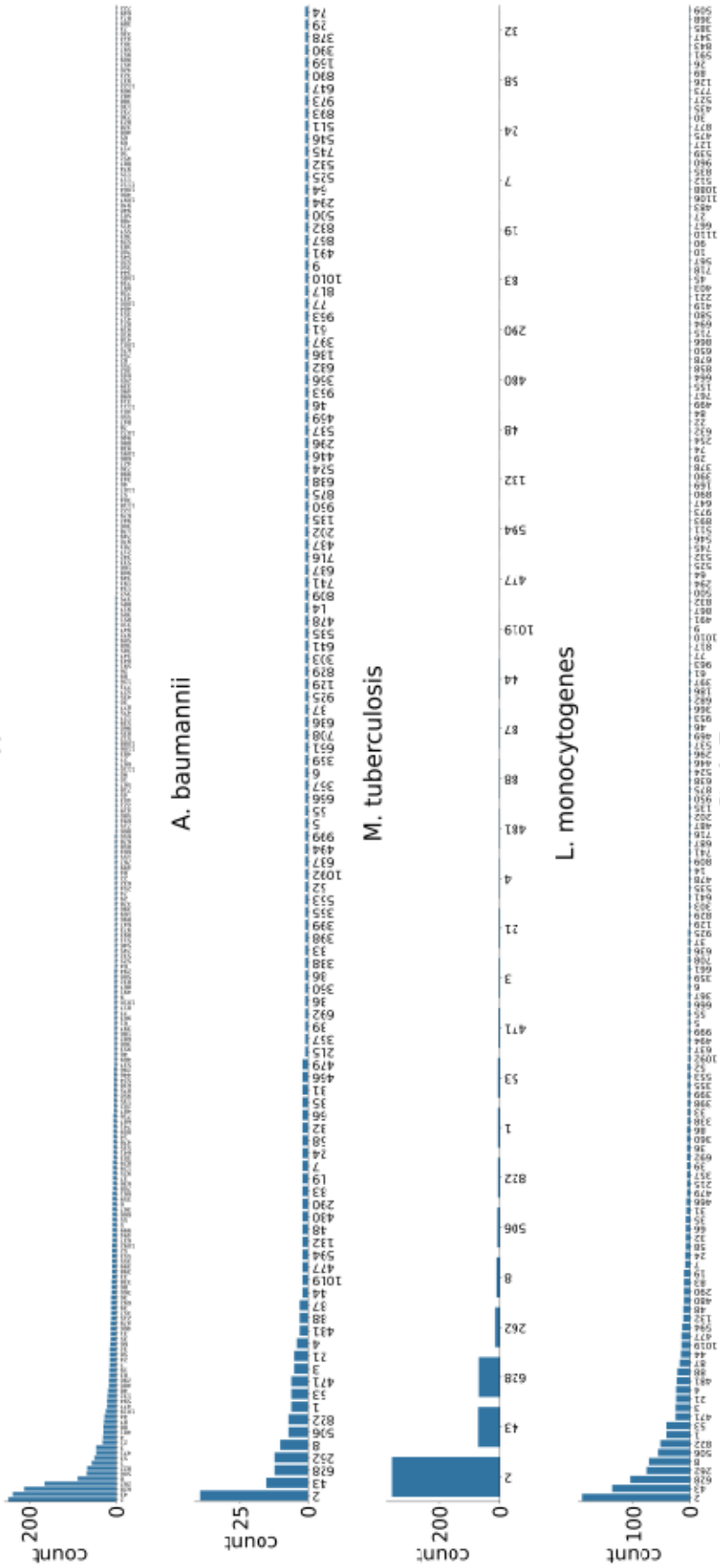


Figure 2.2: Counts of the identified MLST types for each species sorted by number of appearance. Tick labels on the x-axis indicate the strain type. Count is the number of assemblies with that strain type. Assemblies without an identified strain type are not shown.

A. baumannii

Acinetobacter baumannii is a gram-negative bacterium and nosocomial pathogen often causing respiratory but also other infections[171]. *A. baumannii* isolates frequently display antibiotic resistance, with mobile genetic elements playing a large role in its dissemination[154,155]. Nonetheless, homologous recombination is still relevant, and resistance genes are found on the chromosome of several lineages[172]. Analyses by Hernández-González et al.[154] further indicate that the resistome of *A. baumannii* is not only mobile, but also follows the phylogenetic structure of the species. Regarding the phylogenetic structure of *A. baumannii*, the species is split into three clones of major interest, international clones I, II, and III[172,173], which are themselves made up of several lineages. These clones are often observed in Asia, Europe, and the United States, and are thought to make up the majority of infections[172,173]. However, Graña-Miraglia et al.[173] point towards evidence of a higher relevance of international clones IV, V, and VII in Latin America. Their study further indicates the potential coexistence of different lineages in the same hospital environment. Lastly, research also points towards the existence, and potential relevance from a one health perspective, of *A. baumannii* in non-human hosts and non-hospital environments, which remains largely understudied[174]. *A. baumannii* was chosen for its relevance in antibiotic resistance and its prevalence in different ecosystems. Additionally, a highly curated database provided by our collaborator Santiago Castillo Ramírez provides a solid basis for the applications of the different tools. Furthermore, *A. baumannii* was used as the species for model development of my own tool presented in Chapter III and Chapter IV, as my collaborator Santiago Castillo Ramírez is an expert on this species, and his advice was essential during model development.

For *A. baumannii*, a curated set of 1473 high-quality assemblies compiled by Hernández-González et al.[154] was used, which constitutes yet another approach to how reference databases might be constructed in practice. After assessment of the assembly statistics as done for *E. coli* and *M. tuberculosis*, 100 (6.8%) assemblies were excluded (Table 2.2, Figure A4). Again, 10% of the database were excluded for the reference database construction, resulting in a total reference database of 1240 assemblies, and both the reference database and the total database were clustered via complete linkage hierarchical clustering in the same way as for the other species. Figure 2.3 shows the number of strains in each cluster. Here, it is noteworthy that many of the assemblies cluster into very few clusters, indicating high similarity between strains at lower resolutions. Most of the assemblies were collected in China (517 assemblies). Thailand (167 assemblies), South Korea (161 assemblies), and Pakistan (131 assemblies) were further countries with a high number of collections. Outside of the Asian continent, the USA was the country with the most collected assemblies (91 assemblies). For 138 assemblies no information was available. MLST analysis of *A. baumannii* (using the Oxford MLST scheme[175], Figure 2.2) revealed one dominant identified strain type (229). However, 975 assemblies could not

be assigned a strain type. This may or may not indicate many very closely related assemblies in the dataset. Again, when investigating the clustering performed above, we can see that at the 10 and 100 cluster level, indeed mainly one cluster was identified (Figure 2.3). However, at the 500 cluster level multiple clusters can be observed at relatively high frequency. Notably, the clustering for the other species shows a more even distribution of assemblies across clusters (Appendix Figures A1-A3). The *A. baumannii* assemblies show a comparable number of ambiguous bases to *M. tuberculosis* (Table 2.2). However, the N50 is markedly lower (700 thousand compared to 4.9 million for *E. coli* and 4.4 million for *M. tuberculosis*) and the N50 count is higher (Mean = 11, STD = 13 versus an N50 count of 1 for all assemblies for both *E. coli* and *M. tuberculosis*). This could potentially indicate lower assembly quality. Similarly, the number of contigs is very high (Mean = 130, STD = 216), which points towards the presence of more fragmented assemblies. Notably, the dataset also included Scaffold level assemblies[154], which may explain this deviation in assembly quality. The average assembly length was 3.9 million bp (STD = 130000bp).

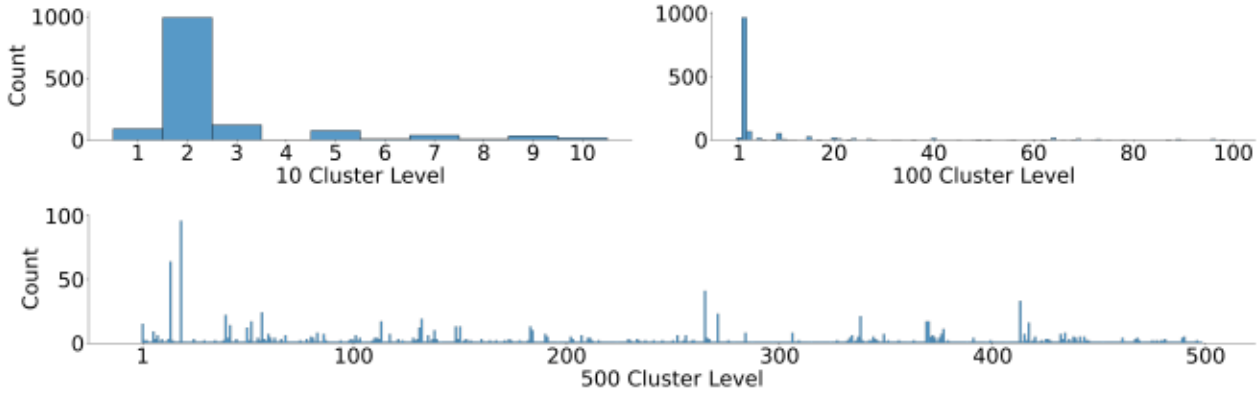


Figure 2.3: Count of assemblies of *A. baumannii* assigned to each cluster at the three clustering levels (assemblies clustered into 10, 100, and 500 clusters, labeled as 10 Cluster Level, 100 Cluster Level, and 500 Cluster Level, respectively). Cluster numbers are arbitrary and numbering increases left to right on the x-axis (some x-axis ticks were left out for visibility).

L. monocytogenes

Lysteria monocytogenes is a gram-positive bacterium, and a major cause of food-borne disease in humans[44]. Infections show a high fatality rate and are on the rise in Europe[44]. *L. monocytogenes* can be divided into four lineages, with lineage I and II causing most human infections[176]. The lineages can be further subdivided into serotypes, with certain serotypes being the major cause of human infection[177,178,179]. While most lineages share the same basic virulence factors, virulence varies across serotypes[44]. Furthermore, *L. monocytogenes* shows resistance to disinfectants used in food-processing facilities[180] and can grow at refrigerator temperatures[181]. *L. monocytogenes* was chosen as the final species, as the the genomes for *L. monocytogenes*

		Genome Length	N count	# of Gaps	N50	N50 Count	# of Contigs
E. coli	Mean (STD)	5.1e6 (3.5e5)	0.72 (38)	0.52 (26)	4.9e6 (3.2e5)	1.0 (0.0)	3.5 (2.2)
	# Excluded	24	2	3	26	0	45
M. tuberculosis	Mean (STD)	4.4e6 (3.9e4)	2.1e4 (5.6e4)	202 (1.7e3)	4.4e6 (3.9e4)	1.0 (0.0)	1.0 (0.11)
	# Excluded	28	25	2	2	0	2
A. baumannii	Mean (STD)	3.9e6 (1.3e5)	2.2e3 (1.9e4)	21 (44)	70e5 (1.3e6)	11 (13)	130 (216)
	# Excluded	15	4	38	0	42	32
L. monocytogenes	Mean (STD)	3.0e6 (0.74e5)	0.0024 (0.049)	0.0024 (0.049)	4.3e5 (3.3e5)	4.3 (3.9)	28 (23)
	# Excluded	2	4	4	0	34	34

Table 2.2: Statistics for assemblies of the different species. Genome Length is the length of the genomes in nucleotides. N count is the number of ambiguous bases in the assemblies. # of Gaps is the number of gaps (-) in the assemblies. N50 is the shortest contig that needs to be included to cover over 50% of the genome length and N50 count the number of contigs that need to be included to reach the N50. # of Contigs is the number of contigs the assemblies consist of. # Outliers shows the number of assemblies that are more than three standard deviations away from the mean for each measure. The same assembly may be an outlier on multiple measures. Hence, some assemblies may be excluded on multiple grounds and the row totals do not add to the number of excluded assemblies for each species.

were sequenced and assembled in-house by my collaborator Alexander Gmeiner at the Technical University of Denmark. This represents a difference to the other species, being sequenced in various different laboratories. As such, the set of strains represents a unique approach to database creation that can be included and tested in the benchmarking process.

1649 assemblies were obtained in this manner in total. Location data was not available for this dataset. Based on assembly statistics, 42 (2.5%) assemblies were excluded (Table 2.2, Appendix Figure A7). Again, 10% of assemblies were excluded from the reference database provided to the strain identification tools and both the final reference database consisting of 1446 assemblies, and the total database were clustered using complete linkage hierarchical clustering to obtain 10, 100, and 500 clusters at the three levels of clustering, respectively (Appendix Figure A3). For *L. monocytogenes* MLST analysis (Figure 2.2) using the Pasteur scheme[182] revealed 167 distinct strain types (with 110 assemblies not assigned a strain type). Multiple strain types show high frequencies, with the five most common ones (count in parentheses) being 121 (185), 9 (135), 1 (105), 8 (75), and 3 (71). Finally, the assembly statistics for *L. monocytogenes* show overall high quality, with very few ambiguous bases or gaps in the assemblies and a low N50 count of 4.3 (STD = 3.9, Table 2.2). The average genome length was 3 million basepairs with a standard deviation of 75000 basepairs.

Data Generation

To provide a thorough evaluation of the different tools, I created a large set of simulated datasets of varying complexities and characteristics. In creating simulated data, multiple avenues exist. One possibility is to spike real metagenomic data with strains from a reference database. This would create highly realistic simulated samples. However, the interpretation of results on strain inference on these may not be straightforward. For example, the real metagenomic data may already contain one or multiple strains of the investigated species

at unknown abundances, making it hard to infer whether found strains are present in a sample or constitute false positives. Hence, this avenue is left for future research. Instead, the focus of the current study lies on strain mixtures from single species, making interpretation of results straightforward. Throughout, the assumption is made that all parts of a genome used for data creation are equally likely to be sampled for read creation. In future work, it may be interesting to investigate the possibility that different regions of a genome have different chances of being sequenced. To still consider complex metagenomic data, mock metagenomic data containing multiple species were also analyzed (see below). For the simulated data, 210 samples were created for each species for the single species strain mixtures (840 samples in total, 210 for *E. coli*, *M. tuberculosis*,*A. baumannii*, and *L. monocytogenes*, respectively). The 210 samples for each species were created with the same sample characteristics described below. The 210 samples split as follows. 200 of the samples were composed of assemblies that were present in the species reference database. 10 further samples were simulated from the assemblies previously excluded from the reference databases, and which are therefore not in the databases of the tools tested. I will first describe the 200 samples created from assemblies contained in the reference databases, with an overview given in Table 2.3. Firstly, single strain samples were created at 1-, 5-, 10-, and 20-fold coverage. For each coverage, 5 different samples were created, resulting in 20 single strain samples in total. The constituent assemblies were selected at random. Next, samples containing multiple strains of a species were created, again with the assemblies being chosen at random. The samples were created using a combination of parameters. Samples contained either 2, 3, or 4 distinct assemblies. These assemblies were either previously clustered into the same cluster at the 100 cluster level, or into different clusters. Here, clusters were selected at random first and strains were selected at random from within these clusters. Additionally, samples were created at a total coverage (sum of coverages across all assemblies in the sample) of 5, 10, and 20. The final parameter relates to the relative abundance profile across the different assemblies. Here, two distinct profiles were tested, one in which all strains within a sample had the same relative abundance (e.g. each strain in a four strain sample makes up 25% of the total coverage) or in which there was one randomly selected strain present at 80% of the total coverage and the remaining strains had an equal share of the remaining 20% of the total coverage. In total, this leads to 36 parameter combinations of number of strains, coverage, clustering type, and relative abundance profile. For each of these combinations, 5 repeats were created, leading to a total of 180 multistrain samples per species. For the remaining 10 samples created from assemblies previously excluded from the reference databases, the following parameters were used. 5 samples containing one strain and 5 samples containing two strains were created for each species, all at 10-fold coverage with assemblies selected at random. Throughout, ART[37] was used for data creation using the following settings

Strains	Total Coverage	Clustering	Relative Abundance Profile	Repeats
1	1; 5; 10; 20	One cluster	1	5
2; 3; 4	5; 10; 20	Same cluster; Different clusters	All equal; One strain at 80% and all others equal	5

Table 2.3: Parameters for metagenomic data generation. For each species 5 repeats were created for every combination of parameters in a row. See Appendix Table A1 for more details on the individual samples.

```
./art_illumina -ss HS25 -sam -i -l 150 -p -m 600 -s 100
```

These settings produce Illumina HiSeq simulated reads of length 150bp with a mean fragment length of 600bp (standard deviation 100bp). Details on the number of reads in the final generated samples can be found in Appendix Table A1.

Lastly, metagenomic samples are not simply composed of a single bacterial species. Hence, it is important to test the performance on complex strain mocks. The NIST dataset was chosen [183], as the available samples contain *A. baumannii*, *E. coli*, as well as *L. monocytogenes* in varying combinations (Supplementary Table S1d in Valencia et al. [183]). The samples contained an average of 3,476,327 paired-end reads (range 3,050,176 to 3,722,052). The absence of M. tuberculosis from all samples also makes for a good opportunity to evaluate the potential for false positive identifications. For these samples, the constituent strains were separately sequenced, making whole genome assemblies available for analysis. Note that in each sample only one strain from each species was contained. Additionally, the species of interest to us here are not the only ones in the sample, but other species are also present. This makes the analysis of these samples closer to real metagenomic data and increases their usefulness for tool validation.

Tool installation, database creation, and strain extraction

Sigma I attempted to download Sigma via the link provided in the manual (`https://github.com/BioHPC/Sigma/blob/main/SIGMA_UserManual_1.1.0.pdf`). However, the referred website does not exist anymore. Attempting to build Sigma from source resulted in a Sigma internal error message reporting a non-zero exit status. Hence, Sigma was excluded from all further analyses.

KrakenUniq I attempted to install the KrakenUniq database with the following command

```
krakenuniq-download --db databaseDirectory
    --taxa "archaea,bacteria,viral,fungi,protozoa"
    --dust --exclude-environmental-taxa microbial-nt
```

Within the maximum available runtime of 72 hours (using 64 cores), database creation did not complete. Hence, all but the bacterial taxa were excluded. This exclusion may lead to more false positive identifications. However, at a strain level this impact should be minimal. In total four databases were created, one for each species, by also adding the assemblies for each species to the database. The final command for database construction was then

```
krakenuniq-build --threads 10 --db $databaseDirectory
    --taxids-for-genomes --taxids-for-sequences
    --jellyfish-bin $(type -P -a jellyfish)
```

On the system used for this research the `jellyfish` package had to be installed separately. Future users may find this information helpful.

KrakenUniq provides various metrics for each identified taxon in its output, requiring the definition of a criterion for positive strain identification. Firstly, I extracted all identified strains from the database (assembly level taxonomic assignment). For each identified strain the E-value as proposed by Guellil et al.[87] was calculated. This E-value is defined as

$$E = \left(\frac{K}{R}\right) \times C \tag{2.1}$$

where K is the k-mer count, R the read count, and C the coverage of each identified strain as returned by KrakenUniq. Only strains for which $E < 0.01$ were inferred as present. In a related work using KrakenUniq, Valencia et al.[183] used 0.001 as cut-off value, however, this cut-off was loosened for the present study given that strains are the target of investigation, which will have lower numbers of unique k-mers assigned to them than species.

StrainGE For StrainGE four databases were created from the different species' reference assemblies. When setting up the database for StrainGE, three parameters need to be set. The k-mer size, set to the default of 23, the percentage of shared k-mers between a genome and another one to be excluded (`-C` option), set to 0.99, and the minimum Jacquard-similarity between two genomes to be clustered together need to be specified (`-c` option). The aim was to create as many clusters as possible to achieve as much taxonomic resolution as possible, thus setting the clustering threshold as high as possible such that a database is created by the software (for some settings no database could be created). Based on this, the parameter was set to 0.989 for *M. tuberculosis*, 0.94 for *A. baumannii*, 0.75 for *E. coli*, and 0.97 for *L. monocytogenes*. In their publication, van Dijk et al.[89] only investigated thresholds up to 0.9, with 0.9 being optimal in their research. I attempted

to use higher thresholds where possible, but the 0.75 threshold for *E. coli* is lower than the recommended threshold, hence, lower performance for this species may be expected. A higher threshold was not possible to use, as StrainGE did not return a clustering for any higher threshold for unknown reasons. In total, the four final databases contained 286 clusters for *A. baumannii*, 358 for *E. coli*, 133 for *M. tuberculosis*, and 311 for *L. monocytogenes*.

StrainScan For StrainScan separate databases for each species were again created. Default settings were used. For *E. coli*, StrainScan did not complete running with 60 cores and 72h runtime. Hence, I down-sampled the *E. coli* assemblies using complete-linkage hierarchical clustering and choosing the maximum threshold that resulted in up to 1000 clusters of *E. coli*. 895 assemblies were included in the final database (down-sampling was also attempted for 2000 and 1500 strains, but again StrainScan database creation did not complete within 72 hours runtime). For the other three species, all assemblies in their respective reference databases were used to create the three databases.

MIST For the databases for running `MIST.py strain` module I again used all the assemblies for each species to create a separate database for each species. For the different clustering cutoffs that MIST requires to be set 0.98, 0.99, and 0.999 were chosen, in accordance with the original publication and the documentation. Prior to running `MIST.py strain`, `MIST.py species` needs to be run. This requires a pre-built pangenome to classify reads into species. The default databases provided with MIST were used, as would likely be done in research practice.

StrainEst To create the StrainEst databases I followed the steps as laid out by Albanese and Donati [84]. However, in the choice of representative genomes, they write that they chose a threshold for the complete-linkage hierarchical clustering which is 'the lowest value below which the core genome of the species fell below 10% of the length of the species representative genome.' How the core genome is determined here is not clear. Instead, complete-linkage hierarchical clustering was performed for each species and 500 representative assemblies were mapped against the species representative. complete-linkage hierarchical clustering was run using the clustering function of scipy (scipy.cluster.hierarchy.linkage) where the underlying distance matrix is the matrix of all pairwise Mash distances between assemblies. This alignment is the foundation of the SNP calling StrainEst performs. Further, a threshold providing 1000 representative genomes for classification was chosen. This represents a much more fine-grained clustering than in the original publication [84], but did not exceed the available computational resources.

Outcome measures

I compare the different tools on three main measures. Firstly, the cluster-based, as well as assembly level, F1-scores are computed. At the assembly level the F1-score measures correct identification of exactly the assemblies used for data creation. For the cluster-based F1-scores it is checked whether the assemblies used to simulate the data and the assemblies that were identified by the tools were assigned to the same cluster, where the clusters are the ones obtained by clustering the total database of each species including the 10% previously excluded assemblies. F1-scores were computed for the 10, 100, and 500 cluster level. Secondly, the JSD is employed as a measure of the divergence of the estimated relative abundance distribution. The JSD is defined as

$$JSD(A||B) = \frac{1}{2}\left(D_{KL}(A||\frac{1}{2}(A+B)) + D_{KL}(B||\frac{1}{2}(A+B))\right), \tag{2.2}$$

where $D_{KL}(A||B)$ is the Kullback–Leibler divergence (a non-symmetric entropy based measure of distribution similarity defined as $\sum_{x\in\mathcal{X}} A(x)\log\frac{A(x)}{B(x)}$, with $\mathcal{X}$ being the sample space of the distributions A and B) between distributions A and B. Applying the JSD is useful as we can view the estimated and true relative abundances as probability distributions, as the relative abundance of each assembly is greater or equal to 0 and the relative abundances across strains for a species sum to 1. JSDs were computed again at an assembly level and the 10, 100, and 500 cluster levels. For the cluster levels, the clusters of all inferred strains at a given level of resolution were determined and the relative abundances of all inferred strains belonging to the same cluster were summed to give the cluster relative abundance. The cluster level relative abundance distributions for the strains making up the simulated samples were obtained in the same manner. The cluster level JSDs were then calculated from the cluster relative abundances. Note that the range of the computed JSDs is between 0 and $\ln 2 \approx 0.69$.

Additionally, the Wasserstein distance[184,185] (WSD), a rarely used measure in this context, was employed to enable simultaneous comparison of classification accuracy and accuracy of the predictions of relative abundances. Intuitively, the WSD represents a measure of the average nucleotide identity between two populations, where a population can be seen as the constituent organisms' combined genomes in amounts relative to the organisms abundance.[1] To calculate the WSD, we pick a multiplicity $m = 100$. We can then create the multiset v, where each entry v_i is a sequence of a strain, with multiple entries per strain possible. The total number of sequences of strains corresponds to the multiplicity m. To create the multiset, for each strain i we add $\lfloor mP_i \rceil$ instances of the sequence of the strain i, where $\lfloor mP_i \rceil$ is the closest integer to mP_i and P_i is the relative abundance of strain i. Intuitively, this means that we add a number of copies of assembly

[1] The WSD may also be interesting as a measure of similarity between metagenomic samples, with a basis of reads or k-mers instead of strains. This interesting research avenue remains to be investigated.

i to the multiset v corresponding to its rounded percentage relative abundance. For example, if a strain has a relative abundance of $P_i = 0.333$, then $\lfloor mP_i \rceil = 33$, i.e. 33 copies of the sequence of strain i are added to the multiset. We finally take the WSD between two such multisets v and w as

$$WSD(v, w) = \frac{1}{L} \inf_{\pi} \left(\frac{1}{m} \sum_{i=1}^{m} D(v_i, w_{\pi(i)}) \right),$$

where L is the average genome length of the species considered (e.g. 5.1 million basepairs for *E. coli*). $D(x, y)$ is the Mash distance distance between sequences x and y (specifically, the sequences of the assemblies of strain x and strain y) and π is a permutation of the indices of w. The ground truth v of a sample is created from the true strains in the samples together with their true abundances. The multiset w is created from the inferred strains and their proportions. In the result section we will not only need to compare the tools to each other, but also need to interpret the quality of the WSD independently for one of the tools. To make the WSD more interpretable, we require a comparison measure. To this end, I computed a set of WSDs between randomly chosen populations consisting of between 1 and 5 strains. These WSDs will represent a measure of what would be expected to be observed if a tool simply picked a random set of strains. To estimate these expected WSDs, the following process was used. We aim to estimate the expected WSD between two populations of strains. To this end, we can randomly sample two populations. First, we randomly sample the number of strains for each population, ranging from 1 to 5 strains (or equivalently, assemblies). Then, for each population we randomly select the corresponding number of strains from the total database of assemblies for the species we aim to compute the expected WSD from. The strains are assumed to be present at equal relative abundance. Given the two populations we can then compute the WSD between the populations. By repeating this procedure 10000 times we obtain a distribution of expected WSDs which we can compare to the WSDs observed in the inference of strains using the different strain identification tools.

Results

I first address any tools that failed throughout the benchmarking process. Sigma could not be installed. Furthermore, while MIST could be installed and a database could be created, it was not possible to analyze the samples using MIST, due to high runtime. Reducing the size of the input data as recommended in the MIST documentation and beyond, as well as reducing the database size did not result in MIST running to completion within 72 hours and using 32 cores. Hence, no results for MIST are presented going forward. Furthermore, StrainScan returned an error for 28 of the *A. baumannii* samples (24 of the samples made up

of strains contained in the reference database and 4 of the samples made up of strains not contained in the reference database). For all other species, however, StrainScan could be run successfully for all samples. It is worth mentioning that StrainScan is the most recently released tool and future developments may resolve the current issues. The reasons for the reported errors are unclear. For the samples where StrainScan failed to run the performance was classified as $F1 = 0.0$, $JSD = ln(2)$, $WSD = 1.0$. I now present the results on the simulated datasets, before evaluating the tools on mock metagenomic data.

Simulated Data

I first investigated the overall performance of the tools across all samples made up of strains contained in the reference databases (Figure 2.4A). StrainGE and StrainScan show the best performance both for F1 and JSD metrics, whereas KrakenUniq and StrainEst show low performance, especially beyond the lowest clustering level. Only for *A. baumannii*, StrainEst remains competitive at the 10 cluster level. Interestingly, StrainGE generally outperforms StrainScan on coarse grained clusterings (the 10 and 100 cluster levels, particularly at the 10 cluster level StrainGE outperforms StrainScan for every species), but shows lower accuracy on the assembly level. This points to potentially different ideal use-cases for the two tools, where StrainScan may be used to track a particular strain of interest, for example in an outbreak, while StrainGE may be more applicable to cases where clades or other lower resolution groupings are of more interest. This disparity may also be based on the clustering performed in the creation of the StrainGE database. This is supported by *M. tuberculosis* showing a drop-off in performance at the 100 cluster level, which is to be expected given that only 133 clusters were created. Similarly, *E. coli*, with the most clusters, shows a noticeable drop in performance only beyond the 500 cluster level. The strong drop-off here may also be related to the relatively low Jaccard similarity cutoff of 0.75. This drop is also observed for StrainScan, which is likely due to the fact that the database was down-sampled to 895 strains, which naturally precludes achieving perfect accuracy at the assembly level. Additionally, *E. coli* shows the highest gap in resolution between the 500 cluster level and the assembly level (500 vs 3671 strains) compared to other species, which also explains the uniqueness of this drop among species. For *L. monocytogenes* clear drops in accuracy and increases in JSDs are observed at more fine grained clustering levels for StrainGE, likely also due to the clustering. StrainScan on the other hand shows fairly stable performance across taxonomic resolutions for both *L. monocytogenes* and *M. tuberculosis*. Results for *A. baumannii* and *E. coli* were more accurate at the lower clustering level both for StrainGE and StrainScan (disregarding the failed runs for *A. baumannii*) compared to other species, though the difference is small. This could be due to the species population structure. *E. coli* and *A. baumannii* generally show high intra-species variability, while *M. tuberculosis* is not as variable. This may lead to the

intra-species clusters being more differentiable, and hence, lead to higher accuracy. I will investigate the difference of performance with regard to intra-species variability further below. For the samples made up of strains contained in the reference database, StrainGE shows the lowest WSD for three of the four species (*A. baumannii*: Mean = 0.0029, SD = 0.0025; *E. coli*: Mean = 0.0054, SD = 0.0058; *L. monocytogenes*: Mean = 0.00089, SD = 0.00067). Only for *M. tuberculosis* StrainScan has a lower WSD (Mean = 0.00031, SD = 0.00037). Notably, KrakenUniq shows comparatively high performance on WSD scores for *A. baumannii* (Mean = 0.018, SD = 0.0043) even though the F1 and JSD metrics do not suggest very high accuracy. This may be due to KrakenUniq often predicting too many strains (Median = 7, 25%-Percentile = 2, 75%-Percentile = 47) leading to low F1 scores, but, on average, these strains may still show a low divergence and hence the WSD may be low. However, if this was the case then we should observe high accuracy at the cluster level. To investigate this, observed WSDs were compared to the expected WSDs between two random *A. baumannii* populations as described in the outcome measures section. This comparison showed that the WSD expected between two random populations was lower than the one reported by KrakenUniq (Mean = 0.011, SD = 0.0056; Figure 2.4B). Therefore, we can conclude that the strains identified by KrakenUniq differ substantially from the strains actually contained in the samples. Further, a comparison of the results obtained for StrainScan to the results reported by Liao et al.[91] can be performed, as their experiments were similar and they also tested *E. coli* and *M. tuberculosis*. For both species much lower performance than what Liao et al.[91] reported is observed. This may be due to multiple factors, including the difference in databases used. For *E. coli* the database they used for data generation and the reference database was the same, while the data in the present experiments were generated from a larger database than was provided as reference database, due to hardware constrains. However, even at a clustered level lower performance is observed. For StrainGE, as a similarly performing tool, the comparison to the results reported in their respective publications is not possible as their testing was far less comprehensive.

I next investigated the different tools' performances when analyzing the samples made up of strains not contained in the reference database (Figure 2.5A). Overall, performance is clearly reduced for these datasets. However, at a low taxonomic resolution, StrainGE shows high accuracy both for *A. baumannii* and *E. coli*. Similarly, StrainScan still shows high accuracy for *E. coli* and *L. monocytogenes*. Importantly, at a low clustering level, StrainEst shows to be more accurate compared to the samples made up of strains contained in the reference database and has competitive performance to StrainScan and StrainGE. Nonetheless, it is still outperformed by StrainGE on the majority of measures. Generally, performance across species and tools varies more. Especially for *M. tuberculosis* low performance beyond the coarsest clustering level can be observed. Only StrainScan shows good performance at the 100 cluster level. This could be due to the

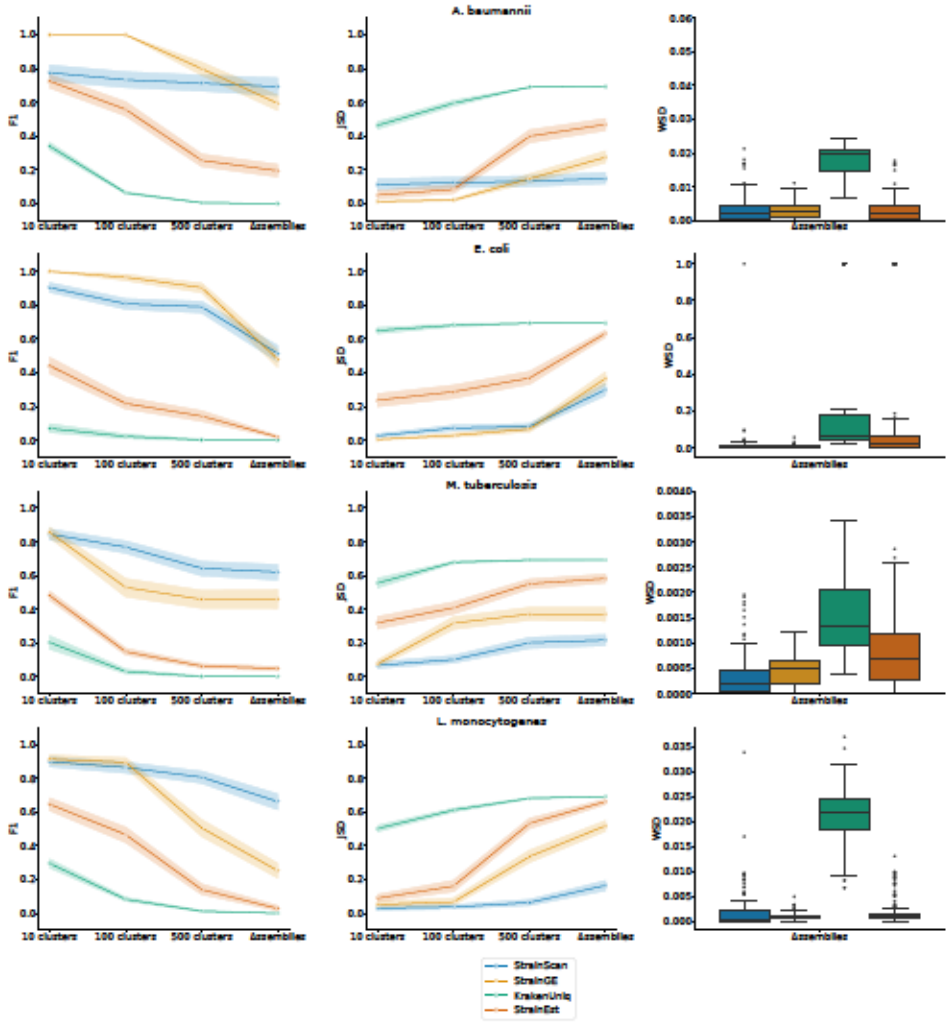


Figure 2.4: Results for samples made up of strains contained in the reference database across tools. **(A)** F1-scores, JSDs, and WSD for the different tools for the samples made up of strains contained in the reference database. The WSD for KrakenUniq in *M. tuberculosis* was comparatively high. For visibility of the results the y-axis has been truncated for this figure, such that outliers are not shown for KrakenUniq. Similarly, the samples of *A. baumannii* for which StrainScan failed to run were assigned a WSD of 1.0 and are hence also not shown. Shaded areas of lineplots indicate 95% confidence interval. For the boxplots, line indicates median, edges of box the 25% and 75% quartile, and whiskers extend to include the datapoint furthest from the median that is within 1.5 times the interquartile range. Dots are outliers. Assemblies on x-axis indicates assembly level analysis (see Outcome Measures section). **(B)** Expected WSDs for *A. baumannii* samples from 10000 random choices of paired strain populations (see Outcome Measures section) compared to the observed WSDs across samples composed of strains contained in the reference databases for the four different tools. Line indicates median, edges of box the 25% and 75% quartile, and whiskers extend to include the datapoint furthest from the median that is within 1.5 times the interquartile range. Dots are outliers.

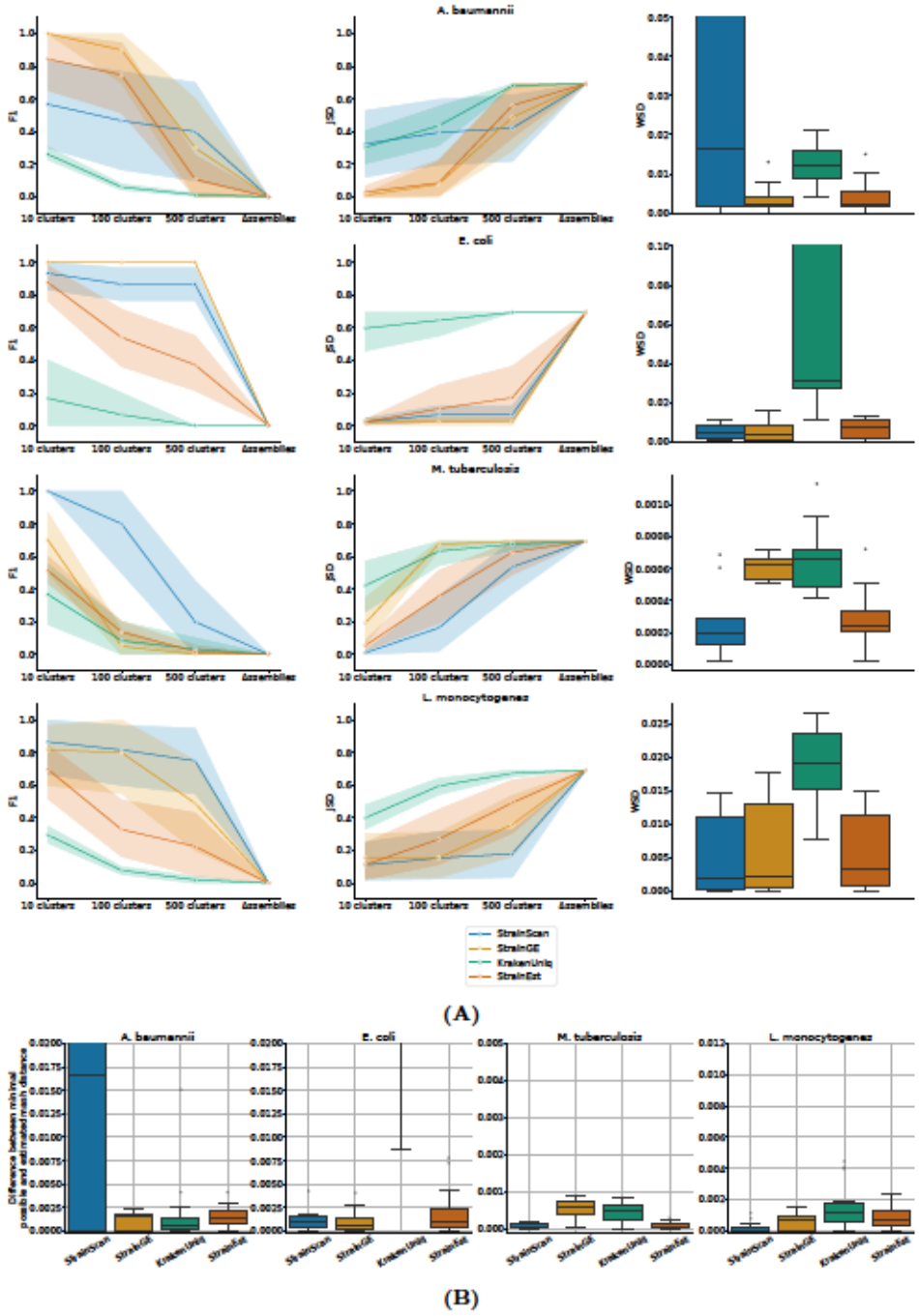


Figure 2.5: Results for samples made up of strains not contained in the reference database across tools. **(A)** F1-scores, JSDs, and WSD for the different tools for the samples made up of strains not contained in the reference database. For the boxplots, line indicates median, edges of box the 25% and 75% quartile, and whiskers extend to include the datapoint furthest from the median that is within 1.5 times the interquartile range. Dots are outliers. WSD for *A. baumannii* does not show WSDs for StrainScan for samples which failed to run. Assemblies on x-axis indicates assembly level analysis (see Outcome Measure section). The WSD for KrakenUniq in *M. tuberculosis* and *E. coli* was comparatively high. For visibility of the results the y-axis has been truncated for this figure. **(B)** The difference between the minimum possible distance any estimated strain has to the true strained contained in a sample and the minimum possible distance achievable given all strains present in the reference database (see text for details). Y-axes are truncated for *A. baumannii* and *E. coli* for visibility. Note the difference in scale for the different panels for higher visibility.

homogeneous population structure of *M. tuberculosis*. To gain more insight, performance was compared to the best achievable performance given the reference database and the strains in the samples. For each strain in the sample (i.e. strains not in the reference database) the strain in the reference database with the lowest Mash distance to the strain in the sample was determined. In an ideal scenario, the tools would infer this closest strain as present in the sample and the distance determined is therefore the lowest obtainable Mash distance. However, other reference strains may be similarly well suited to describe the strains in the sample. As such, for every strain in a sample this strain's Mash distance to all the strains inferred by a tool in the sample was computed. Then, the lowest among these Mash distances was chosen as the Mash distance of this strain to the most related strain in the sample. Finally, to determine the quality of inference compared to the best possible inference, the Mash distance of each strain in the sample was compared to the lowest possibly obtainable Mash distance of that strain (Figure 2.5B). Indeed, across the different species the minimal differences of the inferred strains Mash distances to the lowest obtainable Mash distances were observed for *M. tuberculosis*, indicating that the low performance on higher clustering levels does not necessarily reflect the accuracy of identification considering the genetic distance between the strains. Before analysing samples with potentially unknown strains, one may want to make further considerations regarding the species in question and adjust the analyzed clustering level. When investigating performance via these Mash distances, StrainEst shows good results compared to the other metrics, though this may be due to the fact that the tool overestimates the number of strains (Median = 8, Range: 1-52), and hence, may just by chance be more likely to include at least one strain with a low Mash distance. On the other hand, StrainScan shows the highest capabilities in recovering closely related strains from the databases for both *M. tuberculosis* and *L. monocytogenes*. We can again compare the performance to the results presented by Liao et al.[91]. However, they only conducted analyses for *E. coli*. Here, they report lower Mash distances than were found in the current investigation. Additionally, contrasting the results presented by Liao et al.[91], StrainGE shows a lower median Mash distance than StrainScan in the present analysis.

In identification of strains we desire both that the strains are close in their sequence similarity (as shown by F1-scores and WSDs), but also that we estimate the correct number of strains, which will help in investigating the population structure within a microbiome, as well as in identifying the co-existence of multiple strains. Knowing whether more than one strain is present in a sample also may have other applications. For example, when creating MAGs one may want to employ this knowledge to determine whether a given MAG may be chimeric. The estimated versus actual number of strains across species are shown in Figure 2.6A for StrainScan and StrainGE, as the overall best performing tools. Overall, both StrainGE and StrainScan estimate the correct number of distinct strains well, with StrainGE generally more likely to overestimate

the number of strains, while StrainScan is more likely to underestimate. KrakenUniq and StrainEst both overestimate the number of strains to a large degree (Figure 2.6B and Appendix Figures A8 and A9). For KrakenUniq this may be due to the fact that strains are identified simply on the number of unique k-mers identified, with no preference for sparsity and no real dependency between the strains. However, strainEst does aim for parsimonious solutions, so its overestimation is harder to explain. It may indicate that the underlying regression approach is not suitable for strain inference in metagenomics. Notably, StrainScan shows to be highly accurate at classifying single strain samples as such, though this is not surprising considering its tendency towards sparser results. For samples made up of strains not contained in the reference database, both StrainGE and StrainScan estimated the number of strains well for most species (Figure 2.6A). However, for *M. tuberculosis* StrainGE identified two strains in all samples. Similarly, StrainGE tends to overestimate the number of strains for *L. monocytogenes*.

The initial results show that StrainGE and StrainScan both clearly outperform KrakenUniq and StrainEst across metrics and species on both data generated from strains in the reference databases or not. Hence, I focus the rest of the investigation in this section on StrainScan and StrainGE. Firstly, we can establish to what degree the results obtained by StrainScan and StrainGE agree at different levels of clustering to see to what extend the results depend on the chosen tool. For the samples made up of strains contained in the reference database there is a high average similarity between the identified clusters at a low clustering level. Across species, for the identification among 10 clusters, between 39% and 57% of results match exactly between the two tools, while between 0.5% and 14% disagree completely. This suggests at least partial interchangability between tools. However, this similarity drops significantly at the 500 cluster level (30% of all samples match completely versus 47% for 10 clusters and 26% of samples disagree completely versus 5.9%) and already at the 100 cluster level for *M. tuberculosis* (17% versus 39% complete matches at the 10 cluster level and 38% complete mismatches versus 6.5%). Hence, tools cannot simply be exchanged for all species in all circumstances. Next, I aimed to determine how the tools' performance relates to the different species' characteristics. To investigate the hypothesis that relatedness within species influences the results, the strains were clustered at 10000 different Mash distance thresholds, i.e. the assemblies of a species were clustered such that all assemblies with a Mash distance below the cutoff fall into one cluster. 10000 linearly spaced Mash distances between 0 and 0.02 were used as clustering thresholds. For the clustering, complete linkage hierachical clustering was performed for each threshold, similar to the clustering in the previous analyses. The F1 scores for StrainScan and StrainGE were computed at each clustering level, providing a measure of identification performance at different resolutions. My hypothesis was that the performance decline should be similar across species with regard to the Mash distance clustering threshold. StrainGE

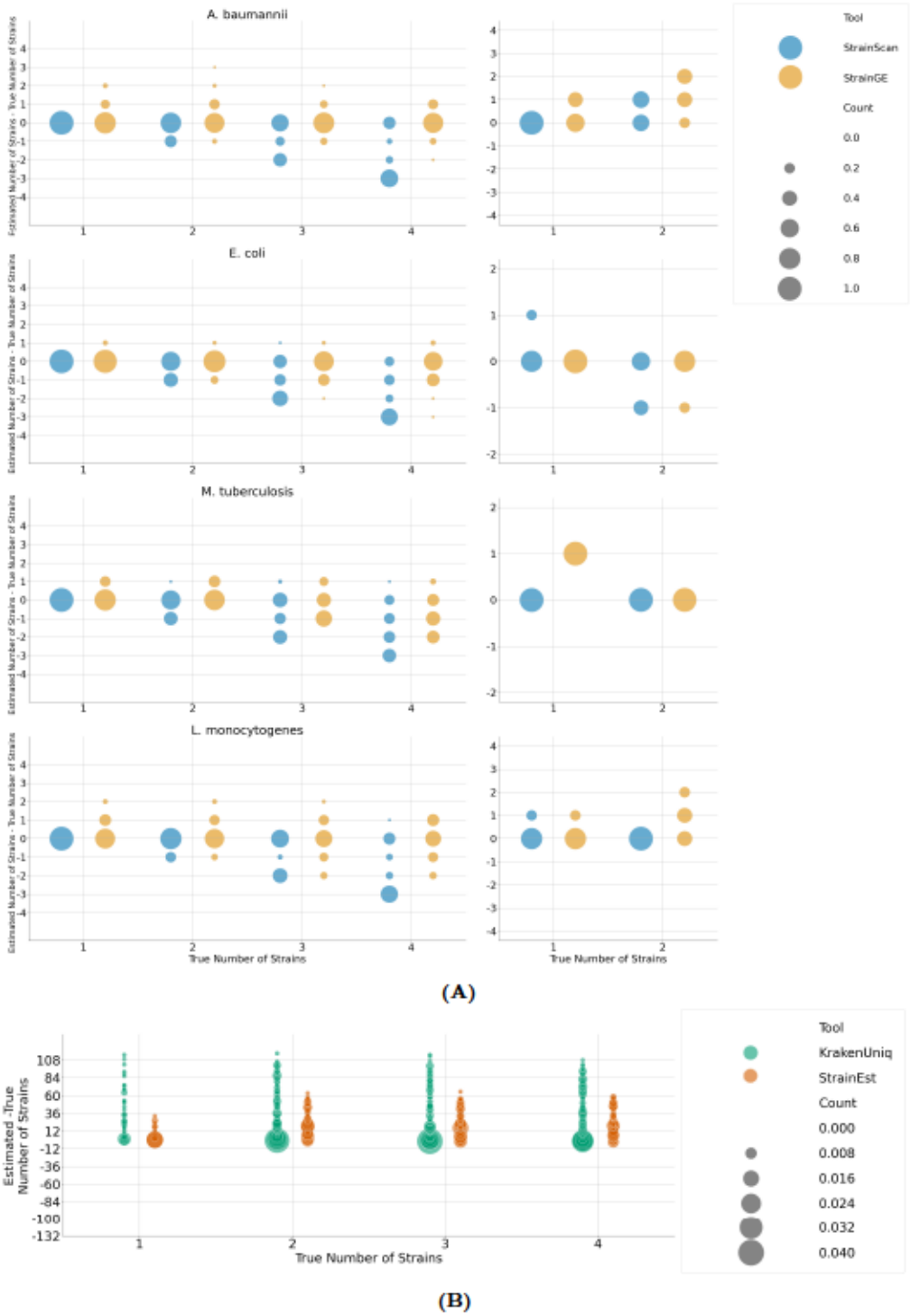


Figure 2.6: Number of strains estimated for StrainScan and StrainGE **(A)** across species for the samples made up of strains contained in the reference database (left columns) and the samples made up of strains not contained in the reference database (right column; size of circles represent fraction of samples). **(B)** Number of strains estimated for KrakenUniq and StrainEst cross species for samples with strains contained in the reference database (size of circles represent fraction of samples).

shows a sharp dropoff in performance at around the same distance threshold across all species besides *E. coli* at a threshold around 0.004 (Figure 2.7), whereas the performance of *E. coli* drops already around 0.01

which again can be related to the much lower threshold used during database creation. This relationship to the clustering threshold used also indicates that the threshold used in database creation is an indicator of the resolution achievable by StrainGE, and as such should be carefully considered and be reported in any research conducted using StrainGE. Notably, performance seemed to cluster more by the tool used rather than by species, at least to some degree. For the number of clusters a much stronger split between species was observed than for the distance threshold, indicating in agreement with the hypothesis that the latter is a more reliable measure across species. In practice the population structure of the species of interest should be considered and the level of taxonomic resolution that can be interpreted confidently should be adapted accordingly. To determine an appropriate resolution one may conduct simulations for the species of interest before attempting identification in real data. While it is hard to provide an exact dissimilarity between clades or members of a species for which they will be differentiable, one should err on the side of caution when investigating highly clonal species such as *M. tuberculosis* and only trust taxonomic identifications at lower clustering levels. Finally, it may be interesting to investigate in each species whether there are certain clusters or strains that are more easily identified correctly. However, the current dataset would need to be expanded systematically with this question in mind to sufficiently address it. This may be an interesting avenue for future research to explore.

Lastly, the datasets were simulated using different parameters, such as the number of strains or the coverage. Detailed results for the different parameters can be found in Appendix Figures A10-A25. Across species StrainScan's performance drops with an increased number of strains, as well as with lower coverage (not considering samples with 1 strain and 1-fold coverage, as only single strain samples were tested at 1-fold coverage). Additionally StrainScan shows essentially perfect performance at 1-fold coverage for single strain samples. StrainGE shows a more robust performance across scenarios. This stability is especially important when considering different relative abundance profiles. Here, StrainScan shows a appreciable decline in performance for samples containing one dominant strain at 80% relative abundance, whereas StrainGE shows little variability between conditions. The JSD for StrainScan however decreases or stays constant across conditions for many species and clustering levels. This indicates that while strain identity is harder to distinguish for low abundance strains, the overall abundance composition is estimated well. StrainGE again shows relatively similar performance across abundance profiles. It is notable that the performance of StrainEst actually drops with increased coverage. This is likely due to more false positive identifications. A direct comparison shows that at low coverage StrainScan outperforms StrainGE, while StrainGE shows higher and more consistent performance at higher coverages and for samples containing more than one strain at sufficiently low clustering levels (with the exception of *M. tuberculosis*; Figure 2.8).

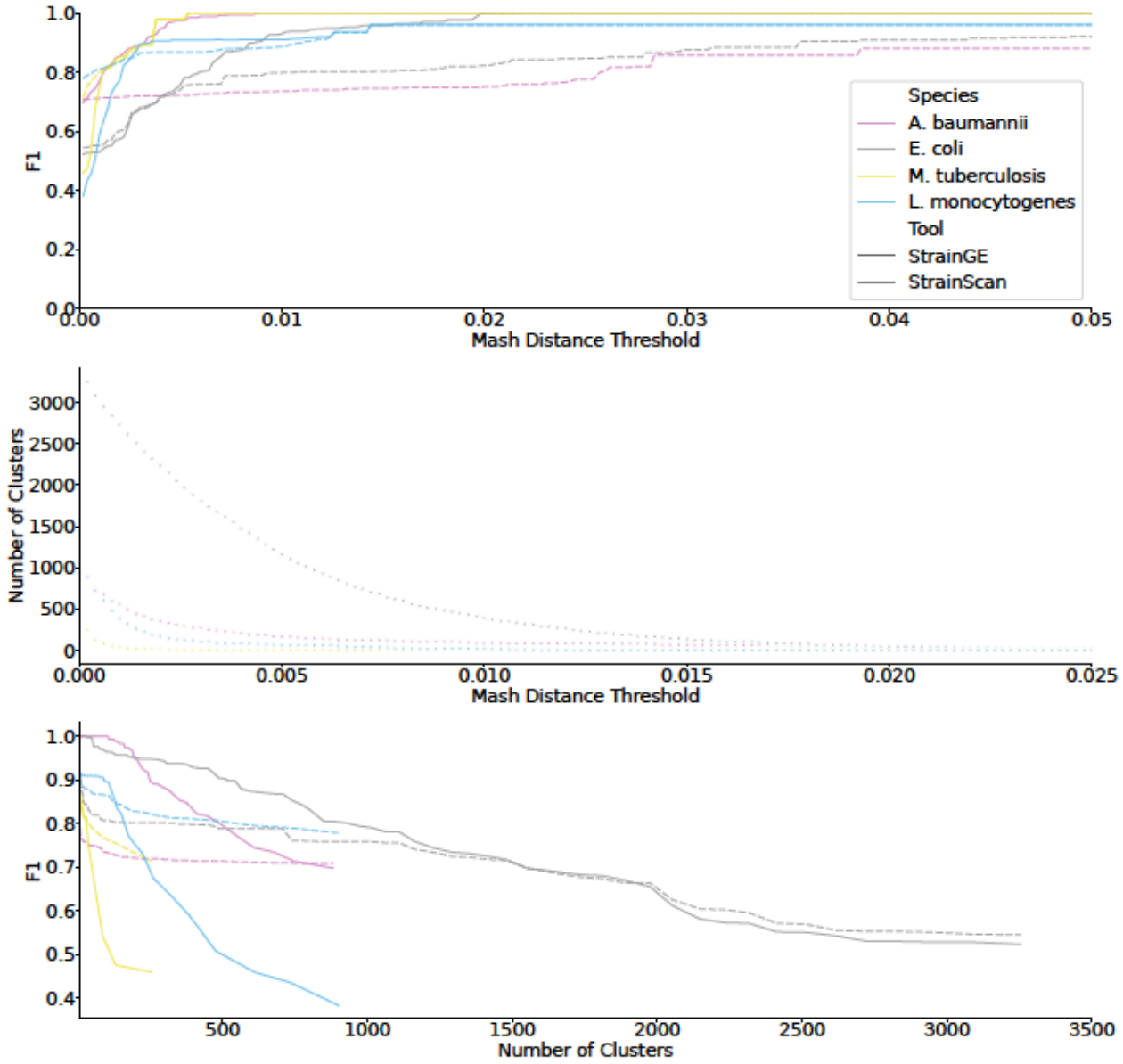


Figure 2.7: Figures show relationships between F1 scores, cluster sizes and Mash distance thresholds used for complete linkage hierarchical clustering across species for StrainScan and StrainGE. The first figure shows the F1 score depending on the Mash distance threshold used for clustering. Each Mash distance threshold results in a different number of clusters, which is shown in the second figure. The third figure then shows the F1 scores in relation to the number of clusters that were inferred. Note here that the same clustering threshold in two different species can lead to different numbers of clusters. Hence, the difference in the number of clusters results are plotted for in the bottom panel.

Complex strain mocks

In any practical application, metagenomic samples are not just made up of a single species, but contain a mixture of organisms. Additionally, while ART may provide accurate simulations of metagenomic sequencing techniques, it cannot mirror all the intricacies of real metagenomic sequencing. Hence, it is essential to

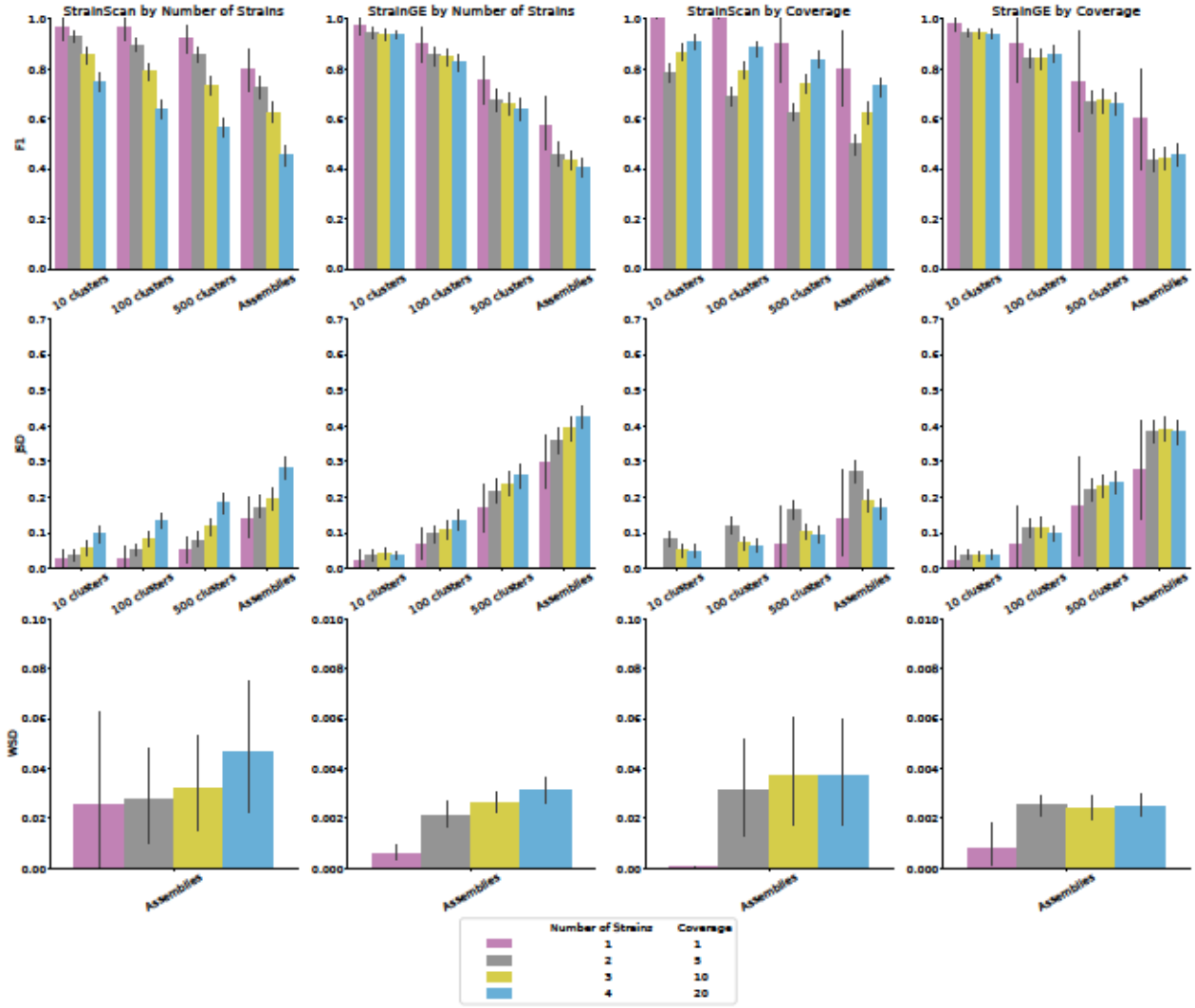


Figure 2.8: Results for samples made up of strains contained in the reference database combined across all species for StrainScan and StrainGE, stratified by number of strains in the samples as well as total coverage. Here, all samples irrespective of the species making up the sample are analyzed. Similarly, samples analyzed include both samples with strains originating in the same and in different clusters at the 100 cluster level. The coverage shown is the total coverage of the sample, with both samples analyzed that have equal relative abundance for all strains as well as the samples that have one dominant strain at 0.8 relative abundance with the remaining strains having equal shares of the remaining 0.2 relative abundance. The high performance for the 1-fold coverage samples can be explained by the fact that only single strain samples were simulated at 1-fold coverage, whereas higher coverage samples were simulated for samples containing 1, 2, 3, and 4 strains.

evaluate the different tools on complex strain mocks. Here, I used the NIST dataset [183]. The results for all samples are shown in Table 2.4. The best performing tools are again StrainGE and StrainScan. Both tools provide mostly accurate number of strains, while slightly overestimating the strain diversity in some cases. This is in contrast to KrakenUniq and StrainEst which both often identify too many strains for species that are contained in the samples and also frequently deliver false positive results (51 false positive across all samples versus 2 false positive identifications for StrainScan and StrainGE; a tighter E-value of 0.001 for

KrakenUniq did not qualitatively change this observation). StrainScan and StrainGE both showed 2 false negatives for *L. monocytogenes*. However, these were for samples where *L. monocytogenes* was present at 0.03% relative abundance (approximately 0.06-fold coverage). Furthermore, neither StrainScan nor StrainGE infer strains of *M. tuberculosis*, which were not present in any sample, while StrainEst and KrakenUniq do. The lowest Mash distances and WSDs were not necessarily observed for tools estimating the correct number of strains, and sometimes achieved by StrainEst or KrakenUniq. This discrepancy for the Mash distances can be explained by the fact that the minimal Mash distance between all estimated strains and all strains in the sample was calculated, favouring tools which overestimate the number of strains. For the WSD, even when higher number of strains are observed these can still be on average closer to the true strain in the sample, and hence, lead to a low WSD. This shows the importance of considering all measures in combination. Notwithstanding these observations, overall the results on the mock dataset are consistent with the results for simulated data.

NIST Sample	Species	Tool	True Abundance	# Inferred Strains	Mash difference	WSD
A	*A. baumannii*	StrainScan	2.6%	3	0.012	0.018
		StrainGE		**1**	**0.000024**	**0.000096**
		KrakenUniq		99	0.011	0.019
		StrainEst		1	0.00012	0.00019
	E. coli	StrainScan	30.4%	**1**	0.0035	0.0035
		StrainGE		2	**0.00083**	0.012
		KrakenUniq		4	0.024	0.029
		StrainEst		7	0.00096	**0.00096**
	L. monocytogenes	StrainScan	0.03%	0	N/A	N/A
		StrainGE		0	N/A	N/A
		KrakenUniq		**2**	0.026	0.042
		StrainEst		**2**	**0.0023**	**0.000092**
	M. tuberculosis	StrainScan	0.0%	0	0	0
		StrainGE		0	0	0
		KrakenUniq		4	N/A	N/A
		StrainEst		0	0	0
B	*A. baumannii*	StrainScan	30.1%	**1**	**0.000024**	**0.000096**
		StrainGE		**1**	**0.000024**	**0.000096**
		KrakenUniq		225	0.0053	0.019
		StrainEst		2	0.0039	0.00019
	E. coli	StrainScan	0.04%	**1**	0.23	0.23
		StrainGE		2	0.13	0.11
		KrakenUniq		3	**0.029**	**0.032**
		StrainEst		60	0.016	0.023

NIST Sample	Species	Tool	True Abundance	# Inferred Strains	Mash difference	WSD
B	*L. monocytogenes*	StrainScan	0.0%	0	0	0
		StrainGE		0	0	0
		KrakenUniq		2	N/A	N/A
		StrainEst		2	N/A	N/A
	M. tuberculosis	StrainScan	0.0%	0	0	0
		StrainGE		0	0	0
		KrakenUniq		3	N/A	N/A
		StrainEst		0	0	0
C	*A. baumannii*	StrainScan	0.03%	0	N/A	N/A
		StrainGE		0	N/A	N/A
		KrakenUniq		34	0.0096	0.020
		StrainEst		**1**	**0.00012**	**0.00019**
	E. coli	StrainScan	0.0%	1	N/A	N/A
		StrainGE		1	N/A	N/A
		KrakenUniq		2	N/A	N/A
		StrainEst		44	N/A	N/A
	L. monocytogenes	StrainScan	3.9%	**1**	**0.0011**	0.0011
		StrainGE		**1**	0.0012	0.0012
		KrakenUniq		**1**	0.0051	0.0051
		StrainEst		4	0.0015	**0.000048**
	M. tuberculosis	StrainScan	0.0%	0	0	0
		StrainGE		0	0	0
		KrakenUniq		2	N/A	N/A
		StrainEst		1	N/A	N/A

NIST Sample	Species	Tool	True Abundance	# Inferred Strains	Mash difference	WSD
D	*A. baumannii*	StrainScan	0.0%	0	0	0
		StrainGE		0	0	0
		KrakenUniq		32	N/A	N/A
		StrainEst		3	N/A	N/A
	E. coli	StrainScan	2.76%	1	0.0035	0.0035
		StrainGE		1	**0.00083**	**0.00083**
		KrakenUniq		3	0.029	0.031
		StrainEst		16	0.0024	0.00095
	L. monocytogenes	StrainScan	30.0%	2	**0.00024**	0.00056
		StrainGE		**1**	0.0012	0.0012
		KrakenUniq		2	0.026	0.047
		StrainEst		4	0.0014	**0.000048**
	M. tuberculosis	StrainScan	0.0%	0	0	0
		StrainGE		0	0	0
		KrakenUniq		3	N/A	N/A
		StrainEst		0	0	0
E	*A. baumannii*	StrainScan	7.6%	**1**	0.00039	0.00046
		StrainGE		**1**	**0.000024**	**0.000096**
		KrakenUniq		160	0.0029	0.018
		StrainEst		**1**	0.00012	0.00019
	E. coli	StrainScan	8.8%	**2**	0.0035	0.089
		StrainGE		3	**0.00083**	0.034
		KrakenUniq		3	0.024	0.031
		StrainEst		57	0.0017	**0.0094**
	L. monocytogenes	StrainScan	9.6%	**1**	**0.0011**	0.0011
		StrainGE		**1**	0.0012	0.0012
		KrakenUniq		2	0.026	0.047
		StrainEst		5	**0.0011**	**0.000048**
	M. tuberculosis	StrainScan	0.0%	0	0	0
		StrainGE		0	0	0
		KrakenUniq		3	N/A	N/A
		StrainEst		1	N/A	N/A

Table 2.4: Results for NIST samples. True abundance indicates the percent of the sample (including all species) the strain of the species listed in the row makes up (e.g. the *A. baumannii* strain makes up a total of 2.6% of sample A). Note that percentages need not to sum to one within a sample, as more than the listed species are contained in the sample. # inferred strains column shows the number of strains for the species each tool inferred (one strain would be optimal). Mash difference indicates the difference between two Mash distances. The first Mash distance is the minimum of the distances between the inferred strains and the strain in the sample. The second is the minimum distance between the strain in the sample and all strains in the reference database. The first measures how close to the strain in the sample the estimated strain is. The second measures the Mash distance obtained by the best possible inference given the database. As such this column measures how close to the best possible result the inference is. Similarly, the WSD measures the same difference, however, this time for the WSD of the inferred population to the true strain and the best possible WSD given the database strains. In this manner, the number of strains inferred is taken into account. Bold fonts indicate the best performing model for a metric-sample-species combination.

Discussion

I compared six state-of-the-art tools for strain identification in metagenomic data using a variety of measures and datasets. The results show that in general strain identification is often possible, with both StrainGE and StrainScan showing high performance in many different conditions. On the other hand, KrakenUniq and StrainEst have several drawbacks, including a high false positive rate. SIGMA and MIST failed to run completely. StrainGE provides reliable results up to a Mash distance resolution of 0.01 (around 99% ANI), and potentially higher with sufficiently fine-grained clustering (in this research up to around 99.6% ANI was achievable). Additionally, StrainGE works best on mock metagenomes. StrainScan performs similarly, and often provides higher taxonomic resolution, while still giving reliable results on mock data. At low coverage StrainScan does outperform StrainGE. Hence, my recommendation is to apply StrainScan for low coverage samples were only a single strain is suspected, while StrainGE is better suited to samples where multiple strains are suspected to be present, as well as for higher coverages. Similarly, StrainScan performs better at high resolution clusterings, while StrainGE shows to be best at lower resolutions. A notable exception to this pattern is *M. tuberculosis*, where StrainScan outperforms StrainGE more consistently. The present investigation suggests that this is due to the homogeneous population structure of *M. tuberculosis*, with little variation between isolates. This provides a challenge to efficient clustering, as well as high precision identification. Based on this, I generally recommend using StrainScan for highly conserved species such as *M. tuberculosis*. Still, caution should be taken when interpreting results at a high taxonomic resolution and lower resolution identification will be more reliable, while still providing valuable insights into intra-species diversity. Altogether, when applied in appropriate scenarios, such as for samples with sufficient species coverage, strain resolution metagenomics can be possible. Nonetheless, the species of interest will have an impact on our ability for strain-level recovery. Hence, before applying any of the tested tools to a new species, testing should be conducted.

One important observation is the stark difference in results to the ones presented by Liao et al.[91]. We generally show lower performance across species for StrainScan. The reasons for this discrepancy are unclear. Potential reasons include the more challenging simulated data setups, for example non-equal relative abundances across strains were not considered in the work of Liao et al.[91], and differences in the databases. Especially for *E. coli* the required reduction in the database size likely had an effect on the observed accuracy.

Notwithstanding the positive results there are still several hurdles in strain recovery from metagenomic data and the different tools have drawbacks in some areas. First of all, while performance overall is good, it is hard to know whether the results for a particular sample are correct. None of the tools provide confidence

estimates for their inference making it hard to judge the quality of the results in practice. StrainGE does return a score for each of the inferred strains and a more stringent cutoff will reduce the number of low quality identifications. However, determining a good cutoff in practice is hard and especially at low coverages the data may show more variability. Similarly, KrakenUniq requires more interpretation of the results, and a custom cut-off value had to be chosen. Furthermore, while identification of reference strains is important, strains found in metagenomic data will necessarily differ from the known reference strains to some degree. Here, only StrainGE offers a tool for identification of SNPs and large deletions. However, none of the tools provide the possibility of estimating large insertions, such as those that occur upon the acquisition of new genes and none of the tools distinguish between core and accessory genome. For the tools to be applied, species specific considerations on the database, the clustering, and the expected performance need to be made, which can be a difficult and time consuming task. Criteria build into the tools that make the analysis for different species easier would be beneficial. Finally, of the tools that could successfully be run only KrakenUniq provides assignment of reads to strains. Such an assignment would be valuable for downstream analyses, but is missing from StrainScan, StrainGE, and StrainEst.

In conclusion, the conducted benchmark shows the potential of strain identification in metagenomic data. The available tools can be applied for metagenomic analysis and provide useful information. Nonetheless, specific consideration of research goals and the species under investigation should always be taken into account by researchers. There is still a lot of potential for future developments building on the already achieved capabilities of the presented tools. Identification of strains in metagenomic data will further our understanding of microbiomes and bringing surveillance and potentially diagnostics using metagenomics[62] into the realm of possibility by enabling basic research on these topics using the presented tools.

Chapter III

Core Genome based Strain Identification in Metagenomes

'We swim nonstop in an ocean of small, medium-sized, and large analogies, ranging from mundane trivialities to brilliant insights.'

– Douglas R. Hofstaedter

The investigation into previously published tools for strain identification revealed the need for a novel approach addressing some of the gaps in the current literature. In most cases, the available tools do not consider the gene content of strains in a sample. Only the StrainGE toolbox addresses this issue, and then only for deletions of genes. In practice, assignment of genes to strains is essential, at it could be used in a variety of ways, be that for wastewater based surveillance or general research on the function carried out by a bacterial species in a particular environment. Additionally, while the performance of several of the available strain identification tools is high, in practice large recombination events and changes in the accessory genome may impact real world strain identification if the recombination event or change in the accessory genome has not been observed in previously sequenced isolates.

Lastly, interpretability in models is a desirable property, i.e. a model should provide information that can be easily connected to scientific concepts. At the same time a good interpretable model will show no (or only a slight) decrease in descriptive accuracy[186]. While some tools can call SNPs and others can at least partially assign reads, the interpretability in the available tools can generally be improved. The available models provide little insight into the reasons behind their inferences, i.e. how the input data connects to the reported strain mixture (a strain mixture being defined as a relative abundance distribution over the strains identified in the sample) and read assignment is often not possible. In a similar vein, adaptability to different

research goals can further be beneficial, and an interpretable model with an appropriate structure can address this need. This is true not only regarding read assignment, but also with a view towards gene content. In the context of the current work, gene content refers to coding sequences found in the genetic sequences of the strains represented in the metagenomic data. While the presence of protein coding sequences does not necessarily imply that these are expressed, identifying them is a good first step towards an assessment of the functional role of strains in a microbiome. Current tools also do not distinguish between core and accessory genome. The core genome is the set of genes found in all members of a species. As such they can be considered essential to the species definition and highly relevant in survival of the organism (though a distinction to essential genes is sometimes made in the literature[187]). Importantly, as we will see in Chapter IV, and as has been seen in other literature for example for *E. coli*[153], the core genome does depend on the members of a species that are sampled. With an increase in sampled strains, the size of the core genome will go down. As such, the core genome may be sampling dependent but will still represent genes found across a wide range of species members. Importantly, research suggests that core genes are indeed regularly expressed[188]. The accessory genome on the other hand corresponds to genes that are not present in all members of a species. Importantly, they may differ between environments and determine different functional roles a bacterial strain plays as well as provide targets for selection to act on[189]. The inclusion of accessory genes in the model is further addressed in Chapter IV.

I aim to address the above issues by developing a novel tool. Model development is tackled in two main phases. In this chapter, only the core genome of a given species is considered for strain identification. Importantly, the model should applicable to any species through an automated pipeline for core genome extraction. Focusing on the core genome only first will provide a clearer development path for model development and additionally will provide an option for identifying strains, or clusters of strains, based on just the core genome. While an MLST strain type identifier exists for metagenomic data[83], it is limited to the MLST housekeeping loci, as opposed to considering a wider set of core genes or protein coding sequences, and naturally limited to species for which MLST schemes are available. In the first instance, four different models were developed. These are based on either a read-strain mismatch or a single-nucleotide-proportion basis. Either of these approaches has its own advantages. Methods that are based on reads, and perform assignment of reads to strains, may allow for improved, strain-based assembly or other analyses involving specific reads. On the other hand, SNP based methods may allow the direct inference of SNPs in the strains found, though one could imagine doing this in a postprocessing step for read-based methods as well. The different models were compared on a set of data simulated from *A. baumannii* assemblies. *A. baumannii* generally served as example species for model development. *A. baumannii* was chosen for its relevance as a species harboring antibiotic resistance,

with carbapenem-resistant *A. baumannii* having *Priority 1: Critical* status on the WHO's list of bacterial pathogens urgently requiring novel antibiotics[156]. Additionally, a summary of high-quality *A. baumannii* whole genome assemblies by Hernández-González et al.[154] provides a reliable reference database of strains. Finally, our collaborator Santiago Castillo Ramirez is an expert on *A. baumannii* and could provide valuable advice during model development. All of the presented models have an MILP at their core. The flexibility in encompassing biological problems, the efficient methods and tools developed to solve MILPs, and their high level of interpretability make MILPs an ideal basis for model development.

Methods

In total four different MILP models were developed. These can be easily grouped into two classes. The first class of models is based on per-locus nucleotide proportions observed in metagenomic reads, where each locus is one position in a reference multiple sequence alignment (MSA) of a core gene. The second class of models works on a basis of observed number of basepair mismatches between each metagenomic read and each reference sequence. These latter approaches then aim to decode strains on a gene-by-gene basis, by inferring strains as a mixture of alleles. Both types of models will be interpretable by providing insights into which loci or genes drive the decision process, which may be useful in postprocessing and interpretation of the models. Furthermore, the models based on reads are designed to enable assignment of these reads to specific strains. As a basis for the models both information on the per-locus nucleotide proportions as well as on the mismatches between reads and reference sequences are required. First, I will describe the preprocessing by which a curated database of core gene reference sequences for *A. baumannii* was created. Then I will describe the four models. Lastly, I will discuss the calibration of the models' hyper-parameters on a tuning dataset of simulated metagenomic samples as well as the performance of the best performing models on simulated test datasets.

Notation

Throughout this chapter and Chapter IV various symbols will be introduced and a unified naming convention will be useful. Sets are typed in calligraphic font (e.g. $\mathcal{T}$) and the cardinality of a set is denoted in boldface (e.g. $\mathbf{T}$). Latin letters (e.g. T or t) represent variables of the models, and Greek capitals (e.g. Δ) input data. Lastly, parameters are represented by lower case Greek letters (e.g. λ). Often variables or data will be indexed, for example as $\Psi_{a,b}$. Unless mentioned otherwise, the indices correspond to elements of the

corresponding set indicated by the same letter (here $a \in \mathcal{A}$ and $b \in \mathcal{B}$). If an operator is over multiple sets it is assumed to go over the Cartesian product of the sets. For example,

$$\sum_{a \in \mathcal{A}, b \in \mathcal{B}} X_{a,b} \tag{3.1}$$

is the sum over entries of $X_{a,b}$ over all combinations of a and b.

Reference Database

The models in the current chapter was build and tested on *A. baumannii*, whereas the next chapter also considers other species. As for the tools compared in Chapter II, the tool I develop here makes use of a reference database of assemblies (again also referred to as strains throughout). The reference assemblies for *A. baumannii* were the same as in Chapter II, i.e. the curated list of assemblies provided by Hernández-González et al.[154]. For the present stage of model development, the total database of 1373 assemblies was used, such that at this stage the 10% of assemblies excluded from the reference database in Chapter II were kept in the database. For the validation of the full model described in Chapter IV, the reduced database is used again to enable an accurate evaluation and comparison to the other tools (see methods section in Chapter IV).

Preprocessing

All four models work on single-copy core genes (SCCGs) extracted from the reference strains. This has the benefit that duplicate genes can be excluded. Further, genomes can then be separated into core and accessory, increasing interpretability. Duplicate genes would skew the observed abundances of reads in a sample, hence complicating analyses. Additionally, core and accessory genes require different approaches, as will be described below and in Chapter IV, due to the presence of accessory genes in only some of the reference strains. The developed method is based on SCCGs and as the tool developed here should be easily expandable to new species or sets of reference strains used by any researcher an automated way of generating core gene sets was required. To generate SCCGs, Prokka[190] was used with default options to extract open reading frames (ORFs). From these, protein coding sequences, which all references to a *gene* from here on refer to for ease of exposition, were extracted. Next, any gene that appeared more than once in at least one strain was discarded to arrive at single copy genes. SCCGs were then determined as those genes that appear in every strain. Next, each SCCG was aligned independently using MAFFT[191] (with

options `--auto --inputorder --nuc --thread` 10). Here, alignment was done in a reference free manner between the sequences extracted from the reference database assemblies. This resulted in 38 SCCG MSAs. All other 1269 single copy genes were classified as accessory. Further, 366 singletons (genes only appearing in one strain) were identified. The histogram of the number of strains genes appear in can be seen in Figure 3.1. Notably, another option for identifying gene sets is presented by using Panaroo [192], a tool for generating the pangenome of a species. Panaroo also provides identification alignments for core and accessory genes. Panaroo's gene identification step also involves Prokka, followed by clustering of the genes using CD-hit [193]. However, creating a pangenome in the first place requires considerable resources and may not be feasible for reference databases of arbitrary size. Hence, this option was not further pursued for this part of method development, but may be a promising approach in future work.

Next, the reference strains were clustered. rHierBAPS [194] was used for clustering of the strains using the sequences of the extracted SCCGs. rHierBAPS aims to identify subpopulations in a set of sequences under the assumption of the existence of subpopulations with distinct allele frequencies. Its clustering algorithm is based on distinct alleles found in the set of sequences provided (here, the SCCGs of the reference assemblies) on the basis of which the sequences are clustered into a number of clusters, where an upper limit for the number clusters is a parameter for the software (set to 20). The clustering was performed to evaluate the models' performance at lower levels of taxonomic resolution and because one of the models below works on a cluster basis rather than an isolate basis.

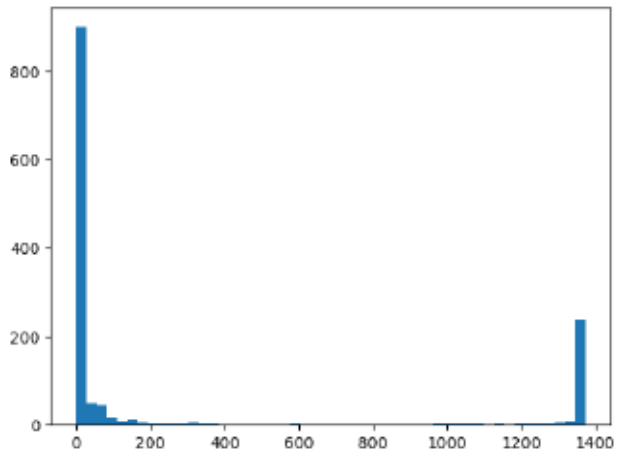


Figure 3.1: Histogram of number of strains genes were found in.

Nucleotide proportion based Models

For the first class of models, metagenomic reads are assigned to the SCCGs using Bowtie2 [195] (a tool that allows fast alignment of sequences against many large reference sequences; options used were `--local --no-unal --no-hd --no-sq -f`), allowing only one match per read. The `-f` option takes fasta files as query sequences, hence the fastq files were converted to fasta files. This was done as only perfect matches between the reads and the SCCGs were allowed, which is unlikely to occur due to a random sequencing error. The MAPQ score was not considered at this stage, as the simulated data used for model validation generally shows very high quality scores and working with fasta files will likely reduce processing time during model development. For analyzing real metagenomic data the quality scores can be utilized during the alignment process with Bowtie2. All reads perfectly matching an SCCG are then added to the MSA of the respective gene created from the reference database assemblies using MAFFT (`--addfragments` option). Only completely matching reads are used to avoid misidentification of sequencing errors as SNPs. Note that mismatches may also be due to actual sequence variation, and sequencing errors could be balanced by correct reads with sufficient depth of coverage. Hence, one may want to include non-perfect matches. The need to include higher number of mismatches likely also depends on the environment under investigation, the species under consideration and our prior believe about the similarity of the strains in a sample to our available reference database. This will be addressed in future research. Based on the added reads in the MSAs the nucleotide-by-locus matrix Ψ was created. Each entry $\Psi_{l,n}$ for nucleotide $n \in \mathcal{N}$ at locus $l \in \mathcal{L}$ is the proportion of reads having that nucleotide at that locus in the MSA. Here, $\mathcal{N}$ is the set of nucleotides {A,T,G,C} and $\mathcal{L}$ is the set of all loci across all SCCGs. Lastly, only those loci were considered that vary within the reference strains' SCCGs to reduce the model size and as these will be the only loci which provide information to the models. All positions with a coverage below γ_{min} were also removed.

I developed two models which estimate strains by minimizing the deviation from the observed nucleotide frequencies Ψ. The MILPs developed for this purpose (as all others presented in this thesis) were implemented in CPLEX 22.1. The underlying assumption in these models is that the frequency at which we sample each nucleotide at each locus in the core genome is approximately equal to the total relative abundances of the strains in the samples that carry that nucleotide at that locus. Here, the constraint of only considering core genes becomes important, as this assumption can be violated by the inclusion of accessory genes (Figure 3.2). A similar argument can be made for duplicate genes.

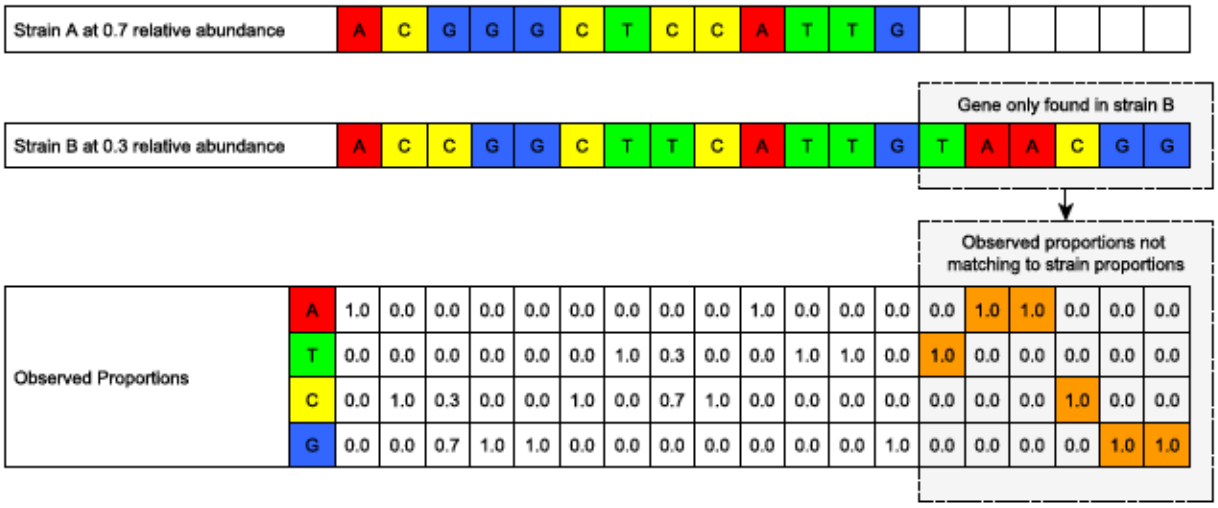


Figure 3.2: Illustration of how genes not shared between strains can skew the observed nucleotide proportions. The last six loci (here assumed to represent an accessory gene) are only present in strain B (actual nucleotides only illustrative). This leads to all of these loci being observed at a proportion of 1.0, even though strain B is only present at a relative abundance of 0.3.

Strain-Mixture Model

The first model using the nucleotide proportion matrix Ψ is the strain-mixture model. The goal of this model is to estimate a proportion P_s for each reference strain $s \in \mathcal{S}$, such that the resulting proportions of the estimated strains' nucleotides matches the observed proportions Ψ as closely as possible, while favoring solutions with fewer strains having non-zero proportions. $\mathcal{S}$ is the set of all reference strains. An additional piece of input data is $\Phi_{l,s,n}$ which is 1 if reference strain $s \in \mathcal{S}$ carries nucleotide $n \in \mathcal{N}$ at locus $l \in \mathcal{L}$ and 0 otherwise. Φ is extracted from the MSAs of the SCCGs. In the MILP formulation each P_s is a continuous variable with $P_s \in [0, 1]$. An optimal strain mixture for generating the observed nucleotide proportions can be obtained by minimizing the difference between the estimated proportions of the nucleotides according to the estimated strain proportions P_s and their nucleotides $\Phi_{l,s,n}$, and the observed nucleotide frequencies $\Phi_{l,n}$. This will generate a strain mixture whose nucleotide proportion matches the observed ones as closely as possible. Formally, this means minimizing

$$\sum_{l \in \mathcal{L}, n \in \mathcal{N}} \Omega_l \left| \Psi_{l,n} - \sum_{s \in \mathcal{S}} P_s \Phi_{l,s,n} \right|, \tag{3.2}$$

where Ω_l is the read coverage at locus l. This factor was introduced to give more weight to positions that are observed frequently, as these will likely carry more reliable information about the strain composition. Additionally, to prevent overfitting and arrive at a solution with as few strains as possible, the number of

reference strains for which $P_s > 0$ is minimized. To this end, a binary decision variable $X_s \in \{0, 1\}$ is defined and constrained via $X_s \geq P_s$ for all $s \in \mathcal{S}$. Including the sum over all X_s in the objective function will minimize the number of strains with a non-zero proportion. Formally, we minimize

$$\sum_{s \in \mathcal{S}} X_s \tag{3.3}$$

While this sparsity objective can be directly incorporated into the MILP formulation, Equation 3.2 needs to be redefined due to the non-linear absolute value in its definition. To this end, decision variables $D_{l,n} \in [0, 1]$ for $p \in \mathcal{L}, n \in \mathcal{N}$ are introduced. Then the constraints

$$D_{l,n} \geq \Psi_{l,n} - \sum_{s \in \mathcal{S}} P_s \Phi_{l,s,n}$$
$$D_{l,n} \geq \sum_{s \in \mathcal{S}} P_s \Phi_{l,s,n} - \Psi_{l,n}$$

are added to the model. The first part of the objective function is then

$$\sum_{(l,n) \in \mathcal{L} \times \mathcal{N}} \Omega_l D_{l,n}. \tag{3.4}$$

Next, the objective function can be formed. The two parts of the objective function require to be weighted in a sensible manner. To renormalize the added weight contributed by each Ω_l, the factor $\hat{\Omega}^{-1}$ is introduced, where $\hat{\Omega}$ is the average coverage across all loci. Additionally, the factor $\mathbf{L}^{-1}$ is introduced, effectively dividing the sum over $D_{l,n}$ by the number of loci in the model. Then, a hyperparameter λ_p can be introduced which weighs the importance of the above part of the objective function relative to the sparsity, weighted itself by λ_s. Introducing λ_s is not strictly necessary, as only the ratio between the weights has an impact. I include it here for consistency, as the models in Chapter IV also include a sparsity weight, but with more than one other weight, and to make the model more easily adaptable (say, by setting $\lambda_s = 0$). The final objective function is

$$\lambda_s \sum_{s \in \mathcal{S}} X_s + \frac{\lambda_p}{\mathbf{L}\hat{\Omega}} \sum_{(l,n) \in \mathcal{L} \times \mathcal{N}} \Omega_l D_{l,n}. \tag{3.5}$$

Lastly, the constraint

$$\sum_{s \in \mathcal{S}} P_s = 1. \tag{3.6}$$

will ensure that the relative abundances P_s indeed define a probability mass function. The complete formulation then is

Strain-Mixture Model

$$
\begin{aligned}
\text{minimize} \quad & \lambda_s \sum_{s \in \mathcal{S}} X_s + \frac{\lambda_p}{\mathbf{L}\hat{\Omega}} \sum_{(l,n) \in \mathcal{L} \times \mathcal{N}} \Omega_l D_{l,n} \\
\text{s.t.} \quad & X_s \in \{0,1\}, s \in \mathcal{S} \\
& P_s \in [0,1], s \in \mathcal{S} \\
& D_{l,n} \in [0,1], l \in \mathcal{L}, n \in \mathcal{N} \\
& X_s \geq P_s, s \in \mathcal{S} \\
& D_{l,n} \geq \Psi_{l,n} - \sum_{s \in \mathcal{S}} P_s \Phi_{l,s,n}, l \in \mathcal{L}, n \in \mathcal{N} \\
& D_{l,n} \geq \sum_{s \in \mathcal{S}} P_s \Phi_{l,s,n} - \Psi_{l,n}, l \in \mathcal{L}, n \in \mathcal{N} \\
& \sum_{s \in \mathcal{S}} P_s = 1
\end{aligned}
$$

Nucleotide-Mixture Model

The second model is the nucleotide-mixture model. Here, all reference strain sequences are discarded during the MILP optimization. Instead, up to $\mathbf{K}$ novel strains are estimated (corresponding to the set of estimated strains $\mathcal{K}$ with elements $\{1, 2, ..., \mathbf{K}\}$) at proportions $P_k \in [0,1]$ for $k \in \mathcal{K}$ as sequences of SNPs. The main difference between the strain-mixture and nucleotide-mixture models is how the nucleotide sequences of the strains are estimated. In the strain mixture model, reference strains are estimated and the nucleotide sequences of the estimated strains correspond exactly to the nucleotides of these reference strains. In contrast, in the nucleotide-mixture model, the nucleotide sequences of the estimated strains are determined de novo to best match the observed nucleotides. $\mathbf{K}$ is set beforehand as a parameter determining the maximum number of strains. In the model, each strain is defined as a sequence of binary variables $Y_{l,k,n} \in \{0,1\}$, where $Y_{l,k,n} = 1$ indicates that estimated strain $k \in \mathcal{K}$ carries nucleotide $n \in \mathcal{N}$ at locus $l \in \mathcal{L}$, while $Y_{l,k,n} = 0$ indicates that the strain does not carry that nucleotide at the locus. $\mathcal{L}$ is the set of the positions in the concatenated MSAs of the SCCGs, as in the strain-mixture model. As before Ψ should be matched as closely as possible by the estimated strains, while having as few strains as possible with a non-zero proportion. Hence, the objective function should contain

$$
\sum_{(l,n) \in \mathcal{L} \times \mathcal{N}} \Omega_l \left| \Psi_{l,n} - \sum_{k \in \mathcal{K}} Z_{l,k,n} \right|, \tag{3.7}
$$

again minimizing the difference between the observed and estimated nucleotide proportion. In contrast to the strain-mixture model, the nucleotides here are estimated de novo to match the observed nucleotide proportions, whereas in the strain-mixture model the nucleotides were the ones that were carried by the estimated reference strains. Here, $P_s\Phi_{l,s,n}$ from the strain-mixture model is replaced by $Z_{l,k,n}$. $Z_{l,k,n} \in [0,1]$ is a decision variable encoding the relative proportion of estimated strain k at locus l if the estimated strain k carries nucleotide n at that locus. In order to make $Z_{l,k,n}$ fulfill this requirement a few new constraints are required. First, $Z_{l,k,n} = P_k$ must be satisfied, i.e. $Z_{l,k,n}$ needs to represent the estimated proportion of strain k. This will be achieved by adding the constraints

$$\sum_{n \in \mathcal{N}} Z_{l,k,n} = P_k \tag{3.8}$$

for all $k \in \mathcal{K}, l \in \mathcal{L}$. Additionally, $Z_{l,k,n} > 0$ should only be possible if the estimated strain k carries nucleotide n at locus l which is enforced through

$$Y_{l,k,n} \geq Z_{l,k,n} \tag{3.9}$$

for all $l \in \mathcal{L}, k \in \mathcal{K}, n \in \mathcal{N}$. Now, all that is left is to ensure that each estimated strain carries exactly one nucleotide at each position, which is achieved through the constraint

$$\sum_{n \in \mathcal{N}} Y_{l,k,n} = 1 \tag{3.10}$$

for all $l \in \mathcal{L}, k \in \mathcal{K}$. Then Equation 3.7 can be expressed as in the strain-mixture model via

$$D_{l,n} \geq \Psi_{l,n} - \sum_{k \in \mathcal{K}} Z_{l,k,n}$$
$$D_{l,n} \geq \sum_{k \in \mathcal{K}} Z_{l,k,n} - \Psi_{l,n}$$

for all $l \in \mathcal{L}, n \in \mathcal{N}$ and we can add

$$\sum_{(l,n) \in \mathcal{L} \times \mathcal{N}} \Omega_l D_{l,n} \tag{3.11}$$

as well as the sparsity objective

$$\sum_{k \in \mathcal{K}} X_k. \tag{3.12}$$

to the objective function. Here, the X_k are constrained through

$$P_k \geq X_k \tag{3.13}$$

for $k \in \mathcal{K}$. The objective function, weighted as in the strain-mixture model, then is

$$\lambda_s \sum_{k \in \mathcal{S}} X_k + \frac{\lambda_p}{\mathbf{L}\hat{\Omega}} \sum_{(l,n) \in \mathcal{L} \times \mathcal{N}} \Omega_l D_{l,n}. \tag{3.14}$$

Lastly, the constraints $\sum_{k \in \mathcal{K}} P_k = 1$ are required again. Additionally,

$$P_{k_1} \geq P_{k_2} \tag{3.15}$$

is introduced for all $k_1 \in \mathcal{K}, k_2 \in \mathcal{K}, k_1 > k_2$. These constraints break symmetries in the MILP, introduced by the fact that the order of estimated strains $k \in \mathcal{K}$ would be arbitrary otherwise. Including them likely makes the MILP easier to solve by excluding equivalent permutations of the decision variables' values [196]. The final formulation takes the form

Nucleotide-Mixture Model

$$
\begin{aligned}
\text{minimize} \quad & \lambda_s \sum_{k \in \mathcal{S}} X_k + \frac{\lambda}{\mathbf{L}\hat{\Omega}} \sum_{(l,n) \in \mathcal{L} \times \mathcal{N}} \Omega_l D_{l,n} \\
\text{s.t.} \quad & X_k \in \{0,1\}, k \in \mathcal{S} \\
& P_k \in [0,1], k \in \mathcal{S} \\
& Y_{l,k,n} \in \{0,1\}, l \in \mathcal{L}, k \in \mathcal{K}, n \in \mathcal{N} \\
& Z_{l,k,n} \in [0,1], l \in \mathcal{L}, k \in \mathcal{K}, n \in \mathcal{N} \\
& D_{l,n} \in [0,1], l \in \mathcal{L}, n \in \mathcal{N} \\
& X_k \geq P_k, k \in \mathcal{S} \\
& \sum_{n \in \mathcal{N}} Y_{l,k,n} = 1, l \in \mathcal{L}, k \in \mathcal{K} \\
& Y_{l,k,n} \geq Z_{l,k,n}, l \in \mathcal{L}, k \in \mathcal{K}, n \in \mathcal{N} \\
& \sum_{n \in \mathcal{N}} Z_{l,k,n} = P_k, l \in \mathcal{L}, k \in \mathcal{K} \\
& D_{l,n} \geq \Psi_{l,n} - \sum_{k \in \mathcal{K}} Z_{l,k,n}, l \in \mathcal{L}, n \in \mathcal{N} \\
& D_{l,n} \geq \sum_{k \in \mathcal{K}} Z_{l,k,n} - \Psi_{l,n}, l \in \mathcal{L}, n \in \mathcal{N} \\
& \sum_{k \in \mathcal{S}} P_k = 1 \\
& P_{k_1} \geq P_{k_2}, k_1 \in \mathcal{K}, k_2 \in \mathcal{K}, k_1 > k_2
\end{aligned}
$$

Strain-Allele Model

Next, I describe the models working on a read instead of a nucleotide basis. The first of the two read based models is the strain-allele model. Firstly, we need to define the input variables available to us. As before the reads are aligned against the reference strains using Bowtie2 and added to the MSAs of the different core genes. From these MSAs, the number of mismatches between each read and each core gene allele (variant carried by each of the strains) is extracted, generating the matrix Δ with entry $\Delta_{r,s}$ being the number of nucleotide mismatches between read $r \in \mathcal{R}$ and reference strain $s \in \mathcal{S}$ (or more specifically, the gene variant carried by strain s). I also explored an alternative approach where, after assignment of a read to a core gene using Bowtie2, each read is aligned separately to each strains core gene variant, using pairwise local alignment with the following scores:

- Match score: 1
- Mismatch score: 0
- Gap opening score: -500
- Gap extension score: -500
- Target gap end score: 0
- Query gap end score: 0

The large negative gap-related scores were chosen to prevent the introduction of gaps, under the assumption that due to the unlikeliness of a gap being introduced by a sequencing error, it is much more likely that the read did not originate in the given strain. Hence, we can safely disregard this potential read-strain assignment (which was aimed for in this model). Similarly, read-strain pairs were excluded for which the mismatch between read and strain exceeded a parameter δ_{max}. The determination of this mismatch was done following the alignment. Genes for which fewer than γ_{min} reads are observed are also excluded.

The main MILP then aims to again estimate strain proportions and genetic make-up. As opposed to the previous models, this is now done on a gene-by-gene basis and instead of estimating a proportion for every reference strain, up to $\mathbf{K}$ strains are estimated as a combination of core genes, where $\mathbf{K}$ is a parameter set a priori. A further aim is to assign each read to a strain, which will be useful for downstream analysis. We can rephrase these aims into decision variables we aim to estimate. Firstly, for each of the $\mathbf{K}$ estimated strains we can define a proportion $P_k \in [0, 1]$. Secondly, let us define binary variables $L_{s,k,g}$ which should take on the value of 1 if the estimated strain $k \in \mathcal{K}$ has the gene variant of reference strain $s \in \mathcal{S}$ for gene $g \in \mathcal{G}$, with $\mathcal{G}$ being the set of SCCGs. Lastly, we can encode read-strain assignments in the binary variables $M_{r,s}$, assigning it 1 if read $r \in \mathcal{R}$ is assigned to strain $s \in \mathcal{S}$. To estimate these variables' values three characteristics are then minimized. Firstly, the read-strain mismatch is minimized, i.e.

$$\frac{1}{\mathbf{R}\delta_{max}} \sum_{r \in \mathcal{R}, s \in \mathcal{S}} M_{r,s}\Delta_{r,s}, \tag{3.16}$$

weighted by λ_m. Intuitively, this quantity corresponds to the average mismatch between reads and strain sequences based on the assignments of reads to strains via $M_{r,s}$ and the corresponding mismatch $\Delta_{r,s}$. Minimizing this quantity will result in those $M_{r,s}$ being assigned a value of 1 that will lead to the least total mismatch, as only those entries of Δ contribute to the sum for which $M_{r,s} = 1$ Secondly, a sequence of gene variants for each estimated strain should be established by the model, where a gene variant is simply defined

to be a unique sequence of nucleotides for the gene. A natural assumption is that for any estimated strain there should be a reference strain that it resembles to some degree. Specifically, we can aim for a minimum difference in gene variants from at least one reference strain. This can be formalized as finding the minimum distance between the estimated strain and the reference strains across all reference strains. The first step in integrating this demand into the MILP is to define the decision variables z_k, as well as $y_{s,k}$ for $s \in \mathcal{S}$ and $k \in \mathcal{K}$ and to set up the constraints

$$\begin{aligned} z_k &\geq \left(1 - \frac{1}{\mathbf{G}} \sum_{g \in \mathcal{G}} L_{1,k,g}\right) - y_{1,k} \\ z_k &\geq \left(1 - \frac{1}{\mathbf{G}} \sum_{g \in \mathcal{G}} L_{2,k,g}\right) - y_{2,k} \\ &\vdots \\ z_k &\geq \left(1 - \frac{1}{\mathbf{G}} \sum_{g \in \mathcal{G}} L_{\mathbf{S},k,g}\right) - y_{\mathbf{S},k} \\ \sum_{s=1}^{\mathbf{S}} y_{s,k} &= \mathbf{S} - 1, \end{aligned}$$

for all $k \in \mathcal{K}$. Here, $1 - \frac{1}{\mathbf{G}} \sum_{g \in \mathcal{G}} L_{s,k,g}$ is the percent of disagreement in gene variants between estimated strain k and reference strain s, i.e. a measure of how many of the alleles carried by the estimated strain are also carried by the reference strain. Note that due to the last constraint exactly one $y_{s,k} = 0$, while all others equal 1. $y_{s,k}$ serves as strain selecting variable, constraining z_k to be bigger or equal to $1 - \frac{1}{\mathbf{G}} \sum_{g \in \mathcal{G}} L_{2,k,g}$ for exactly the one strain s for which $y_{s,k} = 0$. For all other strains s, $y_{s,k} = 0$ and as $1 - \frac{1}{\mathbf{G}} \sum_{g \in \mathcal{G}} L_{2,k,g} \leq 1$ subtracting $y_{s,k} = 1$ leads to a constraint for z_k to be bigger or equal to at most 0, effectively not constraining the variables z_k. By adding

$$\frac{1}{\mathbf{K}} \sum_{k \in \mathcal{K}} z_k, \tag{3.17}$$

weighted by λ_d, to the objective function this will constrain exactly the $y_{s,k}$ to 0 for which the difference of the estimated strain k to the reference strain is minimal. Thus, we obtain the effective constraint

$$z_k \geq 1 - \frac{1}{\mathbf{G}} \sum_{g \in \mathcal{G}} L_{s,k,g}. \tag{3.18}$$

This will enable the estimation of strains closely related to the reference strains.

Next, assume that the relative coverage of any gene by any strain is proportional to that strain's relative abundance. We can use this to estimate strain proportions. We aim to minimize the deviation between the estimated proportions of reads in a gene assigned to a particular allele and the total proportion of the

estimated strains carrying that allele. We obtain the proportion of reads assigned to the gene variant of strain s for a gene g as $\frac{1}{|\mathcal{R}_g|}\sum_{r\in\mathcal{R}_g} M_{r,s}$, where $|\mathcal{R}_g|$ is the cardinality of the set of reads $\mathcal{R}_g \subseteq \mathcal{R}$ assigned to gene g. To obtain the total proportion of the estimated strains carrying that allele we calculate $\sum_{k\in\mathcal{K}} Y_{s,k,g}$. $Y_{s,k,g}$ is the relative abundance of estimated strain k if it carries the gene variant of reference strain s for gene g and 0 otherwise. We can then take the absolute deviation between these two quantities and minimize the total deviation across all genes $g \in \mathcal{G}$ and strains $s \in \mathcal{S}$

$$\frac{1}{\mathbf{GS}}\sum_{g\in\mathcal{G}}\sum_{s\in\mathcal{S}}\left|\frac{1}{|\mathcal{R}_g|}\sum_{r\in\mathcal{R}_g} M_{r,s} - \sum_{k\in\mathcal{K}} Y_{s,k,g}\right|, \tag{3.19}$$

This part of the objective function is weighted by λ_p. Importantly, $Y_{s,k,g}$ still needs to be constrained within the MILP. Specifically, it is desired that $Y_{s,k,g} = L_{s,k,g}P_k$. However, this is not linear. The linearization is achieved by the constraints

$$\begin{aligned} Y_{s,k,g} &\leq P_k \\ Y_{s,k,g} &\leq L_{s,k,g} \\ Y_{s,k,g} &\leq P_k + L_{s,k,g} - 1. \end{aligned}$$

Furthermore, we again need to linearize the absolute value by introducing the continuous decision variable $D_{g,s} \in [0,1]$ and constraining it by

$$\begin{aligned} D_{g,s} &\geq \frac{1}{|\mathcal{R}_g|}\sum_{r\in\mathcal{R}_g} M_{r,s} - \sum_{k\in\mathcal{K}} Y_{s,k,g} \\ D_{g,s} &\geq \sum_{k\in\mathcal{K}} Y_{s,k,g} - \frac{1}{|\mathcal{R}_g|}\sum_{r\in\mathcal{R}_g} M_{r,s} \end{aligned}$$

and then adding

$$\frac{1}{\mathbf{GS}}\sum_{g\in\mathcal{G}}\sum_{s\in\mathcal{S}} D_{g,s} \tag{3.20}$$

to the objective function.

Lastly, several constraints need to be included that guarantee that sensible solutions are obtained. Firstly, introduce

$$\sum_{k\in\mathcal{K}} P_k = 1, \tag{3.21}$$

as before to guarantee that the proportions sum to 1, as well as

$$\sum_{s \in \mathcal{S}} M_{r,s} = 1, \tag{3.22}$$

for each read, ensuring that each read is mapped uniquely to one gene variant. Similarly, including

$$\sum_{s \in \mathcal{S}} L_{s,k,g} = 1 \tag{3.23}$$

for each estimated strain k and gene g ensures that only one gene variant is assigned for each gene and estimated strain. Lastly, we can introduce constraints that break symmetries, i.e.

$$P_k \leq P_{k+1}, \tag{3.24}$$

for $1 \leq k \leq \mathbf{K} - 1$, ensuring that the strain proportions are weakly increasing. The complete model then is

Strain-Allele Model

$$\text{minimize} \quad \frac{\lambda_m}{\mathbf{R}\delta_{max}} \sum_{r\in\mathcal{R},s\in\mathcal{S}} M_{r,s}\Delta_{r,s} + \frac{\lambda_d}{\mathbf{K}} \sum_{k\in\mathcal{K}} z_k + \frac{\lambda_p}{\mathbf{GS}} \sum_{g\in\mathcal{G},s\in\mathcal{S}} \left| \frac{1}{|\mathcal{R}_g|} \sum_{r\in\mathcal{R}_g} M_{r,s} - \sum_{k\in\mathcal{K}} Y_{s,k,g} \right|$$

$$\text{s.t.} \quad P_k \in [0,1], k \in \mathcal{K}$$

$$Y_{s,k,g} \in [0,1], s \in \mathcal{S}, k \in \mathcal{K}, g \in \mathcal{G}$$

$$y_{s,k} \in \{0,1\}, k \in \mathcal{K}, s \in \mathcal{S}$$

$$z_k \in [0,1], k \in \mathcal{K}$$

$$L_{s,k,g} \in \{0,1\}, s \in \mathcal{S}, k \in \mathcal{K}, g \in \mathcal{G}$$

$$M_{r,s} \in \{0,1\}, r \in \mathcal{R}, s \in \mathcal{S}$$

$$D_{g,s} \in [0,1], g \in \mathcal{G}, s \in \mathcal{S}$$

$$z_k \geq \left(1 - \frac{1}{\mathbf{G}} \sum_{g\in\mathcal{G}} L_{s,k,g}\right), k \in \mathcal{K}, 0 \leq s \leq \mathbf{S}$$

$$\sum_{s=1}^{\mathbf{S}} y_{s,k} = \mathbf{S} - 1$$

$$Y_{s,k,g} \leq P_k, k \in \mathcal{K}, g \in \mathcal{G}, s \in \mathcal{S}$$

$$Y_{s,k,g} \leq L_{s,k,g}, k \in \mathcal{K}, g \in \mathcal{G}, s \in \mathcal{S}$$

$$Y_{s,k,g} \leq P_k + L_{s,k,g} - 1, k \in \mathcal{K}, g \in \mathcal{G}, s \in \mathcal{S}$$

$$D_{g,s} \geq \frac{1}{|\mathcal{R}_g|} \sum_{r\in\mathcal{R}_g} M_{r,s} - \sum_{k\in\mathcal{K}} Y_{s,k,g}, g \in \mathcal{G}, s \in \mathcal{S}$$

$$D_{g,s} \geq \sum_{k\in\mathcal{K}} Y_{s,k,g} - \frac{1}{|\mathcal{R}_g|} \sum_{r\in\mathcal{R}_g} M_{r,s}, g \in \mathcal{G}, s \in \mathcal{S}$$

$$\sum_{k\in\mathcal{K}} P_k = 1$$

$$\sum_{s\in\mathcal{S}} M_{r,s} = 1, r \in \mathcal{R}$$

$$\sum_{s\in\mathcal{S}} L_{s,k,g} = 1, k \in \mathcal{K}, g \in \mathcal{G}$$

$$P_k \leq P_{k+1}, 1 \leq k \leq \mathbf{K} - 1$$

Computational limits and approaches for solving them

While the model is complete, a further step can be taken in its implementation to potentially reduce the model complexity. Gene variants are not always unique for each strain. Hence, gene variants that are the

same can be identified and grouped. Then, the mismatch can be determined relative to a gene variant instead of a strain. Reads will therefore be assigned to gene variants. By simple reverse indexing we can still implement the optimization for the minimum distance to any strain, while technically estimating a gene variant sequence that has a smaller variant database.

Even with this improvement the strain allele models' central limitation is its runtime. Many of the strain-allele models did not converge within 72 hours of allotted runtime (see Results). For this reason, I pursued multiple strategies to reduce the computational load by reducing the problem size. Most of these approaches suffered from poor performance, either in accuracy or in runtime, and are hence only briefly mentioned. The model variant that was included in the final comparison, the cluster-allele model, is laid out in the next section. Besides this, I implemented the following variants.

One natural approach to reduce the problem size is to reduce the input size. I tried selecting a random subsample of reads. Further, a model with a higher minimum per-gene coverage was implemented where any gene was excluded that had fewer than half of the median coverage of all genes. Both approaches, tested separately or combined, did not result in high accuracy when reducing the input size sufficiently to enable timely convergence. One approach that was implemented that did not result in high accuracy was to randomly subsample a small set of reads many times over (with the number of reads between 10 and 100, to guarantee low runtime) and to solve each model independently. Following this step, a final model would be run, only keeping the 10 or 100 strains that appeared most often in a solution to the subsampled problems. However, one can clearly see that this might not work for low proportion strains, as any subsample of reads that is large enough will be dominated by the highest proportion strain. As this would suggest, the model did not show high accuracy. Lastly, reducing the set of genes used was also considered. Here, I identified a minimal (not necessarily unique) set of genes that can discriminate between all reference strains, as well as a doubly identifying set, i.e. one where the difference between any two strains is at least two gene variants. Neither of these approaches showed promising results, nor did replacing the core genes by MLST loci.

Cluster-Allele Model

The last approach that was implemented to reduce the problem size was a reduction of the number of reference strains via clustering. In this model, the cluster-allele model, the strains in the reference database were first clustered. For this, I applied rHierBAPS to the set of core gene MSAs, resulting in 10 clusters at the highest level. I alternatively considered k-modes [197], a variant of k-means for clustering categorical data. In k-modes the underlying difference between two data points are the Hamming Distance between them.

For the present data, all core genes were concatenated. Based on this, the Hamming Distances between the resulting strings was calculated. K-modes then proceeds by assigning each strain to one of a fixed number of clusters. In each cluster the consensus sequence is determined (i.e. the most common nucleotide at each position; the mode) in the concatenated gene sequences. Then each strain is assigned to the cluster with the closest consensus sequence. This process is repeated until the resulting clusters are stable. K-modes has the potential advantage of specifying the number of clusters, adjusting the number of clusters according to the computational resources available. At the same time however, the fixed number of clusters does not necessarily represent a sensible clustering of the data, in contrast to rHierBAPS. The incorporation of phylogenetic factors in rHierBAPS makes it a good choice, as the resulting clusters will be more easily interpretable from a biological standpoint. If the cluster-allele model performs well, the resulting clusters could then also be independently analyzed, given that read-strain assignments are generated, to arrive at a higher taxonomic resolution. Complete hierarchical binary clustering using k-modes was also considered. However, this model did not return reliable results and further would require 10 consecutive model runs to arrive at a strain level resolution. Therefore, the focus was on the approach using the rHierBAPS clustering with the plan to optimize the cluster-allele model fully if this initial cluster model setup shows promising performance compared to the other three model variants. To implement the cluster-based approach, read-strain mismatches are first determined as for the strain-allele model. The model then uses the minimum mismatch between a read and all the strains in a cluster as the cluster's mismatch. The strains in the strain-allele model formulation are then replaced by the clusters. To still be able to calculate the minimum distance to any reference strain I created a map identifying each gene variant with the reference strains carrying it. Besides this, the MILP formulation remains the same and is hence not repeated here.

Data Simulation

To tune the four models' parameter settings I simulated three metagenomic sequencing datasets, one containing only one reference strain, one containing two (at relative abundances of 70% and 30%), and one containing three (at relative abundances of 50%, 30%, and 20%). Each dataset had a 10-fold total coverage (coverage across all strains) assuming a genome length of four million basepairs (bp). To enable tuning of the cluster-allele model I ensured that each of the strains in a sample is in a different cluster as estimated by RHierBAPS. Clusters were chosen at random and the strains were sampled randomly from the determined clusters. Additionally, to evaluate the performance of the best performing model variants, I created a test-set of 25 simulated datasets, again at a 10-fold total coverage, with 5 datasets each for 1, 2, 3, 4, and 5 constituent strains, respectively. Strains were chosen at random and relative abundances were drawn

from a beta distribution $\beta(5,5)$ and normalized. This means that opposed to the initial three datasets, the test-set covers a wider range of numbers of strains as well as different relative abundance distributions of these strains. To simulate each dataset, a random contig is first selected from the sampled reference strain, where the sampling probability of each contig corresponds to the fraction of the genome length a contig's length constitutes. Based on the selected contigs, start position for a read of length 150bp within that contig is selected uniformly at random. To generate the reverse reads the fragment length was sampled from a normal distribution (Mean = 400, SD = 20). This process was repeated for each reference strain until its' desired coverage was reached (total coverage times the relative abundance of the reference strain). To assure realistic simulation, sequencing errors were generated based on the HiSeq2500L150 quality profiles from ART[37]. This approach was chosen, as opposed to using ART directly, as it enabled improved analyses of outcomes and results during the development phase. For the models incorporating the accessory genome (described in Chapter IV) I reverted to using ART to obtain representative results.

Metrics

To evaluate the models' accuracy in the identification of strain clusters and relative abundances the cluster-based F1-scores and JSDs were calculated, respectively. The estimated RHierBAPS clusters and their proportions were obtained either directly from the models or by determining the cluster of origin of the estimated strains (strain-mixture and strain-allele model) or the reference strains with the lowest SCCG hamming distance to the estimated strains (nucleotide-mixture model) and those strains' clusters. To obtain the cluster-wise proportions, the proportions of all strains from the same clusters are summed. F1-scores and JSDs were then calculated as in Chapter II. Secondly, the 1-dimensional Wasserstein distance[184,185] (WSD) was used with a slight modification to Chapter II. As the interest here is mainly in the core genome, I replaced the mash distance by the hamming distance $H(x,y)$ between the core gene sequences x and y. For whole genomes, computing a hamming distance is not feasible, however as only the SCCG sequences for which we have MSAs are compared here, computing the hamming distance is feasible and gives an exact measure of the number of differing nucleotide positions.

$$WSD(v,w) = \frac{1}{L} \inf_{\pi} \left(\frac{1}{m} \sum_{i=1}^{m} H(v_i, w_{\pi(i)}) \right). \tag{3.25}$$

Here, w was obtained from the estimated proportions and estimated reference strains (or the closest sequences of the estimated strains for the nucleotide-mixture model) returned by the models. As the cluster-allele model does not return reference strains no WSD was computed for it.

Parameter	Nucleotide-based Models	Allele-based Models
Model Type	Strain-mixture, Nucleotide-mixture	Strain-Allele, Cluster-Allele
γ_{min} (minimum coverage)	1, 2, 5	1, 10
λ_p (weight of Equations 3.2, 3.7, and 3.19)	0.1, 1, 10	1, 10
λ_m (weight of Equation 3.16)	N/A	1, 10, 100
λ_d (weight of Equation 3.17)	N/A	1, 10
δ_{max}	N/A	1,3
Mismatch Determination	N/A	From MSA, From individual alignment
Minimum z_k	N/A	1,0.9,0.5

Table 3.1: Parameter settings used for model tuning. All different combinations of the listed parameter settings were used. $\lambda_s = 1$ (weight of the sparsity objective in Equations 3.3 and 3.12) and $\mathbf{K} = 3$ (maximum number of inferred strains) were used across models. Minimum z_k indicates that $z_k \geq 1$, $z_k \geq 0.9$, or $z_k \geq 0.5$ is added as a constraint. Hence, the inferred strains' allelic profile must match at reference strain to that percentage. As reminder, λ_p weighs the importance of minimizing the difference between observed and estimated nucleotide or read proportions (depending on whether a nucleotide- or allele-based model is utilized), λ_m weighs the total mismatch between reads and their assigned gene variant, and λ_d weighs the importance of the set of gene variants of the estimated strains being close to at least one reference strain. Finally, mismatch determination refers to the alignment process used prior to determination of the number of mismatches between reads and strain variants, with reads either being added to the SCCGs MSAs, or reads being aligned individually to each gene variant via pairwise alignment.

Results

Each model was tested on the tuning datasets with each combination of the parameter settings listed in Table 3.1. 24 out of 36 nucleotide-based and 591 out of 1152 allele-based model variants did not complete for a 5% gap and 72 hours of runtime. The corresponding allele-based models were excluded from further analysis. The 24 nucleotide-based models that did not close the gap all corresponded to the nucleotide-mixture model running on samples containing more than one strain. However, solutions with a gap higher than 5% were generated and these are shown. The results of the tuning are shown in Figure 3.3. For evaluation of model performance, each models' performance under each set of parameters was averaged across the three tuning datasets. The strain-mixture model performs consistently across parameter settings and outperforms the nucleotide-mixture and cluster-allele models. The strain-allele model has a wide spread of performances, with the model under some parameter settings outperforming the strain-mixture model in both F1-scores and JSDs (Figure 3.3).

Based on these results, I further evaluated the strain-mixture models as well as the four variants of the strain-allele model with an F1-score above 0.90 (now with $\mathbf{K} = 5$) on the test-data. Here, the strain-allele models perform worse (WSD-mean = 0.026, F1-mean = 0.53, JSD-mean = 0.29) compared to the strain-mixture models (WSD-mean = 0.0020, F1-mean = 0.86, JSD-mean = 0.084). Most of the strain-mixture model variants models show a very similar performance (Figure 3.4A). For the best performing model variant

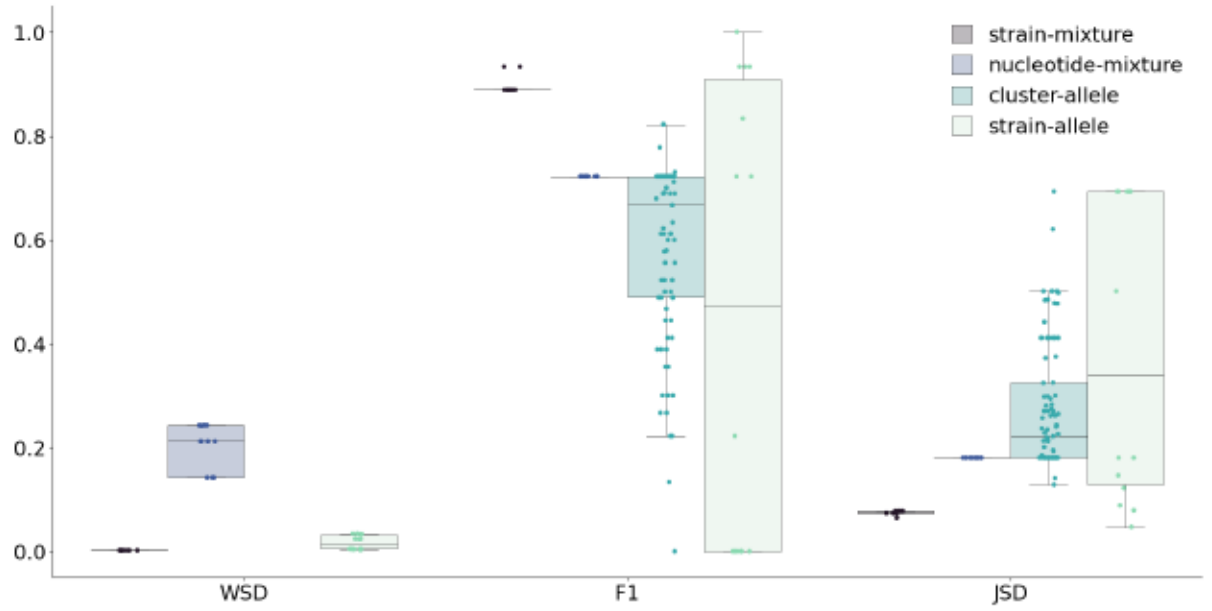


Figure 3.3: Performance metrics of all model variants. Each model variants' performance is averaged across samples with 1, 2, and 3 strains and displayed by a dot. Boxplots show model performances grouped by their model type (strain-mixture, nucleotide-mixture, cluster-allele, and strain-allele model; cluster-allele model not shown for WSD as no assembly level inference is provided). Lines show median and whiskers show 1.5 times the interquartile range.

according to WSD ($\lambda_p = 10$, $\gamma_{min} = 1$) a comparison of the WSDs to the expected WSDs was performed. The expected WSDs was obtained by simulating 1000 random population pairs for each combination of one to five strains in either population, similar as was done in Chapter II with the difference of the WSD being computed via the hamming distances of the SCCGs. Specifically, between 1 and 5 assemblies were selected to be part of either population, with equal relative abundances, and the WSD between the two populations was computed using the hamming distance between the constituent strains SCCG sequences. This process was repeated to obtain the distribution of expected WSDs. Here, the model shows significantly better performance than expected by chance (Figure 3.4B; two-sided t-test, $t = 3.39, p < 0.001$, $WSD_{estimated} = 0.0019$, $SD_{estimated} = 0.0016$, $WSD_{expected} = 0.004$, $SD_{expected} = 0.003$). The ANI (average nucleotide identity, a measure of whole genome similarity) between the estimated and the true strains in the samples is also of interest, both for the SCCGs and for the whole genome. Hence, these ANIs were computed for the best performing model. Figure 3.4C shows a comparison of the ANI between each estimated strain and the true strain with the highest SCCG ANI to that estimated strain. For the SCCG ANI I simply counted all differing positions in the SCCG MSAs, i.e. computed the hamming distance. Whole genome ANI between all pairs of reference strains was estimated using fastANI[198] with default settings. For whole genome ANI two clusters of results can be observed. One of these, with the majority of 74% of estimated strains, still

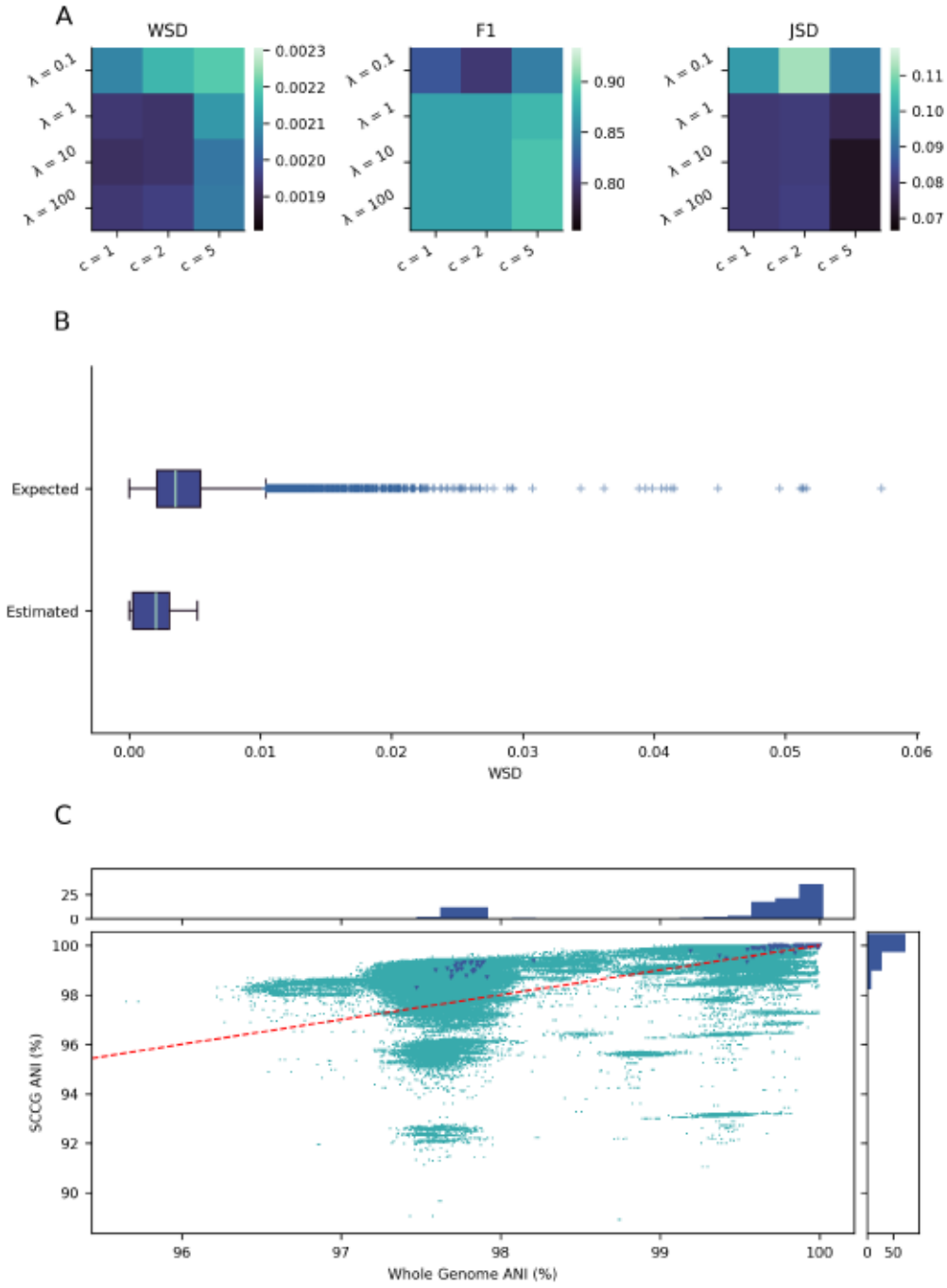


Figure 3.4: Figure presents results on the 25 test datasets for the strain mixture model. **(A)** Average performance metrics of strain-mixture models with different parameter settings on the test dataset. $\lambda = \lambda_p$ and $c = \gamma_{min}$. **(B)** Expected WSD from random populations with 1 to 5 strains each and estimated WSD obtained from the best performing strain mixture model ($\lambda_p = 10, \gamma_{min} = 1$). **(C)** ANI between estimated and true strains across all test samples. Turquoise squares show all-to-all pairwise whole genome and SCCG ANI for all reference strains. Blue triangles indicate ANIs between each estimated strain and the true strain with the highest SCCG ANI to that estimated strain. Red dashed line indicates $x = y$. Histograms show the marginal distributions of the whole genome and SCCG ANIs of the estimated strains.

approximates the whole genome well, displaying a whole genome ANI above 99%. Additionally, SCCG ANI again shows good identification of strains with 58% of estimated strains having an SCCG ANI above 99.9%, and 31% above 99.99%. This analysis further serves to show that even among strains closely related in the SCCGs, whole genome variability can be big (Figure 3.4C).

Discussion

The strain-mixture model shows consistently good performance in the identification of strain clusters of *A. baumannii* based on the species' SCCGs. This approach clearly outperforms the other three models. When considering models that aim to estimate the gene composition from scratch using reads, i.e. the strain-allele and cluster-allele model, the high computational resource requirements are the main hurdle. The need to include all gene variants and strain combinations as variables in the model, with the additional overhead of minimizing the distance to the reference strains, clearly exceeds reasonable computational resources and is not practical. Overall the performance of the nucleotide-mixture model is limited regarding strain identification. This may be due to the fact that the gap could not be closed, and hence, no optimal solutions could be found, or due to the data being too noisy to enable accurate estimation without the aid of a reference database. I also derived a model which incorporates novel estimation of nucleotides but minimizes the distance to at least one reference strains. However, learning from the experiments similar issues as for the strain- and cluster-allele models will be encountered, namely the large number of variables required to encode the minimum distance to the reference strains. Specifically for every locus we will need to introduce at least as many variables as there are reference strains, resulting in over $\mathbf{LS}$ additional variables. In contrast, the largest contribution of any one set of decision variables in the nucleotide-mixture model is $\mathbf{LKN}$, with $\mathbf{N} = 4$ and $\mathbf{K} << \mathbf{S}$. Hence, this model is not laid out in full here and not tested in practice.

Closer investigation of the strain-mixture model showed robustness against many parameter changes. Furthermore, the WSDs of the estimated strain mixtures are significantly better than expected by chance. Additionally, the estimates are consistently high in core genome similarity and, even considering whole genome ANI, very closely related strains can often be identified. Nonetheless, a group of identifications with lower whole genome ANI is observed. This is unsurprising given that only with a small proportion of the whole genome was incorporated in the model. From the present experiments, it is hard to infer how the relatedness between strains in a sample impacts inference, as all strains in a sample were chosen from different clusters.

While an approach wholly based on core genes may not be as accurate as possible, it may still be valuable

in practical applications. First of all, we have seen in Chapter II that even tools such as StrainScan or StrainGE may not always deliver precise enough results at higher resolutions. Moreover, a frequent strategy for constructing phylogenies, and one that can sometimes be more appropriate due to core genes being less affected by recombination events, is to base them on core genes[199]. However, this strategy has been under debate[199]. Nonetheless, core genome phylogenies have been shown to be robust to relatively high recombination rates in bacterial species[199]. In a similar vein one may want to limit the metagenomic analysis to a set of genes that makes inferences comparable and interpretable. The strain-mixture model's flexibility here is highlighted by the possibility to use any gene database. While an automated approach was aimed for, researchers could instead use MLST genes for inference of MLST strain types or curated core genome MLST schemes (an extension of MLST based on loci in core genes), simply by replacing the database strain names by their strain type and the SCCG MSAs by MSAs of the MLST loci. Note here that MLST loci may be duplicated in the genome of some strains, and future research should address this. This adaptability to different contexts and research aims that demand different information from a metagenomic analysis makes the inferences of the strain-mixture model more readily implementable and interpretable.

Overall, the presented results are promising and point to the applicability of the strain-mixture model. However, accuracy still can be improved and no information about the accessory genome is obtained. The latter will be of interest in many cases. Hence, development of the strain-mixture model was continued by including the accessory genome, described in the next chapter. A more thorough discussion of the contributions and limitations of the present work, as well as promising avenues for future work and applications are deferred to Chapter IV in an effort to avoid repetition.

Chapter IV

Whole Genome based Strain Identification in Metagenomes

'The more we learn about the world, and the deeper our learning, the more conscious, specific, and articulate will be our knowledge of what we do not know, our knowledge of our ignorance.'

– Karl Popper

The models presented in Chapter III only consider the core genome. However, gaining information on the accessory genome is essential, both for achieving competitive accuracy in strain level identification and for a quantification of the content of the accessory genome of strains found in metagenomes. The latter will greatly improve interpretability of the strain mixture estimations. While all of the tools in Chapter II consider the whole genome, none of them enable identification of genes or coding sequences that are present in the metagenomic strains, but potentially not in the identified reference strains (though StrainGE can identify large deletions). The ability to identifying acquired genes is essential in determining the spread of these genes and in tracking the evolution of a species across time and sample points. Additionally, k-mer based approaches (such as StrainScan, StrainGE, or KrakenUniq) consider the whole genome but are potentially hard to interpret. Here, I present two models that extend the strain-mixture model shown in Chapter III, taking into account coding sequences found in the accessory genome. As in the previous chapter, the term gene refers only to coding sequences in this chapter. The first model takes the accessory genome into account during estimation of the reference strains in a sample and the other additionally has the ability to assign genes that are known to be present in at least one reference strain to strains identified in a metagenomic sample, even if the identified reference strain does not contain the given gene. I developed and evaluated the models on a set of simulated metagenomic samples of *A. baumannii*. Additionally, I tested the final model

variant on the same datasets as the other tools evaluated in Chapter II, both for different species and complex strain mocks. Finally, I assess the ability of the model to recover coding sequences.

Methods

The main issue in conceptualizing a model that incorporates genes that are not present in 100% of reference strains is that the relative abundances of these genes may be skewed. Take for example a sample that contains two strains only one of which carries a certain gene. Then, if we were to estimate two strains the true solution could only explain that proportion of the gene equivalent to the proportion of the strain carrying it. For example, if two strains were present at 70% and 30%, respectively, we would estimate any nucleotide in a gene carried only by the second strain at a proportion of only 0.3 in the accessory gene even though the observed proportion is 1.0 (as only the one strain carries the gene; Figure 3.2). Hence, we could incur significant errors under the previous formulation. There are multiple approaches to deal with this in order to incorporate the accessory genome after all. First of all, we can simply ignore the above consideration and apply the strain-mixture model to genes not present in 100% of reference strains. If we choose not to include all genes, but only those present in more than X% of the reference strains, where X is large, say above 95%, then in most cases our approach could still work. However, this will greatly increase the computational demands and including all of the genome is unlikely to be successful. Furthermore, we would still have to contend with potential inaccuracies of the nucleotide proportions. Other approaches can be taken that promise higher accuracy while not increasing the problem size as much. These approaches work on a gene-by-gene basis instead of on a nucleotide basis. Importantly, the nucleotide-based approach for the core-genes is still a central part of the models. While promising greater accuracy (as seen in comparison to the allele-based models) keeping the nucleotide-based approach for the core genome further enables the identification of SNPs and is beneficial in inferring phylogenies down the line when these are based on the assumption of neutral evolution, more likely to be found in the core genome of a species as these genes are likely under stabilizing selection due to their apparent necessity in all members of a species. Indeed, research suggests that core genes are regularly expressed and conserved[188], which is not necessarily true for all genes. Including the accessory genome separately will enable a more fine-grained analysis of the properties presented by the identified strains and the ancestry of different parts of their genomes. I again introduce two new models, at different levels of complexity. For ease of reference, I call these the accessory-presence model and the accessory-proportion model. While extending to non-core genes, I still only considered single-copy genes to preserve the accuracy of any abundance information obtained from the metagenomic data. In this chapter,

I start out by describing the assemblies for *A. baumannii* used in database creation, which is the same as in Chapter II. As in Chapter III, *A. baumannii* was used as example species during model development. However, in this chapter, I also evaluate the model on other species (the same as in Chapter II) and therefore a separate database for every species is required. Hence, I also describe the assemblies used for the database creation of *E. coli*, *M. tuberculosis*, and *L. monocytogenes*. Then I will describe some adaptations of the preprocessing pipeline from Chapter III, also describing the new reference database underlying the model for *A. baumannii*. Then, I will describe the databases for the other species. Finally, I will lay out the two model extensions I developed. These will then first be applied to a new dataset of simulated metagenomic samples created from *A. baumannii* assemblies, after which an application to the other species is performed on the same databases and simulated datasets for each species as for Chapter II.

Assemblies underlying Reference Database Construction

For all four species, the same assemblies as in Chapter II were used. I lay out the basic properties here again for ease of reference.

A. baumannii

The curated dataset of 1473 assemblies by Hernández-González et al. [154] was used again. Here, 100 assemblies were excluded based on the assembly statistics and another 10% of the total database were excluded to again create samples from strains that are not in the reference database. This final reference database then contained 1240 assemblies. As before the total and reference databases were independently clustered at the 10-, 100-, and 500-cluster levels using complete linkage hierarchical clustering for data generation and analysis of results.

E. coli

For the *E. coli* dataset 74 of 4153 assemblies (Complete Genome level only) downloaded from the NCBI database were excluded based on assembly statistics. A further 10% of the total database were excluded when creating the reference database passed to the model, resulting in a final reference database consisting of 3671 assemblies. Both the total and reference database were clustered using complete linkage hierarchical clustering.

M. tuberculosis

For the *M. tuberculosis* dataset 703 assemblies (Complete Genome and Chromosome level) were downloaded from the NCBI database. 32 assemblies were excluded based on assembly statistics. A further 10% of the total database were excluded to create the reference database, resulting in a final reference database consisting of 607 assemblies. Both the total and reference database were clustered using complete linkage hierarchical clustering.

L. monocytogenes

For the *M. tuberculosis* dataset the 1649 assemblies provided by my collaborator Alexander Gmeiner at the Technical University of Denmark were used. 42 assemblies were excluded based on assembly statistics. A further 10% of the total database were excluded to create the reference database, resulting in a final reference database consisting of 1446 assemblies. Both the total and reference database were clustered using complete linkage hierarchical clustering.

Preprocessing and Database Creation

Beyond just expanding the strain-mixture model, I further re-evaluated the preprocessing pipeline. To briefly reiterate the procedure, Prokka was used on the reference database assemblies to identify coding sequences (genes). Then, single copy genes were identified and for each gene a MSA was performed using MAFFT to generate the gene MSAs. In contrast to Chapter III, we now proceed like this for all found single copy genes, not just the core genes, keeping track which of the genes are core genes and which accessory genes. Following this gene database creation, metagenomic reads are aligned against the gene MSAs using Bowtie2.

Firstly, manual inspection revealed that for some genes Prokka did not return the whole gene sequence carried by a strain in all reference strains, or identified genes that should not have been identified. Hence, I lowered the e-value (expected number of alignments by chance with at least as good an alignment score as the found alignment) used in the BLAST[166] (a widely used local alignment tool that enables quick local alignments of a sequence against a large database of reference sequences) search Prokka applies to 10^{-50}. This reduced the overall number of called genes, where the term genes again refers to known coding sequences, but noticeably increased the quality of the called genes and the resulting MSAs. Naturally, this further changed the number of core and non-core genes identified. With the adapted e-value 97 single-copy core genes and 759 single-copy accessory genes were identified. Of the 759 accessory genes 319 genes occurred in more than 95% of strains.

The higher number of core genes identified with the lower e-value even though in total fewer genes were identified is likely due to the fact that fewer erroneous gene calls lead to fewer core genes being identified as duplicate.

Next, I re-examined the alignments produced using Bowtie2. Here, a surprisingly large number of alignments differed in the reported number of matches from the actual number of matches upon individual inspection. Hence, reads were included in the analysis that should have been discarded due to their large divergence from the reference strains. To address this each aligned read was quality controlled following its addition to the MSA. The Hamming distance to all strains was determined and if all strains had a distance above 1 to the read it was excluded.

Lastly, I also addressed the possibility of using Panaroo for the identification of the gene pool of the reference strains. Panaroo is a tool that enables the creation of the pangenome of a prokaryotic species, promising accurate annotations of genomes. Panaroo identifies ORFs using Prokka and then clusters the resulting sequences using CD-hit (a tool for fast clustering of nucleotide sequences) with a 0.9 minimum identity threshold. This is followed by the creation of an MSA using MAFFT. The central difference here is the usage of CD-hit, which enables both the clustering of highly similar gene sequences, even when they were called with different identities or as duplicates, as well as the exclusion of genes that show high differences to other genes even though they were called as the same gene by Prokka. Gene groups with more than one sequence for at least one reference strain were excluded (equivalent to removal of duplicate genes). This procedure resulted in a larger database consisting of 15853 genes, 854 of which occurred in more than 95% of strains and 306 of which were found in 100%. However, for many simulated datasets (see below for details on data simulation), this approach generated a database that was too large to enable convergence of the accessory-proportion model (described below) within the maximum allotted runtime of 72 hours. I further attempted to remove genes that were only called as unidentified open reading frames. This resulted in the removal of 2631 genes. The resulting database contained 467 genes occurring in more than 95% of strains and 196 occurring in 100%. However, the models using this reduced database still did not complete for two samples, even while delivering highly accurate results (at assembly level for completed models computed as in Chapter II: $F1 = 0.92, JSD = 0.049, WSD = 0.00055$). Hence, using Panaroo for database construction was not deemed practical. Another possibility is to adapt the steps taken by Panaroo, i.e. to identify ORFs using Prokka and then performing a clustering via CD-Hit, but using a higher clustering threshold. This would likely result in fewer gene clusters present in all reference strains than using Panaroo, yet still in more gene clusters of high similarity than using just Prokka, but is left for future research.

Database generation

Next, I will describe the four databases that were created, one each for *A. baumannii*, *E. coli*, *M. tuberculosis*, and *L. monocytogenes*.

A. baumannii

The final database for *A. baumannii* contained 759 single-copy accessory genes and 97 single-copy core genes. The distribution of the genes across number of strains is shown in Figure 4.1. A clear u-shape can be observed, with 156 singletons. Overall, 319 genes occurred in more than 95% of strains.

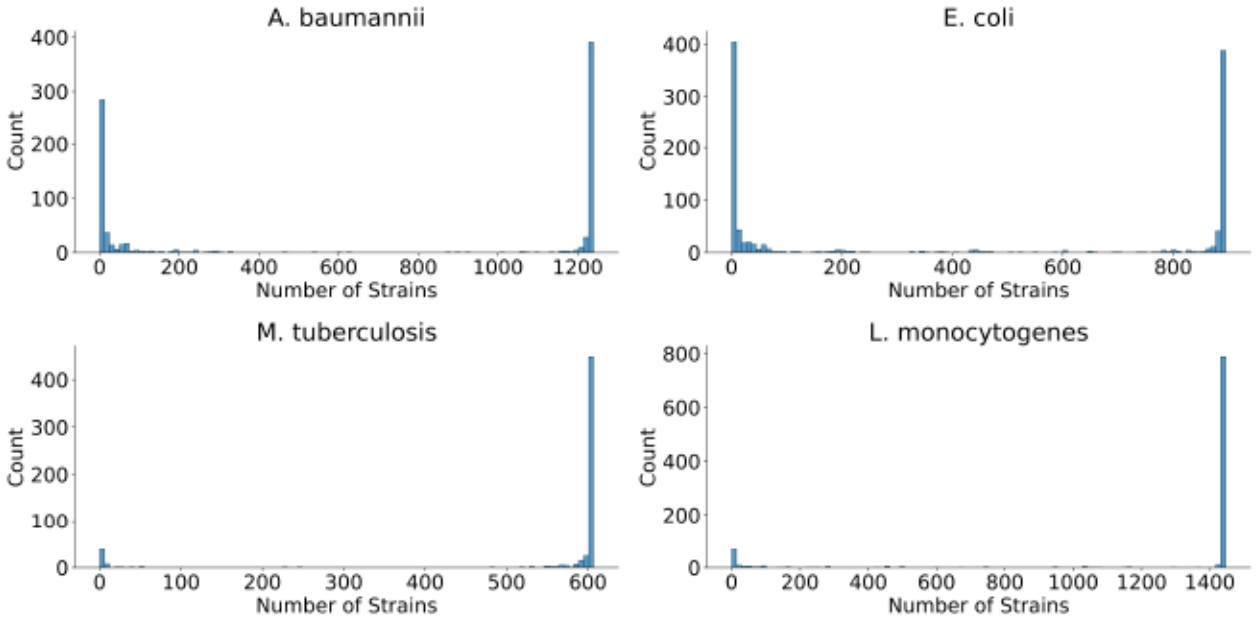


Figure 4.1: Number of assemblies/strains genes occur in for each species. Histogram use 100 bins each. Hence, the first and last bar do not necessarily represent the number of genes present in only one or all strains. A clear distinction between the distributions can be observed, with *M. tuberculosis* and *L. monocytogenes* showing a much higher number of genes shared across many or all strains and only few singletons compared to *A. baumannii* and *E. coli*. Total gene numbers were similar across all species besides *M. tuberculosis* for which fewer genes were identified (*A. baumannii*: 856; *E. coli*: 1056; *M. tuberculosis*: 571; *L. monocytogenes*: 933).

E. coli

For *E. coli*, applying the above approach resulted in only 4 single-copy core genes being identified. There is a high chance that this number of core genes is insufficient for accurate inference. This prompted to a re-investigate of the reference strains (following up on the quality control in Chapter II). All-to-all pairwise MASH distances were calculated for each species and for each strain the mean MASH distance to all other

strains was computed (Figure 4.2).[1] This analysis reveals that 20 of the *E. coli* strains have a mean MASH distance above 0.05 (roughly equivalent to $1 - ANI$, and hence, usable as a species cutoff). Therefore, these strains may be very unrelated to the rest of the reference strains and may not share many genes with the other strains. This led to the exclusion of these 20 strains, followed by a recomputation of the genome content of *E. coli*. However, this still only resulted in 5 core genes being identified. There are two possible reasons for observing such a low number of core genes. The first reason simply lies in the number of genomes (3671 genomes in total). As I show for *A. baumannii* below (Figure 4.5), the size of the core genome shrinks continuously the more strains are included. The second reason is that *E. coli* is a highly variable species. This high variability will likely lead to less overlap in gene content compared to less variable species. One way to address this issue is to reduce the number of strains included in the database preparation, either focusing on a single monophyletic group of strains, or clustering the database and choosing a strain from each cluster. Here, the second option was chosen. The *E. coli* reference database was down-sampled as was done for StrainScan (to obtain comparable results), i.e. by selecting 895 of the reference strains using complete-linkage hierarchical clustering. The aim is to retain much of the diversity of *E. coli*, but arrive at a larger core gene set. A total of 34 core genes and 1022 accessory genes were obtained, which was used as the final database. This number of core genes is still much lower than for either *A. baumannii* or *L. monocytogenes* even though both species' reference databases contained more strains. This further adds to the notion of high variability within *E. coli* strains, but could notably also point to higher number of duplicate genes, i.e. the low number of core genes could also be explained by many genes being present at least twice in at least two assemblies. Indeed, when considering coding sequences that were found multiple times the core genome size was 334 genes (with 3494 accessory genes identified), suggesting that gene duplication is an important factor. Incorporating duplicate genes will therefore be a focus of future research. The distribution of the number of strains genes are found in is shown in Figure 4.1.

M. tuberculosis

For *M. tuberculosis* 228 single-copy core genes and 343 single-copy accessory genes were identified. The distribution of the number of strains containing genes is shown in Figure 4.1. 497 genes occurred in more than 95% of strains and 29 singletons were identified.

[1] Note also that the pattern of average MASH distances recapitulates our stipulations from Chapter II, in particular regarding the very low average MASH distances of *M. tuberculosis*.

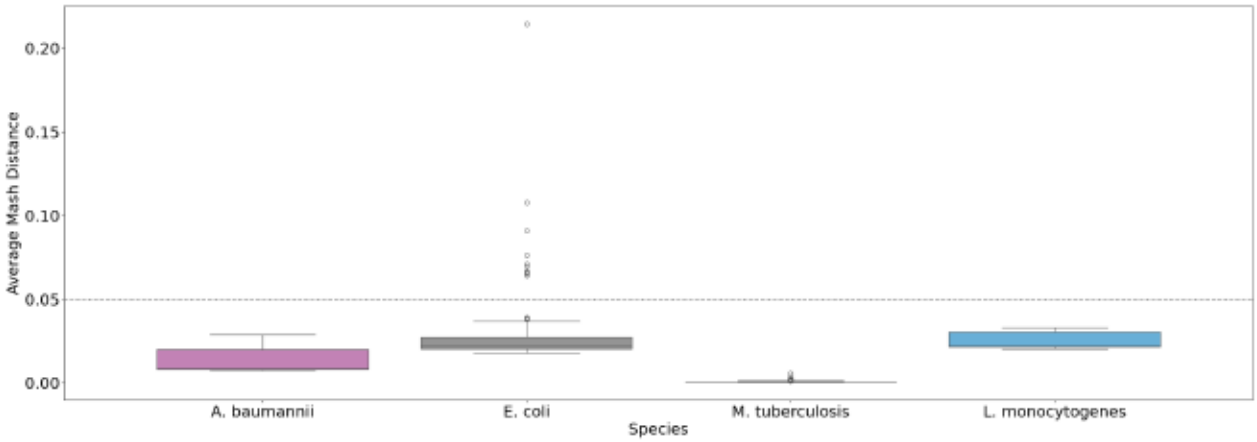


Figure 4.2: Average Mash distances of strains from all other strains in the species for the four analyzed species.

L. monocytogenes

A total of 933 coding sequences and 647 core genes were extracted for *L. monocytogenes*. An attempt to run the model with this number of genes exceeded our computational limitations, with no convergence within 72 hours. Hence, it was necessary to reduce the core genome size. A subset of 90 genes was selected. First, the 100 genes with the most variable positions in their MSAs were identified. However, the 10 genes with the most variable positions were excluded, as these were clear outliers suggesting potentially inconsistent gene calling by Prokka. This resulted in a final subset of 90 genes. In total, 799 genes occurred in at least 95% of strains, and only 29 singletons were identified. The distribution of the number of strains genes appear in is shown in Figure 4.1. Similar to *M. tuberculosis*, most identified genes occur in almost all or all strains, with a clear u-shape of the distribution and relatively few genes being unique to only a few or one strain.

Accessory-Presence Model

In the first model, the accessory-presence model, only the presence or absence of accessory genes in both the metagenomic samples and the reference strains is considered. To achieve this, we can introduce the vector Γ. Each entry Γ_g is 1 if at least one metagenomic read originating in gene $g \in \mathcal{G}$ is observed and 0 otherwise. Here, $\mathcal{G}$ is the set of the accessory genes only. Whether or not a core gene is observed gives no information to the model, as all reference strains will carry this gene. To obtain Γ the same Bowtie2 alignment as before was performed and reads that did not have a perfect CIGAR (Compact Idiosyncratic Gapped Alignment Report) were filtered out. A CIGAR is a combination of numbers and letters summarizing the alignment of two sequences. A number indicating the number of positions is followed by a letter indicating the type of alignment for these positions. For example, a CIGAR of 4M1S10M would indicate that the first four positions

of the alignment are matches, followed by a single substitution, and then another 10 matches. Additionally, we can define a gene presence-absence matrix for the reference strains Λ. Here, $\Lambda_{s,g}$ is 1 if reference strain s carries gene g and 0 otherwise. This will enable the minimization of the difference between the genes we observe and the ones we do not observe and whether or not they are present in the estimated reference strains. For each accessory gene this difference is

$$\sum_{s \in \mathcal{S}} |X_s \Lambda_{s,g} - \Gamma_g| \,, \tag{4.1}$$

where X_s is again the decision variable indicating whether strain s has a non-zero proportion. The issue into which we run here is that it is possible that $\sum_{s \in \mathcal{S}} X_s \Lambda_{s,g} > 1$. This would be the case if two or more estimated strains carry the gene g and are estimated to be present in the sample. To resolve this problem, let us define the decision variable $V_g \in \{0, 1\}$ that takes on the value 1 if at least one strain is both present in the estimation ($X_k = 1$) and carries gene g ($\Lambda_{s,g} = 1$). To this end, V_g is constrained by

$$\begin{aligned} V_g &\geq X_{\mathcal{S}_1} \Lambda_{\mathcal{S}_1,g} \\ V_g &\geq X_{\mathcal{S}_2} \Lambda_{\mathcal{S}_2,g} \\ &\vdots \\ V_g &\geq X_{\mathcal{S}_\mathcal{B}} \Lambda_{\mathcal{S}_\mathcal{B},g} \\ V_g &\leq \sum_{s \in \mathcal{S}} X_s \Lambda_{s,g}, \end{aligned}$$

where the last line ensures that if none of the reference strains carrying a gene is estimated with a non-zero proportion then V_g must also be 0. $\mathcal{S}_i$ is simply a unique way of indexing the strains in arbitrary order. The following is then added to the objective function

$$\sum_{g \in \mathcal{G}} |V_g - \Gamma_g|. \tag{4.2}$$

Due to the absolute value we again need to redefine this summation. Given that Γ_g is a constant we can make the following consideration. If $\Gamma_g = 0$, then the above expression will evaluate to V_g. On the other hand, if $\Gamma_g = 1$, then the above expression evaluates to $1 - V_g$. Hence, for genes where $\Gamma_g = 0$, we can add

$$\sum_{g \in \mathcal{G} \wedge \Gamma_g = 0} V_g \tag{4.3}$$

to the objective function, as well as

$$\sum_{g \in \mathcal{G} \wedge \Gamma_g = 1} 1 - V_g, \tag{4.4}$$

for genes where $\Gamma_g = 1$. Both of these new objective function parts are weighted by $\lambda_v / \mathbf{G}$. Lastly, the variables X_s are not constrained above, i.e. they are free to attain the value 1, only adding a cost to the objective function. Hence, if a gene is required to be present then X_s for a strain s carrying that gene could just be estimated at 1 with P_s being a very small, but non-zero, proportion. To prevent this from happening the constraints

$$P_s \geq X_s - 0.95$$

are introduced for all strains $s \in \mathcal{S}$. This family of constraints will impose a minimum proportion of 5% for any strain. The threshold can naturally be adapted to assumptions about the data analyzed, though more than 20 distinguishable strains in one sample are unlikely to occur. Furthermore, these constraints may be unnecessary, given the constraint on sparsity. However, adding the constraints introduces little computational overhead. Hence, it is kept in the final formulation. The complete model can be formulated as

Accessory-Presence Model

$$\text{minimize} \quad \sum_{s\in\mathcal{S}} X_s + \frac{\lambda_p}{\mathbf{L}} \sum_{(l,n)\in\mathcal{L}\times\mathcal{N}} \Omega_l D_{l,n} + \lambda_v \sum_{g\in\mathcal{G}\wedge\Gamma_g=0} V_g + \lambda_v \sum_{g\in\mathcal{G}\wedge\Gamma_g=1} (1-V_g)$$

$$\text{s.t.} \quad X_s \in \{0,1\}, s \in \mathcal{S}$$

$$P_s \in [0,1], s \in \mathcal{S}$$

$$D_{l,n} \in [0,1], l \in \mathcal{L}, n \in \mathcal{N}$$

$$V_g \in \{0,1\}, g \in \mathcal{G}$$

$$X_k \geq P_s, s \in \mathcal{S}$$

$$P_s \geq X_s - 0.95, s \in \mathcal{S}$$

$$D_{l,n} \geq \delta_l \Psi_{l,n} - \sum_{s\in\mathcal{S}} P_s \Phi_{l,s,n}, l \in \mathcal{L}, n \in \mathcal{N}$$

$$D_{l,n} \geq \sum_{s\in\mathcal{S}} P_s \Phi_{l,s,n} - \delta_l \Psi_{l,n}, l \in \mathcal{L}, n \in \mathcal{N}$$

$$\sum_{s\in\mathcal{S}} P_s = 1$$

$$V_g \geq X_s \Lambda_{s,g}, g \in \mathcal{G}, s \in \mathcal{S}$$

$$V_g \leq \sum_{s\in\mathcal{S}} X_s \Lambda_{s,g}, g \in \mathcal{G}$$

Accessory-Proportion Model

The above formulation may work well when estimating the presence or absence of exact reference strains in a metagenomic sample. However, in many cases, a metagenomic sample might contain a strain that is closely related to a reference strain, but has acquired accessory genes that are not present in the reference strain. To address this, I introduce the accessory-proportion model. First, let $\Lambda_{s,g}$ be replaced by the decision variables $T_{s,g} \in \{0,1\}$. Then, to capture the presence of genes in the reference strains, we can constrain these new variables by

$$T_{s,g} \geq \Lambda_{s,g} \tag{4.5}$$

for all $s \in \mathcal{S}$, $g \in \mathcal{G}$. The accessory model part presented here only considers accessory genes, such that $\mathcal{G}$ is the set of the accessory genes only. However, we do not need constraints when $\Lambda_{s,g} = 0$ and can therefore

drop these. Additionally, the following can be added to the objective function

$$\sum_{s \in \mathcal{S}} \sum_{g \in \mathcal{G}} T_{s,g}. \tag{4.6}$$

The sum is weighted by λ_t. The above objective prevents any $T_{s,g}$ from being assigned 1 unnecessarily. Simply replacing $\Lambda_{s,g}$ with $T_{s,g}$ in the previous model (Equation 4.1) to obtain gene presence will not work, due to the non-linearity introduced by the multiplication with the decision variable X_s. To resolve this issue, let us introduce the additional decision variables $R_{s,g} \in \{0, 1\}$ which are constrained via

$$R_{s,g} \geq T_{s,g} + X_s - 1 \tag{4.7}$$

for all $s \in \mathcal{S}$, $g \in \mathcal{G}$. These variables are constrained to 1 if $T_{s,g}$ and X_s are both 1, i.e. when strain s is estimated as present in the sample and estimated to carry gene g. To constrain them to 0 otherwise we add the constraints

$$R_{s,g} \leq T_{s,g}$$
$$R_{s,g} \leq X_s.$$

Naturally, we only need to define these constraints for gene-strain combinations where $\Lambda_{s,g} = 0$. For $\Lambda_{s,g} = 1$, we have $T_{s,g} = 1$, and hence, the above constraints simplify to

$$R_{s,g} = X_s.$$

A remaining issue in this formulation is that the assignment of a gene to any strain is independent of proportions, and therefore, arbitrary. To address this, let us first take a look at the per-gene coverage of all genes. In particular, we can consider the coverage of the core genes, which we can assume to be, on average, equal to the sum of the individual frequencies of the constituent strains. Then, take the coverage of each non-core gene relative to its length (in bp) compared to the average core gene length and divide it by the mean coverage of core genes to arrive at the relative abundance of each non core gene $A_g \in [0, 1]$, which will reflect the relative abundance of that gene among the expected total coverage of all strains. For example, consider again a gene carried only by one strain with a 0.3 relative abundance. Then, if we observe a total species wide coverage (or equivalently, a core genome coverage) of 10, we would expect the accessory gene to have a coverage of 3. We would then assign $A_g = 0.3$ for the accessory gene. In this manner, if we estimate the strain in question at a relative abundance of 0.3, we would get a difference of zero to the observed accessory

gene proportion, as desired. To achieve this, a decision variable for each gene-strain combination is required that is equal to the proportion of strain $s \in \mathcal{S}$ (P_s) if $X_s = 1$ and $T_{s,g} = 1$. $R_{s,g}$ has already been defined. With this in mind, we can define $H_{s,g} \in [0,1]$ and add the following constraints

$$H_{s,g} \leq R_{s,g}$$
$$H_{s,g} \leq P_s$$
$$H_{s,g} \geq P_s + R_{s,g} - 1$$

for all $s \in \mathcal{S}$, $g \in \mathcal{G}$. Hence, $H_{s,g}$ will be zero if either strain s has a zero proportion or if the strain does not carry gene g. $H_{s,g}$ will be equal to P_s otherwise. Now we can add

$$\sum_{g \in \mathcal{G}} \left| A_g - \sum_{s \in \mathcal{S}} H_{s,g} \right| \tag{4.8}$$

to the objective function instead of the gene presence-absence objective in Equation 4.2. This then gives us the deviation between the observed frequencies of the accessory genes and the estimated frequencies based on $T_{s,g}$ and P_s, which is minimized. Again, we need to redefine the above due to the absolute value as

$$I_g \geq A_g - \sum_{s \in \mathcal{S}} H_{s,g}$$
$$I_g \geq \sum_{s \in \mathcal{S}} H_{s,g} - A_g$$

with the new decision variables $I_g \in [0,1]$. Finally, add

$$\sum_{g \in \mathcal{G}} I_g \tag{4.9}$$

to the objective function, weighted by λ_t. This gives us a complete model able to estimate known reference strains with the addition of as few as possible non-core genes not present in these reference strains. Additionally, by the simplification of working on a gene level for the accessory genes, computational complexity that would be introduced by having to consider each of the base loci in the accessory genome is avoided. Lastly, a minimum strain proportion is introduced, as in the accessory-presence model. The complete model takes the form

Accessory-Proportion Model

$$\text{minimize} \quad \lambda_s \sum_{s \in \mathcal{S}} X_s + \frac{\lambda_p}{\mathbf{L}} \sum_{(l,n) \in \mathcal{L} \times \mathcal{N}} \Omega_l D_{l,n} + \frac{\lambda_i}{\mathbf{G}} \sum_{g \in \mathcal{G}} I_g + \frac{\lambda_t}{\mathbf{G}} \sum_{s \in \mathcal{S}, g \in \mathcal{G}, \Lambda_{s,g}=0} T_{s,g}$$

$$\text{s.t.} \quad X_s \in \{0,1\}, s \in \mathcal{S}$$

$$P_s \in [0,1], s \in \mathcal{S}$$

$$D_{l,n} \in [0,1], l \in \mathcal{L}, n \in \mathcal{N}$$

$$T_{s,g} \in \{0,1\}, s \in \mathcal{S}, g \in \mathcal{G}, \Lambda_{s,g} = 0$$

$$R_{s,g} \in \{0,1\}, s \in \mathcal{S}, g \in \mathcal{G}, \Lambda_{s,g} = 0$$

$$H_{s,g} \in [0,1], s \in \mathcal{S}, g \in \mathcal{G}$$

$$I_g \geq 0, g \in \mathcal{G}$$

$$\sum_{s \in \mathcal{S}} P_s = 1$$

$$X_s \geq P_s, s \in \mathcal{S}$$

$$P_s \geq X_s - 0.95, s \in \mathcal{S}$$

$$D_{l,n} \geq \delta_l \Psi_{l,n} - \sum_{s \in \mathcal{S}} P_s \Phi_{l,s,n}, l \in \mathcal{L}, n \in \mathcal{N}$$

$$D_{l,n} \geq \sum_{s \in \mathcal{S}} P_s \Phi_{l,s,n} - \delta_l \Psi_{l,n}, l \in \mathcal{L}, n \in \mathcal{N}$$

$$R_{s,g} \geq T_{s,g} + X_s - 1, s \in \mathcal{S}, g \in \mathcal{G}, \Lambda_{k,g} = 0$$

$$R_{s,g} \leq T_{s,g}, s \in \mathcal{S}, g \in \mathcal{G}, \Lambda_{k,g} = 0$$

$$R_{s,g} \leq X_s, s \in \mathcal{S}, g \in \mathcal{G}, \Lambda_{k,g} = 0$$

$$R_{s,g} = X_s, s \in \mathcal{S}, g \in \mathcal{G}, \Lambda_{k,g} = 1$$

$$H_{s,g} \leq R_{s,g}, s \in \mathcal{S}, g \in \mathcal{G}$$

$$H_{s,g} \leq P_s, s \in \mathcal{S}, g \in \mathcal{G}$$

$$H_{s,g} \geq P_s + R_{s,g} - 1, s \in \mathcal{S}, g \in \mathcal{G}$$

$$I_g \geq A_g - \sum_{s \in \mathcal{S}} H_{k,g}, g \in \mathcal{G}$$

$$I_g \geq \sum_{s \in \mathcal{S}} H_{k,g} - A_g, g \in \mathcal{G}$$

Minimum Evidence for Identification

As last step in making the models applicable to metagenomic samples we need to define a cutoff for the minimum evidence required to identify a species as present. Currently, the models will deliver an estimate even if only a single read alignment is found. We can utilize the assumption that we need sufficient coverage for a confident identification. If the average core gene coverage falls below a pre-set threshold for which I chose 0.1 bases per nucleotide, the species is considered absent.

Read Assignment following Strain Identification

Whereas for single strain samples inferring SNPs could be fairly straightforward (considering observed SNPs and sequencing error probability), this is not the case for samples containing multiple strains. To lay a foundation for SNP assignment (as well as other post-processing steps) a model for assigning the metagenomic reads to the inferred strains was developed. For unambiguous reads (i.e. those mapping to one strain uniquely) this is again easy. For other reads we need to rely on the relative abundance estimate. However, we can utilize a simplified version of the strain-allele model from Chapter III. The aim is to assign reads to strains within one gene at a time. First, let us introduce a binary variable $M_{s,r}$ for all estimated strains s and reads r. The fraction of reads assigned to a strain should be as close as possible to the strains proportions (now given as data) P_s. Note that for genes not estimated to be present in all strains, we need to modify P_s by rescaling it to the total proportion of all strains carrying the gene we are currently investigating. Call $\mathbf{R}$ the total number of reads assigned to gene g. Then we get the objective function part

$$\sum_{s \in \mathcal{S}} \left| P_s - \frac{1}{\mathbf{R}} \sum_{r \in \mathcal{R}} M_{s,r} \right|, \tag{4.10}$$

where $\mathcal{S}$ is now the set of estimated strains and $\mathcal{R}$ is the set of reads assigned to the gene of interest. This objective minimizes the mismatch between the estimated strain proportion and the proportion of reads assigned to that strain. To account for the number of mismatches between the reads and the strains, reads are assigned to the strain with the minimum mismatch. The mismatch matrix $E_{s,r}$ is assumed to be given, calculated as for the strain-allele model. The number of mismatches can then be calculated as

$$\sum_{r \in \mathcal{R}, s \in \mathcal{S}} E_{s,r} M_{s,r}, \tag{4.11}$$

which is also part of the objective function. Additionally, the constraints

$$\sum_{s \in \mathcal{S}} M_{s,r} = 1 \tag{4.12}$$

for all $r \in \mathcal{R}$ prevent reads from being assigned to multiple strains. Further, restructuring the absolute value in the objective function via introduction of a new decision variable $Z_s \geq 0$ and introducing weight λ_m results in the full model

Read to Strain Assignment model

$$\begin{aligned}
\text{minimize} \quad & \lambda_m \sum_{r \in \mathcal{R}, s \in \mathcal{S}} E_{s,r} M_{s,r} + \sum_{s \in \mathcal{S}} Z_s \\
\text{s.t.} \quad & M_{s,r} \in \{0,1\}, s \in \mathcal{S}, r \in \mathcal{R} \\
& \sum_{s \in \mathcal{S}} M_{s,r} = 1, r \in \mathcal{R} \\
& Z_s \geq P_s - \frac{1}{\mathbf{R}} \sum_{r \in \mathcal{R}} M_{s,r}, s \in \mathcal{S} \\
& Z_s \geq \frac{1}{\mathbf{R}} \sum_{r \in \mathcal{R}} M_{s,r} - P_s, s \in \mathcal{S}
\end{aligned}$$

This method provides a set of reads for each strain as well as the strains' reference sequence from which SNPs can be estimated. This can be done by taking the reads for a given strain, aligning them to its core genes, and using one of many popular SNP calling methods[200,201]. Importantly, this feature was not evaluated on simulated data, but will be a focus of future research.

Data Generation

The two models were again evaluated on simulated data. I generated 20 samples at 10-fold coverage with 5 samples each for 1, 2, 3, and 4 reference strains using ART with the same settings as in Chapter II:

```
./art_illumina -ss HS25 -sam -i -l 150 -p -m 600 -s 100
```

Strains were chosen such that they originated in different clusters at the 100-cluster level. First, n clusters at the 100-cluster level were randomly sampled without replacement were n is the number of strains in the sample. Then one strain from each cluster was chosen at random to be included in the data. All strains had

equal relative abundances. The samples had an average of 130000 paired end reads (SD = 3800 paired end reads).

The final model that performed best on the evaluation data above was further evaluated on the same dataset as used in Chapter II, i.e. a total of 840 samples (for details see Appendix Table A1), 210 for each species among *A. baumannii*, *E. coli*, *M. tuberculosis*, and *L. monocytogenes*. For each species, 200 of the 210 samples contained strains that were also in the reference database, while 10 samples consisted of strains that were previously excluded from the reference database (see Chapter II for details). Additionally, the NIST dataset was used again to evaluate the best performing model on mock metagenomic data. Using the same data as in Chapter II enables in-depth analysis of performance, especially in species that were not included in the model evaluation to determine the final model. Furthermore, it provides the opportunity to compare performance to the two best performing published tools, StrainScan and StrainGE.

Results

Model performance was evaluated using the cluster-based F1-scores and JSDs as defined in Chapter II. Additionally, a comparison on the strain level was also performed, including the F1-score, JSD, and WSD as described in Chapter II. The new models have some additional parameters that need to be determined and a grid of different parameter values was explored. The parameter settings are shown in Table 4.1 and the complete results are shown in Figure 4.3.[2] While at a high-level clustering both accessory genome models perform similarly and outperform the strain-mixture model, it is clear that the performance increase over the strain-mixture model is highest at higher taxonomic resolutions. Furthermore, the accessory-proportion model clearly outperforms the accessory-presence model at these higher taxonomic resolutions. Nonetheless, the high performance of the strain-mixture model at low taxonomic resolution is notable, pointing towards applicability in practical scenarios. Additionally, the variance in performance in each model across different parameter settings is very small.

Following this first evaluation, the accessory-proportion implementation was modified such that the sparsity objective for the number of genes represented the devised model accurately. The previous parameter settings then led to low quality results. Specifically, the models largely overestimated the number of strains, using all possible 20 strains (due to the constraint of $P_s \geq 0.05$). This can be remedied by adapting the parameters

[2]For the accessory-proportion model, the sparsity objective for the number of accessory genes was improperly implemented. However a recomputation for all parameter settings was not possible due to resource constraints. Instead parameter choices were based on the theoretical considerations below, delivering similar performance to the accessory-proportion model as shown in Figure 4.3 (see Figure 4.4).

Parameter	Strain-Mixture Models	Presence-AbsenceModels	Accessory-Proportion Models
λ_p (Weight for Equation 3.2)	1, 10, 100	1, 10, 100	1, 10, 100
λ_v (Weight for Equations 4.3 and 4.4)	N/A	0.01, 0.1, 1, 10	N/A
λ_i (Weight for Equation 4.8)	N/A	N/A	0.01, 0.1, 1, 10
λ_t (Weight for Equation 4.6)	N/A	N/A	0.1, 1, 10

Table 4.1: Parameter settings used for model tuning. $\lambda_s = 1$ (weight of the sparsity objetive) and $\gamma_{min} = 1$ (minimum nucleotide coverage) were used across models. λ_p is the weight for the objective minimizing the mismatch in observed and estimated nucleotide proportions for the SCCGs. λ_v weighs the objective of matching as many gene presence or absence observations as possible. λ_i is the weight that determines the importance of the deviation between observed and estimated proportions of accessory genes in the optimization. Finally, λ_t is the weight corresponding to the minimization of the number of genes estimated for the strains that are not carried by these strains in the database.

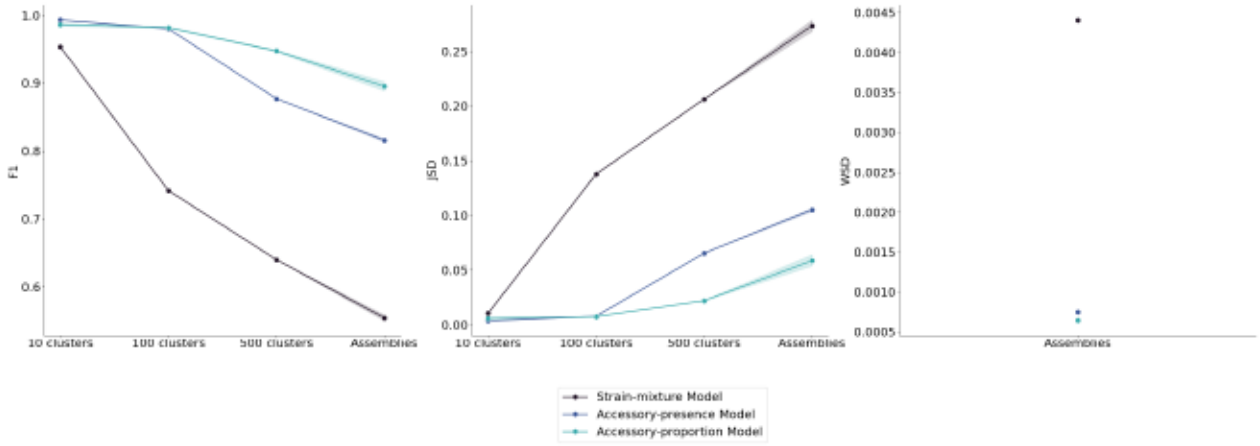


Figure 4.3: Performance of the different tested models across different taxonomic resolutions (10, 100, and 500 cluster level as well as for the identification of individual assemblies) across all parameter settings. Assembly label on x-axis indicates the analysis where recovery of unique assembly genomes is considered. Each line shows the mean performance averaged across all parameter settings for the corresponding model and samples. Shaded area shows the 95% confidence interval across parameter settings (i.e. each datapoint corresponds to the mean performance of the model with one set of parameters across all 20 samples).

according to a few considerations. Firstly, observe that I_g (measuring the deviation between the observed and estimated gene proportions) and $T_{k,g}$ (measuring the number of estimated genes not found in the estimated reference strains) are interdependent. Specifically, if $T_{k,g} = 0$ (possibly due to a high weight λ_t) for all strains of a gene, then $I_g > 0$ for all genes observed and $I_g = 0$ for all genes not observed. Given that $T_{k,g} = 1$ if a reference strain carries a given gene, these $T_{k,g}$ are forced to be one. Hence, a high weight on λ_t will favor estimating many strains to explain potentially spurious genes found. Secondly, the objective function part including $T_{k,g}$ showed a very high value after optimization. This indicates that the most influential objective is to reduce $T_{k,g}$. While one may think that a large number of estimated strains will lead to a higher $T_{k,g}$, observation showed that the estimated strains originated from the same cluster, indicating high genetic similarity, which minimizes the influence on $T_{k,g}$. Furthermore, a constraint on $T_{k,g}$ was already imposed implicitly. Specifically, if there are no or very few reads observed for a given accessory gene then the estimated proportion should be as close as possible to zero. This is true in particular if the number of reads observed as fraction of the average core gene coverage is closer to zero than to the estimated strain

abundance, though these are of course still interdependent during optimization. Hence, $T_{k,g} = 0$ is optimal in this situation and including $T_{k,g}$ in the objective function may not be necessary. Based on these observations, λ_t was set to 0, while the other weights were set to the values of one of the previously best performing models ($\lambda_p = 10, \lambda_s = 1, \lambda_f = 0.01$). For this models only one strain was estimated for each sample, indicating that λ_s or λ_f is too high. Hence, λ_s was set to 0.1. This led to high performance (Figure 4.4). However, the number of strains was still slightly underestimated. For samples containing 3 strains, one sample was estimated as only containing 2 and for samples containing 4 strains all but one of the samples were estimated to contain fewer than 4 strains. Hence, the model was rerun with $\lambda_s = 0.01$. The model with $\lambda_s = 0.01$ clearly outperforms the model with $\lambda_s = 0.1$ on all metrics (Figure 4.4).

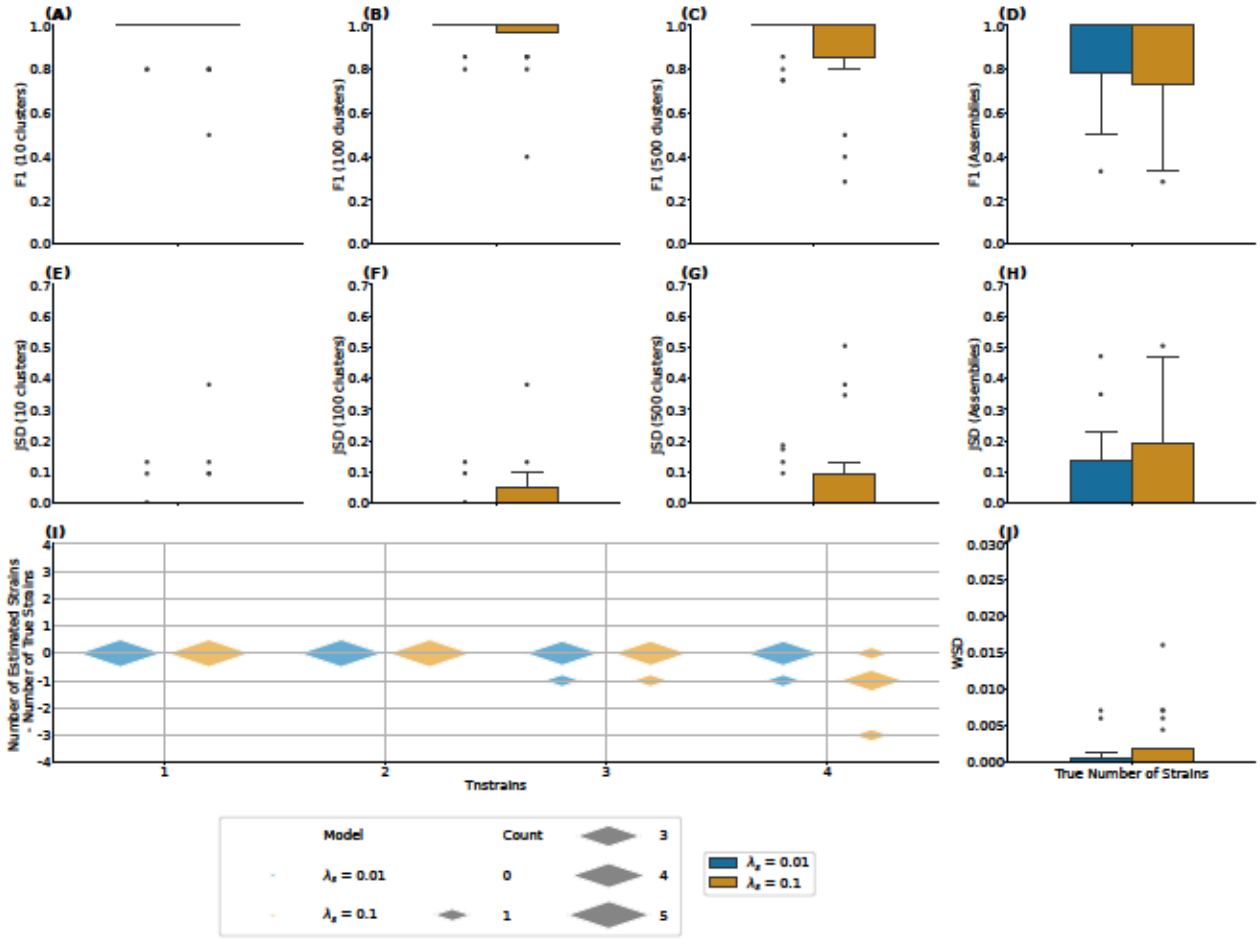


Figure 4.4: Performance characteristics of accessory-proportion models with $\lambda_s = 0.01$ and $\lambda_s = 0.1$. Shown are F1-scores **(A-D)**, and JSDs **(E-H)** at the different levels of clustering (10 clusters, 100 clusters, 500 clusters, and assemblies, i.e. no clustering). WSD at the assembly level is shown in **(J)**. For the boxplots, line indicates median, edges of box the 25% and 75% quartile, and whiskers extend to include the datapoint furthest from the median that is within 1.5 times the interquartile range. Dots are outliers. **(I)** shows the difference between the estimated number of strains versus the true number of strains in the samples. Size of diamonds indicates the count of how often the difference is observed.

Following obtaining the above results, investigations were focused on the final accessory-proportion model

(APM from hereon). As the number of reference strains is increased, the number of single copy core genes present in 100% of strains will likely decrease for many or all species, which may indicate a dependence of core genes on sampling. To confirm this for *A. baumannii*, the number of core genes from random subsets of reference strains were determined (repeated 100 times for each subset size, subset sizes increased from 1 to 1371 in increments of 10). The results show that the rate of decline in the number of core genes decreases, however, no complete leveling out is observed, even when including all available reference strains (Figure 4.5). This further solidifies the conclusion that the identification of a gene as core gene depends on which assemblies of the species are sampled. Hence, another model was tested, which was equivalent in its formulation to the APM, but including genes present in more than 95% of strains in the core genome and implementing a linear transition between the nucleotide and gene based portions of the model (see Appendix B for details and other modifications that were considered). Evaluation of this model with the same parameter settings that achieved the best results previously, i.e. $\lambda_p = 10, \lambda_s = 0.01, \lambda_j = 0.01$, and $\lambda_t = 0.0$ resulted in 11 out of 20 samples not converging within the maximum runtime of 72 hours. Hence, this model was not further considered.

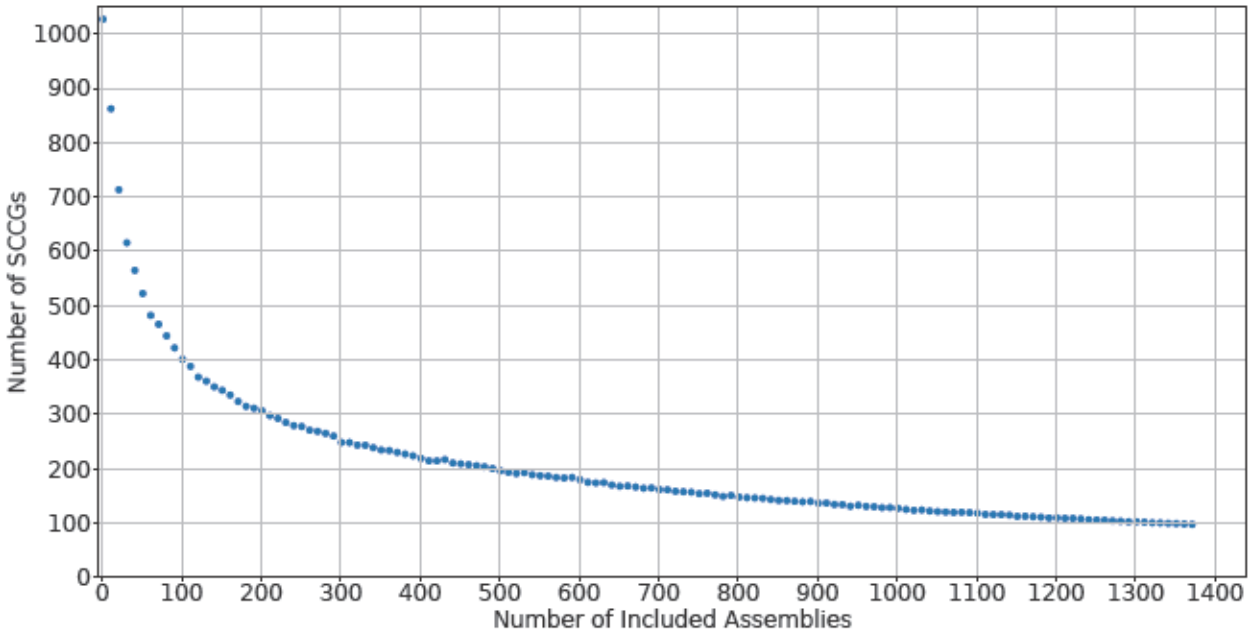


Figure 4.5: Mean number of SCCGs for different number of randomly sampled assemblies. Each number of assemblies was sampled 100 times.

Gene Content Recovery

Next, the ability of the APM for gene recovery was investigated. Specifically, which non-core coding sequences identified at least once in the reference database strains does the APM identify as present or absent, and do the coding sequences identified correspond well to the ones that are actually present in the strains in the sample? As all of the samples were composed from strains among the reference strains ideally exactly the genes of these strains would be called. For incorrectly identified strains we would then expect the genes not present in those strains to be identified as present ($T_{s,g} = 1$), while we naturally may observe some false positives due to the constraint $T_{s,g} = 1$ for genes g carried by strain s. The F1-score, precision, and recall for gene recovery are shown in Figure 4.6A. Near perfect recall is observed, with only one false negative. Further, high precision is observed (Mean = 0.89, SD = 0.015). The precision is slightly higher when only one strain is contained in the sample (Figure 4.6D; Mean = 0.90, SD = 0.010), likely as the model was able to better recover the actual strain in the sample (Figure 4.4). Notwithstanding the high recall, some false positive gene calls are observed (Figure 4.6C). Given that the strains in question are contained among the reference strains, ideally zero false positives would be observed and the perfect recall is not surprising. Due to this, the analysis of gene recovery is repeated in the next section for samples composed of strains not in the reference database and mock metagenomic data, which constitutes a much more stringent test. Investigation of the per nucleotide coverage of the recovered genes (Figure 4.6B) shows clearly distinct distributions between true and false positive gene calls. This implies that a minimum coverage threshold for calling a gene can be set, depending on the requirements towards sensitivity and specificity.

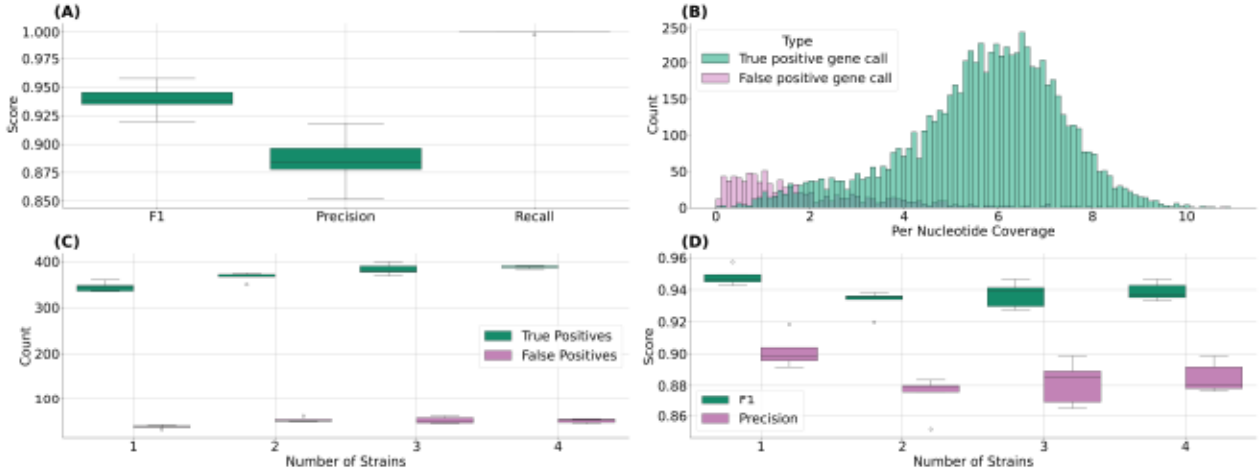


Figure 4.6: **(A)** F1-score, precision, and recall for inferred gene content across samples. Recall is almost perfect, resulting in low visibility of boxplot (line at 1). **(B)** Histogram of per nucleotide coverage for true positive and false positive gene calls. **(C)** Number of true and false positive genes inferred by number of strains in the sample. **(D)** F1-score by number of strains.

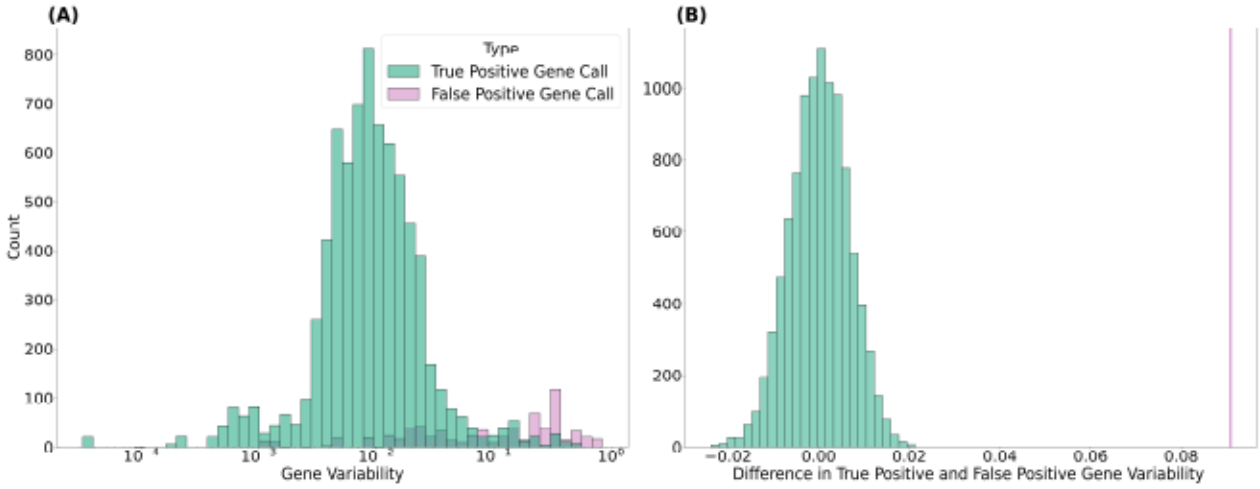


Figure 4.7: **(A)** Histogram for true and false positive gene calls by gene variability, defined as the mean of all-to-all pairwise hamming distances between strains. **(B)** Histogram of differences in gene variability from randomly permuted genes. Line indicates the observed mean difference between variability in true and false positive gene calls.

The false positive identifications deserve further investigation. For a part of the false positive gene calls, high coverage is observed (Figure 4.6B). Given that the data is only generated from strains in the reference database, it is reasonable to hypothesize that these sequences are actually present in at least one of the generating strains. Hence, the issue potentially lies with the gene calling using Prokka. One way to investigate this is to look into the properties of the genes called as false or true positives. Varying the e-value for the gene calling via Prokka is another option, which was explored for mock metagenomic data below. Here, I chose to look into the variability of the called genes as proxy for likelihood to be called reliably from a genome. Higher variability may indicate lower chance of being identified by Prokka as the same gene given a low e-value. For each gene, all-to-all pairwise hamming distances between the strains gene variants were calculated and the mean of these distances was used as the genes variability measure. When looking at the genes called across all samples (non-uniquely, i.e. considering the same gene multiple times if it was called multiple times as true or false positive) shows that false positively called genes generally show higher variability (Figure 4.7A) supporting the hypothesis. To investigate statistical significance, a permutation test with 10000 permutations was performed (Figure 4.7B). Each permutation randomly permutes the labels of the date (here true positive or false positive). Then, the difference between groups can be taken. By performing many such permutations, a null hypothesis distribution for the difference between groups is obtained against which the actually observed difference between groups can be compared. This test showed the gene variability to be significantly increased among false positive gene calls (5-σ-level = 0.033, Observed Difference = 0.091). Finally, 371 false positive gene calls representing singletons were observed (genes only present in one strain), for which gene variability can naturally not be determined, but for which a similar conclusion may still hold.

Runtime

An analysis was conducted to gain an insight into how the APM's runtime scales with the input size. Here, of the 97 SCCGs nine subsets of genes were created and the analysis was rerun with these subsets. Each of these subsets represents a down-sampled core genome database of smaller size. The nine subsets consisted of 10, 20, 30, 40, 50, 60, 70, 80, and 90 genes sampled randomly from the 97 total SCCGs. For each of these subsets the APM was rerun on 10 of the simulated samples (one for each number of strains from one to five). The total runtimes of the model depending on core genome size is shown in Figure 4.8. This analysis shows that when the core genome database consists of only a few genes (10, 20, or 30) there is a large variability in runtimes. However, for more genes, a clearer pattern is discernible, with runtimes slightly increasing with more included core genes. Importantly, there is a significant jump in processing time when switching from 90 to 97 core genes. This may indicate that above 97 core genes this steep increase may persist, though this is hard to infer from the available data. Nonetheless, the very sudden increase in processing time is a notable feature and deserves further investigation, which is left for future research.

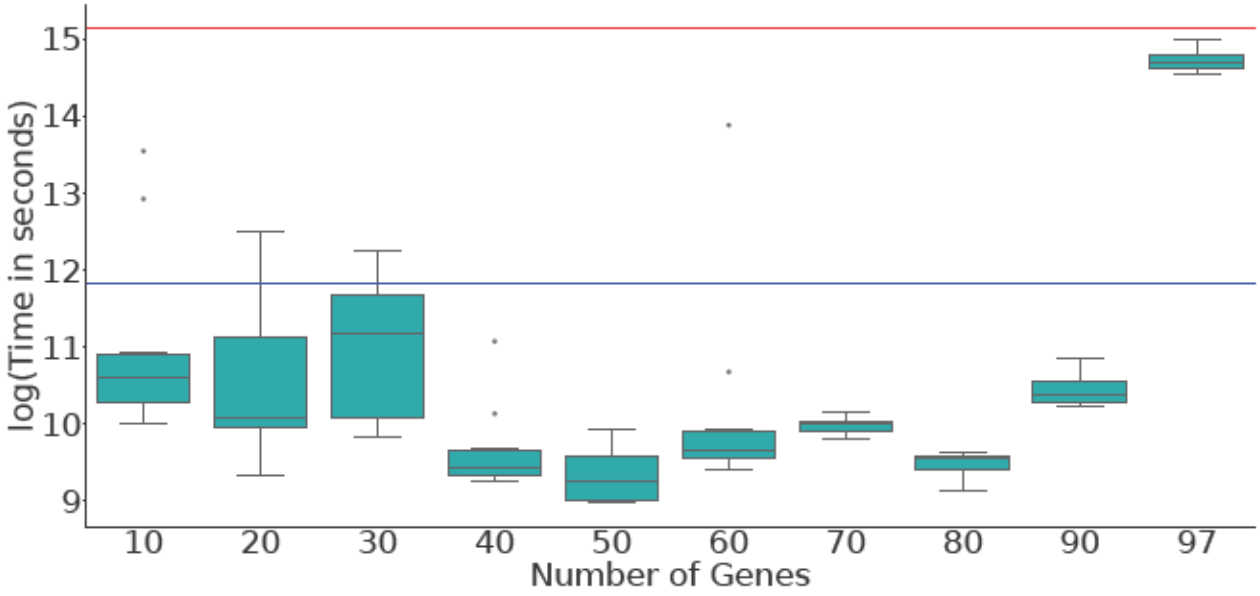


Figure 4.8: Processing times for different number of core genes in the reference database on a log base 2 scale. For the boxplots, line indicates median, edges of box the 25% and 75% quartile, and whiskers extend to include the datapoint furthest from the median that is within 1.5 times the interquartile range. Dots are outliers. The blue line indicates one hour and the red line ten hours.

Evaluation on other species

Next, the final APM ($\gamma_{min} = 1, \lambda_p = 10, \lambda_s = 0.01, \lambda_f = 0.01, \lambda_t = 0.0$) was applied to the datasets described in the benchmarking study of strain identification tools (Chapter II). To reiterate briefly, 210 samples were created for each species. 200 of the samples were created from strains that are also available to the APM and the other tools in their respective reference databases. These samples varied in the number of strains (1, 2, 3, or 4 strains), their coverage (1-, 5-, 10-, or 20-fold, with 1-fold coverage only tested for samples with one strain), different strain relatedness (strains either all originating in one cluster or different clusters as determined at the 100 cluster level), and different relative abundance profiles (either all strains had equal relative abundance or one strain was picked at random to have 80% relative abundance while all other strains had an equal share of the remaining 20% relative abundance). 10 of the samples were created from strains that were not contained in the reference databases (the 10% excluded strains as described in the Methods in Chapter II). These samples were composed of either 1 or 2 strains, with equal relative abundance, originating in different clusters, at a 10-fold coverage. Details of the samples can be found in Table A1. Performance was compared to the results obtained for StrainScan and StrainGE in Chapter II. Firstly, the core gene databases were constructed for each species as described in the method section and the final number of identified core and accessory genes are summarized again in Table 4.2.

Species	Number of core genes	Number of accessory genes
A. baumannii	97	662
E. coli	34	1022
M. tuberculosis	228	343
L. monocytogenes	90	286

Table 4.2: Number of core and accessory genes for the different analyzed species used in the final version of APM.

For the samples made up of strains contained in the reference database the APM showed competitive performance (Figure 4.9). For *A. baumannii*, the APM shows a performance profile similar to StrainGE, and while showing lower performance at high resolutions, outperforms StrainScan at low resolutions. Additionally, the lowest WSD for *A. baumannii* is observed using the APM. For *E. coli*, the APM shows similar performance to StrainScan, while outperforming it on the WSD metric. StrainGE performs best for this species, not considering the assembly level, where performance of all tools is close to equal. Next, when investigating *M. tuberculosis*, slightly lower performance is observed for both F1-scores and JSD at the lowest resolution of 10 clusters using the APM. However, for higher resolutions StrainGE is slightly outperformed. Lastly, inferences on *L. monocytogenes* closely match the quality of StrainGE. When looking at the number of inferred strains, a similar performance profile to StrainScan can be discerned (Figure 4.10). Overall, the number of estimated strains match the true number of strains slightly better than for StrainScan, however, generally sparser re-

sults than would be optimal are still obtained. Further reducing the sparsity weight, or even removing it, may improve the models performance. Testing this is left for future research, but the ability to adjust the weight enables the adjustment of the model to a particular species.

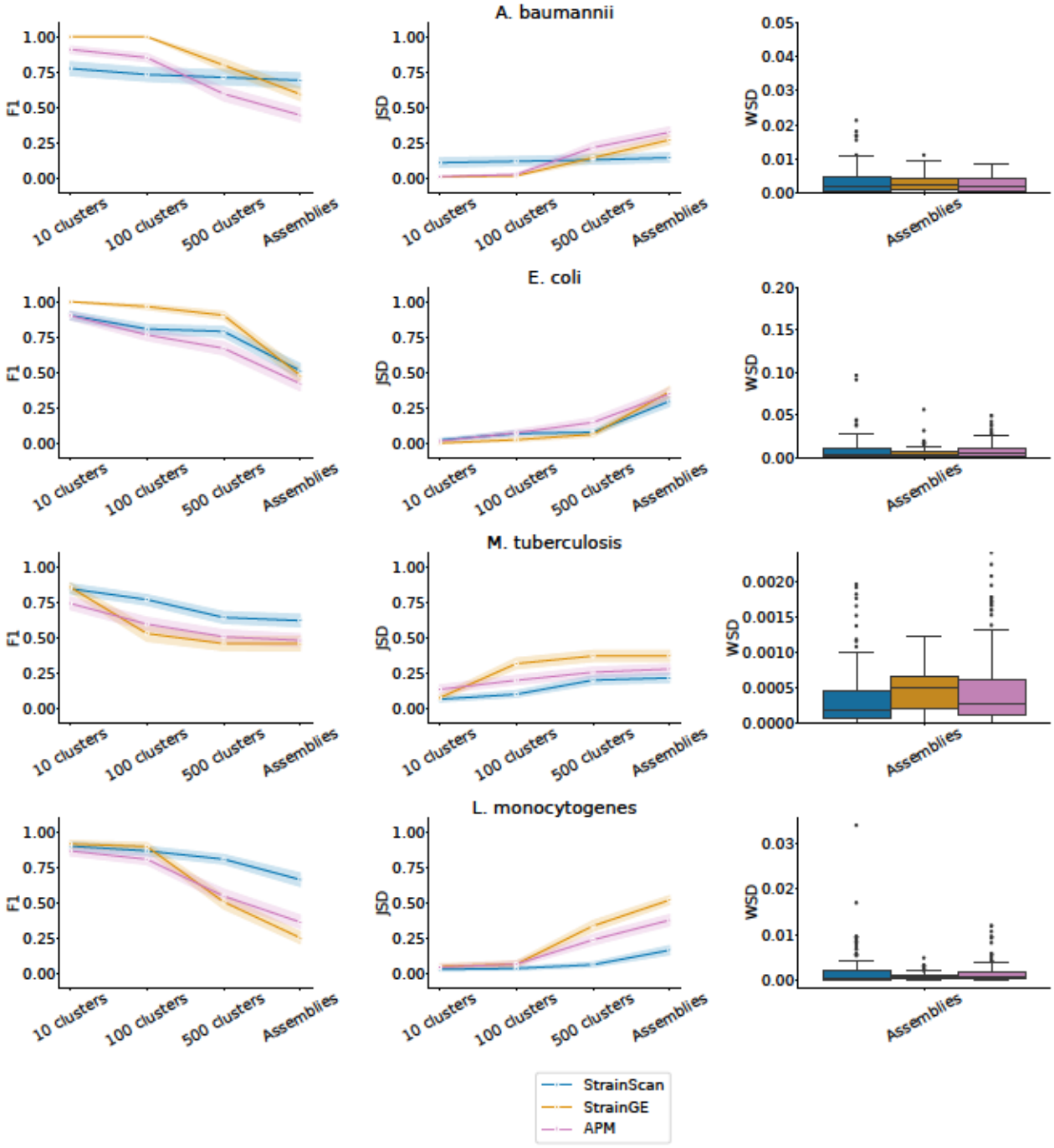


Figure 4.9: F1-scores, JSDs, and WSD for the APM, StrainScan, and StrainGE for the samples made up of strains contained in the reference database. Shaded areas of lineplots indicate 95% confidence interval. For the boxplots, line indicates median, edges of box the 25% and 75% quartile, and whiskers extend to include the datapoint furthest from the median that is within 1.5 times the interquartile range. Dots are outliers. WSD for *A. baumannii* does not show WSDs for StrainScan for samples which failed to run. Assemblies label on x-axis indicates assembly level analysis.

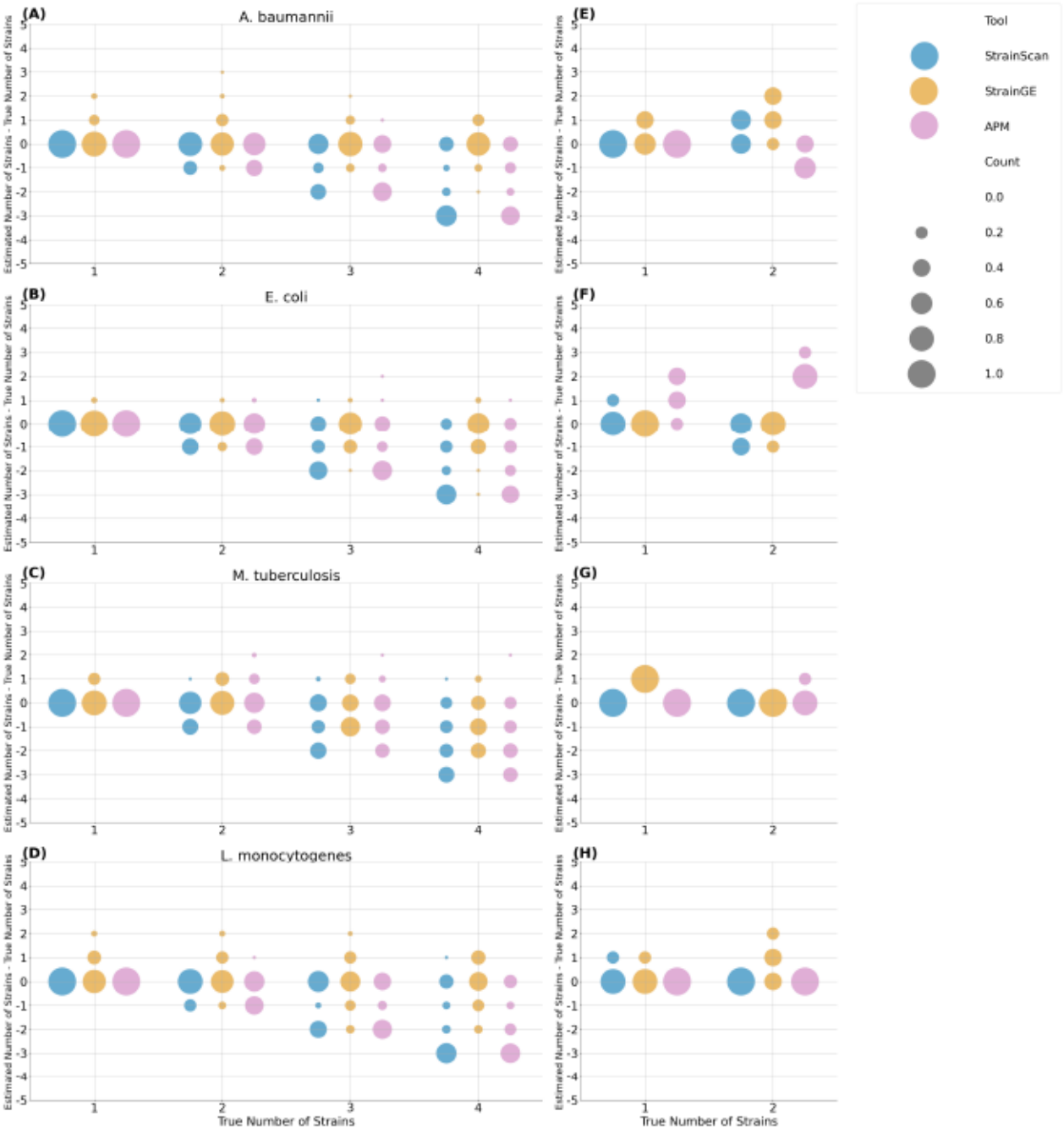


Figure 4.10: Number of inferred strains versus true strains for StrainScan, StrainGE, and the APM on **(A-D)** the samples made up of strains contained in the reference database and **(E-H)** the samples made up of strains not contained in the reference database.

Establishing good performance for the samples made up of strains not contained in the reference database is essential in validating the present model. Again, performance comparable to StrainGE and StrainScan is observed overall (Figure 4.11A). For *E. coli* the APM has a highly reduced performance. The reasons for this are not clear, as the database as well as the performance on samples made up of strains contained in the

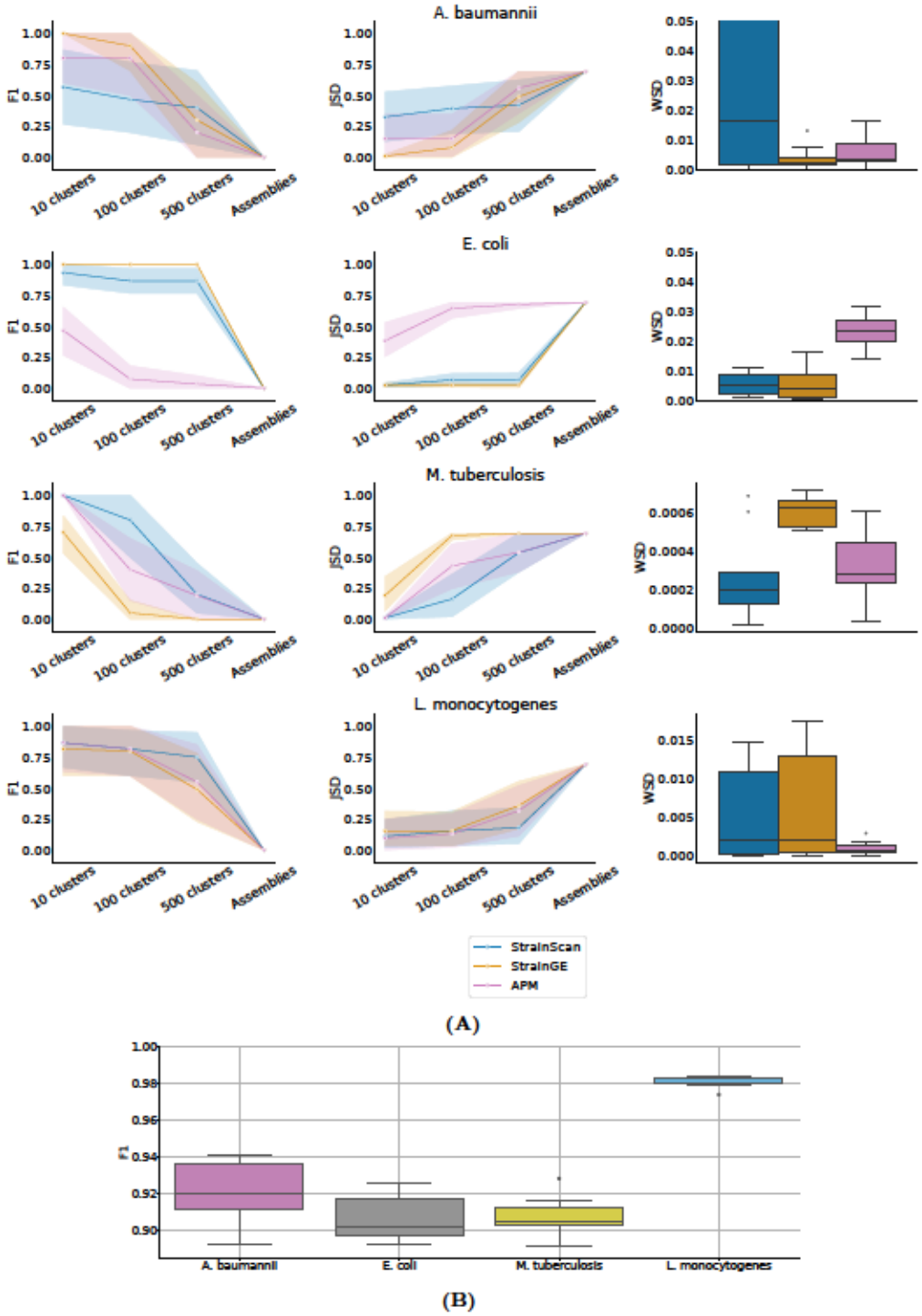


Figure 4.11: **(A)** F1-scores, JSDs, and WSD for the APM, StrainScan, and StrainGE for the samples made up of strains not contained in the reference database. Shaded areas of lineplots indicate 95% confidence interval. For the boxplots, line indicates median, edges of box the 25% and 75% quartile, and whiskers extend to include the datapoint furthest from the median that is within 1.5 times the interquartile range. Dots are outliers. WSD for *A. baumannii* is truncated on the y-axis to increase visibility. Assemblies on x-axis indicates assembly level analysis. **(B)** F1-scores for accessory gene assignment in samples made up of strains not contained in the reference database across species.

reference database is essentially identical to StrainScan. Of course, the number of core genes considered is much lower for *E. coli* than for the other species (Table 4.2), which may be a factor here. It is still surprising to observe much higher performance for the samples made up of strains contained in the reference database. Further extending the gene database, for example by including genes that are present in less than a 100% of strains, may prove valuable here. When investigating the number of inferred strains it is clear that the APM tends to overestimate the number of strains in the sample (Figure 4.10). This explains the lower performance at least to some degree. For *M. tuberculosis* comparable performance to StrainScan is attained by the APM, while the same is true for *A. baumannii* and StrainGE. However, a higher WSD for *A. baumannii* is observed using the APM compared to StrainGE. The inferred numbers of strains for these two species closely track the true number of strains. For *L. monocytogenes*, high accuracy of the APM is observed at the the 10 and 100 cluster levels. Further, the APM shows the lowest JSD at these resolutions. At the 500 cluster resolution, StrainScan performs slightly better. However, applying the APM results in by far the lowest WSD. Similarly, only the APM always estimates the correct number of strains for *L. monocytogenes*, showing that for this species the new approach outperforms the available methods. The high accuracy for *L. monocytogenes* was again confirmed when investigating the accessory gene content recovery (Figure 4.11B), where significantly a higher F1-score for *L. monocytogenes* compared to the other species was achieved. Nonetheless, for the other species comparable accuracy to the initial evaluation of gene recovery can be seen, with a median F1-score above 0.9 for every species. Finally, the APM's performance when investigated for different sample characteristics shows the expected pattern of results (Appendix Figures B1-B4).

Evaluation on Mock Data

As last step, it is important to validate the APM on mock metagenomic data, which was done by applying it to the NIST mock metagenomic datasets (see Chapter II for more details on the samples). Here, high performance is observed overall (Table 4.3). For *A. baumannii*, low Mash distances and WSDs are achieved, comparable to what was achieved for StrainScan and StrainGE. For *L. monocytogenes*, the the highest overall performance is observed, returning the best possible result across samples. Across *A. baumannii*, *M. tuberculosis*, and *L. monocytogenes* no false positive strains are inferred. On the other hand, false identifications for *E. coli* are present, similar as observed for the other tools. Further, a drop in accuracy measured via Mash distances and WSDs compared to StrainScan and StrainGE can be seen for *E. coli*. It is worth noting that the APM is the only tool that did not return a false positive identification for *E. coli* for sample C. Additionally, while there is a drop in performance for identification of *A. baumannii* in sample C, where the relative abundance is only 0.03%, the identification is still of good quality, and neither StrainScan nor

StrainGE recovered any strains for this sample. These results point to the conclusion that the APM is more robust when investigating low abundance samples, which will be very valuable when investigating real metagenomic data, as oftentimes low abundances will be observed.

Finally, I again evaluated accessory gene content recovery. Precision is similar to the one for simulated data of *A. baumannii* and the samples made up of strains not contained in the reference database, though a slight drop is observed. For both *A. baumannii* and especially *L. monocytogenes* the APM infers few false positives while recovering many true positive genes (Figure 4.12). Notably, this is limited to genes in the database, which are restricted by definition and therefore excludes duplicate genes. However, it would be fairly straightforward to extend the model by adding genes of interest to the reference database. Given that these will be accessory genes and due to the relatively low λ_j the skew in observed proportion should not impact the results significantly. For *E. coli* many false positives are inferred, which is in line with the number of false positive strain identifications and the generally reduced performance for *E. coli*. Importantly, performance was evaluated across different e-values used for gene calling via Prokka in the genomes of the mock strains (for which assemblies were available). As would be expected, there are more false positive identifications at lower e-values, though the magnitude of this effect is limited. The increased number of false positives is likely due to fewer genes being identified in the mock strains. This demonstrates that the gene calling parameters have a noticeable effect. However, the effects are small enough to not affect any of the previous results. While this is not optimal, precision is high across settings for the mock metagenomic data. This does provide confidence that high accuracy is achievable for real metagenomic data, though the species analyzed should be taken into account as demonstrated by the lower performance for *E. coli* and species specific benchmarks should be conducted before application.

Discussion

In the last two chapters, I presented the development and evaluation of a metagenomic strain identification tool. I started by developing and evaluating several model variants, based solely on single copy core genes. Having established the best performing core genome model, I here extended it to include the accessory genome. The final tool shows competitive performance when compared with the state-of-the-art tools as established in Chapter II. In particular, the APM show the best performance for *L. monocytogenes* and provides reliable identification in mock metagenomes, especially for low abundance species. Furthermore the APM is tuneable and interpretable, providing independent information on the core and the accessory genome and enabling the identification of acquired genes, and will provide a valuable addition to the bioinformaticians' toolbox.

NIST Sample	Species	Tool	True Abundance	# Inferred Strains	Mash difference	WSD
A	*A. baumannii*	StrainScan	2.6%	3	0.012	0.018
		StrainGE		1	**0.000024**	**0.000096**
		APM		1	0.00012	0.00012
	E. coli	StrainScan	30.4%	1	0.0035	**0.0035**
		StrainGE		2	**0.00083**	0.012
		APM		2	0.12	0.015
	L. monocytogenes	StrainScan	0.03%	0	N/A	N/A
		StrainGE		0	N/A	N/A
		APM		1	**0.00024**	**0.00024**
	M. tuberculosis	StrainScan	0.0%	0	0	0
		StrainGE		0	0	0
		APM		0	0	0
B	*A. baumannii*	StrainScan	30.1%	1	**0.000024**	**0.000096**
		StrainGE		1	**0.000024**	**0.000096**
		APM		1	0.00012	0.00012
	E. coli	StrainScan	0.04%	1	0.23	0.23
		StrainGE		2	**0.13**	**0.11**
		APM		2	**0.13**	0.21
	L. monocytogenes	StrainScan	0.0%	0	0	0
		StrainGE		0	0	0
		APM		0	0	0
	M. tuberculosis	StrainScan	0.0%	0	0	0
		StrainGE		0	0	0
		APM		0	0	0
C	*A. baumannii*	StrainScan	0.03%	0	N/A	N/A
		StrainGE		0	N/A	N/A
		APM		1	**0.0028**	**0.0028**
	E. coli	StrainScan	0.0%	1	N/A	N/A
		StrainGE		1	N/A	N/A
		APM		**0**	**0**	**0**
	L. monocytogenes	StrainScan	3.9%	1	0.0011	0.0011
		StrainGE		1	0.0012	0.0012
		APM		1	**0.0**	**0.0**
	M. tuberculosis	StrainScan	0.0%	0	0	0
		StrainGE		0	0	0
		APM		0	0	0
D	*A. baumannii*	StrainScan	0.0%	0	0	0
		StrainGE		0	0	0
		APM		0	0	0
	E. coli	StrainScan	2.76%	1	0.0035	0.0035
		StrainGE		1	**0.00083**	**0.00083**
		APM		1	0.0035	0.0035
	L. monocytogenes	StrainScan	30.0%	2	0.00024	0.00056
		StrainGE		1	0.0012	0.0012
		APM		1	**0.0**	**0.0**
	M. tuberculosis	StrainScan	0.0%	0	0	0
		StrainGE		0	0	0
		APM		0	0	0
E	*A. baumannii*	StrainScan	7.6%	1	0.00039	0.00046
		StrainGE		1	**0.000024**	**0.000096**
		APM		1	0.00012	0.00012
	E. coli	StrainScan	8.8%	2	0.0035	0.089
		StrainGE		3	**0.00083**	0.034
		APM		3	0.085	0.059
	L. monocytogenes	StrainScan	9.6%	1	0.0011	0.0011
		StrainGE		1	0.0012	0.0012
		APM		1	**0**	**0**
	M. tuberculosis	StrainScan	0.0%	0	0	0
		StrainGE		0	0	0
		APM		0	0	0

Table 4.3: Results for NIST samples. True abundance indicates the percent of the sample (including all species) the strain of the species listed in the row makes up (e.g. the *A. baumannii* strain makes up a total of 2.6% of sample A). Note that percentages need not to sum to one within a sample, as more than the listed species are contained in the sample. # inferred strains column shows the number of strains for the species each tool inferred (one strain would be optimal). Mash difference indicates the difference between two Mash distances. The first Mash distance is the minimum of the distances between the inferred strains and the strain in the sample. The second is the minimum distance between the strain in the sample and all strains in the reference database. The first measures how close to the inferred strain the best inferred strain is. The second measures the Mash distance obtained by the best possible inference given the database. As such this column measures how close to the best possible result the inference is. Similarly, the WSD measures the same difference, however, this time for the WSD of the inferred population to the true strain and the best possible WSD given the database strains. In this manner, the number of strains inferred is taken into account. Bold fonts indicate the best performing model for a metric-sample-species combination.

Contributions

For strains that are contained in the reference database, the presented model performs on par with StrainScan and StrainGE. The same is true for strains not in the reference database, besides when considering *E. coli*

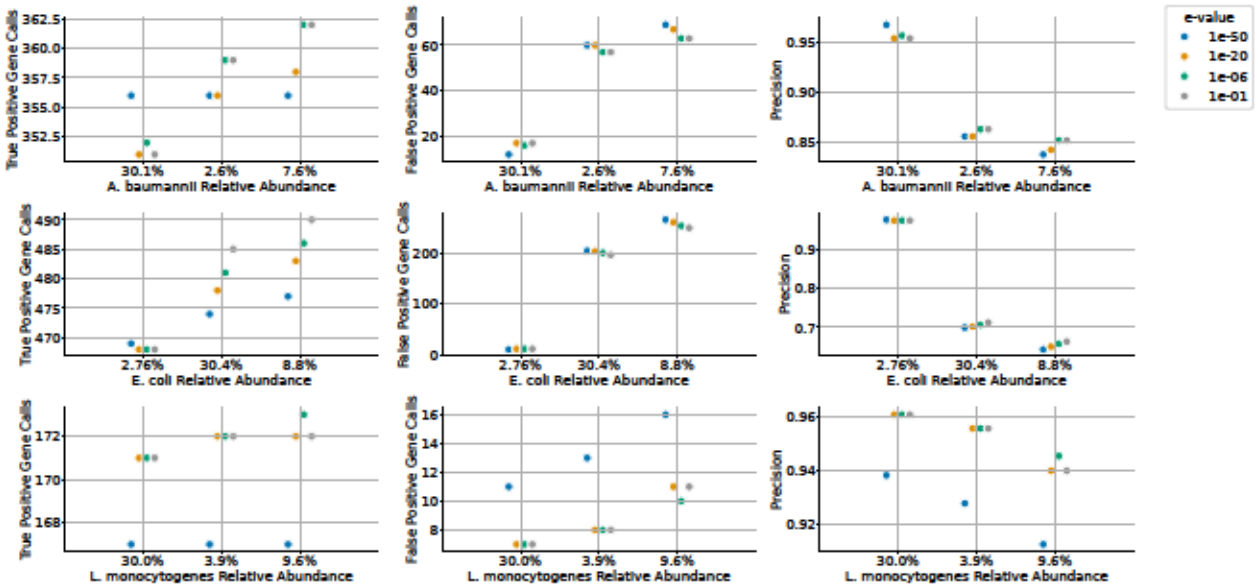


Figure 4.12: True and false positive gene calls as well as precision of gene calling depending on *e*-value used for BLAST search in Prokka for the isolate organism in the samples for *A. baumannii*, *E. coli*, and *L. monocytogenes* in NIST mock metagenomic data. Note the different in scales between the figures to allow for better visibility.

for which a clear dropoff in performance was observed. However, I believe that further species-specific tuning of the APM would provide improved results for *E. coli*. The APM is also highly accurate in analysing mock metagenomic data, where particularly reliable performance is demonstrated at low relative abundances. This will enable practical metagenomic analyses, where many samples will likely show low coverage for the species we may be interested in analysing. While not always delivering as high an accuracy as the other tools, the accuracy is still more than sufficient for most species to conduct valuable metagenomic research, especially when not considering the highest possible taxonomic resolution, but rather conducting analyses at a clade or lineage level. Furthermore, the APM provides better interpretability and customizability compared to StrainScan and StrainGE. First of all, the different weights in the APM can be adjusted easily and the effect of this modification can be predicted and interpreted. As such, when validating different tools on a species of interest prior to application, the APM can be adapted to increase accuracy or with a specific focus on either core or accessory genes. Future work will address this in more detail. Furthermore, the database based on genes enables modifications of the APM to different circumstances. For example, a specific set of genes could be chosen, be that a set of MLST, core genome MLST, or whole genome MLST (where the chosen loci are not constrained to genes in the core genome) loci or a custom set of genes. This too will need to be evaluated in future research, but the core genome based model presented in Chapter III already shows the viability of relying on relatively few core genes for strain identification. I further proposed an approach to assign reads to strains and genes, with this functionality still to be tested, which can be valuable for

downstream analysis, such as SNP calling or assembly. The latter could also benefit from the knowledge of the number of strains in a sample to identify the potential for chimeric assemblies. In a similar vein, the reference strains found by the APM could be used as references for more general read assignment, extending beyond the genes contained in the model, enabling more in-depth analysis of SNPs, deletions, and insertions in the genomes of these strains. Finally, the APM offers the functionality of estimating genes, specifically coding sequences identified using Prokka, present in an identified strain that are not present in the identified reference strains and assigning these genes to strains based on their observed coverage. This functionality is, to the best of my knowledge, not offered by any other available tool and tackles an important issue in high resolution metagenomic analysis. Future work could additionally expand on this, as from a technical standpoint the model structure allows for the incorporation of any sequence. The gene recovery results show high precision and recall. Notably, false positive identification are a factor, however, a distinct distribution in coverages for these is observed, enabling the choice of a sensitivity-specificity trade-off. Still, the method is not necessarily applicable to any species, as seen for *E. coli*, and species specific benchmarks should be conducted prior to application. By extending the method to genes of special interest, such as antibiotic resistance genes, the APM constitutes an important step towards our ability to assign highly relevant genes to strains in metagenomic data and evaluating these possibilities is essential going forward.

Limitations and Future Work

While the model presented here constitutes a valuable addition to the list of available metagenomic analysis tools, there are some limitations. First of all, note the dependence of the reference database on the size and structure of the included strains. As we have seen for *E. coli* and *A. baumannii*, the number of single copy core genes decreases with the number of included genomes. This is not surprising, especially considering that only single copy genes were included. As such the resulting database has to be investigated and potentially steps have to be taken to adjust it. For example, the number of included *E. coli* strains had to be reduced to include a sufficient number of core genes. On the other hand, too many genes may be identified resulting in too much data to process with limited processing time, requiring exclusion of certain genes, as had to be done for *L. monocytogenes*. To address the choice of genes, two possible research avenues exist. First of all, duplicate genes could be included. However, this may lead to low quality alignments in the first instance. Hence, clustering of ORFs identified from Prokka, such as done by Panaroo using CD-Hit, should be performed to group highly similar ORFs together. However, this leads to two further problems. Firstly, the inclusion of duplicate genes will tend to skew the observed proportions for those genes. A possible remedy for this would be to adjust the reference strains nucleotide proportions, for example $\{0.5, 0.5, 0, 0\}$ may be a

possible entry for the nucleotide proportions of $\{A, T, G, C\}$ at a locus. Secondly, the database size will likely increase substantially, though one may reduce the number of included genes for example by determining a set of genes that optimally distinguishes between the reference strains by some criteria. For example, the minimal set of genes that has a given minimum number of nucleotide differences between any two strains could be chosen. Another approach would be to analyze the core gene MSAs, either by manual inspection or based on gene variability, and include only the best suited genes. The exact criteria here will need to be established via further experiments. As an alternative to including duplicate genes with similar advantages and drawbacks, one could extend the genes included in the database beyond pure core genes present in 100% of strains. For example, genes that are present in at least 95% of reference strains could be included, or if this leads to a database that is too large, the cutoff percentage could be decreased until a set number of genes which we know to be computationally feasible is reached. Beyond this, while I aimed to create a mostly automated tool, curation of the database may be valuable, and one may consider manual inspection of the resulting gene alignments and exclusion of certain genes or parts thereof based on this. This may further improve the database, and hence, performance of the APM. Another approach here would be to rely on essential genes, i.e. genes that have been identified as essential to an organisms survival[187,202]. Using essential genes may generally be beneficial as their number will not depend as strongly on sampling, as opposed to the core genes used here, whose dependence on sampling is shown in Figure 4.5. These essential genes, however, may be duplicated, and do not allow straightforward automatic extraction, and both issues will need to be tackled in research going forward to allow the usage of essential genes in the model. The second limitation is the false positive identification of genes, though our confidence in gene calls can be increased through imposing a minimum coverage cutoff. The analysis of the mock metagenomic data also highlights again the variability in gene calling using different e-values. Therefore, identifying genes using a high e-value with post hoc quality assessment may be another avenue to address the issue of false positive gene identifications. At the moment, the model only estimates the presence of additional genes, but not deletions. This capability could be introduced directly into the APM by extending the $T_{s,g}$ variable to instances where $\Lambda_{s,g} = 1$, i.e. to genes g carried by reference strain s. However, this is not necessary, as a post-processing step could be incorporated instead, identifying genes that are not covered by any reads, similar to the postprocessing employed by StrainGE. The APM has the drawback, as do all other available tools, that no information on plasmids is provided. In general, assigning plasmids to strains is a hard problem, complicated by the fact that oftentimes plasmids can be shared across species boundaries[203]. Nonetheless, substantial specificity of plasmids to their hosts remains[45,46]. Hence, including plasmid carried genes with high host specificity could be possible in the APM. However, determining which plasmids display such a specificity is a hard problem in the first place that deserves more attention in future research. A final limitation is that no estimate of

confidence is provided for the inferred strain mixture. This drawback is also present for both StrainScan and StrainGE. Addressing this limitation in future research will be valuable.

Applications

The developed model has a range of possible applications. The identification of strains in environmental metagenomic data, for example from wastewater or the urban microbiome, can help with the surveillance of strain transmission and sharing across locations and environments. The developed model could be used to identify variation in core genes, which could then be used as a basis for building a phylogeny of the identified strains, helping to track the spread of certain clades or strains, which can be hugely beneficial to surveillance efforts, in particular outbreak tracking. Identification of shared strains across environments could in turn be applied to the prediction of antibiotic resistance, where the sub-species level taxonomy across regions could be associated with patterns in the resistance profiles observed in clinics. This is an interesting area for future research to address. Furthermore, the ability to identify genes carried by the identified strains provides opportunities to study the functional roles of strains within a microbiome. One area of possible research would be to identify genes shared between the members of a species found in similar environments, which may point towards ecotypes (a ecologically distinct subspecies[204]) and genes essential in these environments for the given species. This in turn could provide insights into the functional roles of species in different environments. To date, uncovering functional roles of species in environments and how their presence corresponds to community dynamics remains elusive[205]. Extending the model database to include non-single copy genes and potentially ARGs and genes carried on plasmids could further increase the value of this approach. Especially the assignment of resistance genes to strains will be of great importance for example in wastewater surveillance and in research addressing the relationship between clinical and wastewater antibiotic resistance profiles. Notably, this application will require further development and testing, but this work shows that in the near future these areas of application may become feasible.

Conclusion

In conclusion, I presented a novel model for strain inference in metagenomic samples that shows state-of-the-art performance and works especially well for low abundance samples, as demonstrated by the analysis of mock metagenomic data. While further developments are necessary to exploit the APM's full potential, the ability to identify gene additions to the inferred strains already has potential to lead to important discoveries in microbiome research.

Chapter V

Array Testing for Prevalence Estimation

'We shouldn't feel disappointed that the march of understanding removes an element of mystery from the world; we should be appreciative of how much there is to know and the endless variety of ways in which our sensible universe continues to surprise us. The very fact that our world is comprehensible should fill us with wonder and delight. The world is not magic – and that's the most magical thing about it.'

– Sean Carroll

A special case of STD testing designs[120] is array testing (illustrated in Figure 1.4). Importantly, for this simplified design some analytical considerations can be made about the different possibilities for arriving at a given test result that will allow the estimation of prevalence from the array testing design using far fewer tests than when using individual testing. Group testing has previously been used for prevalence estimation, and array testing has been considered for this purpose before, even if only to a limited extent[145,146,147]. Importantly, to the best of my knowledge no fully analytical approach for the estimation of prevalence for array testing has been developed, nor has an exact derivation of the likelihood function been performed. Additionally, while I do not claim that the presented approach using array testing is the optimal group testing design for prevalence estimation, array testing has benefits in practice, such as easy set up and low testing time, as no retesting is required. Additionally, it provides the opportunity of identifying positive samples. Here, it is important to keep in mind ethical and practical consideration about returning false positive identifications. However, array testing does allow retesting if required and array testing can be applied if there is a need for surveillance testing. In this chapter, I lay out an analytical approach for obtaining the probability of a range of prevalence rates using array testing, as well as the maximum likelihood estimate for

the prevalence, and compare the obtained prevalence estimates to individual testing using the same number of tests.

Methods

Here, I present an estimator based on the number of positive row and column tests. In a different approach one could use a decoder and base prevalence estimates on the number of positive samples estimated by this decoder. However, approached naively this will lead to an overestimation of the prevalence, given the increased false positive rate using array testing. This is even more true when we attempt to compensate for sensitivity and specificity in the decoder, for example by using NCOMP, a variant of COMP that declares a sample positive if all but one tests containing it are positive, as this substantially increases the false positive rate (see Appendix C for a derivation). In contrast, an approach based on the number of positive rows and columns allows us to determine the exact probability of any number of positives.

Various considerations need to be made when estimating prevalence, such as the choice of prior as well as the test sensitivity and specificity. I will first derive an estimate for the prevalence assuming perfect specificity and sensitivity, before moving on to error-prone tests. Throughout it is assumed that each group testing design has the same size n, with n being the number of individual samples tested. As the samples are arranged in a square array the number of row tests is $\sqrt{n}$, as is the number of column tests. Therefore, we have a total of $2\sqrt{n}$ group tests. In Bayesian terms, when estimating a prevelance p we aim to estimate the posterior distribution $P(p|data)$. In the case of array testing, the data is the number of positive row tests r and positive column tests c for m rounds of group testing. This removes the need for decoding the actual positive samples. I denote the results for each round of group testing as $(r_i, c_i), 1 \leq i \leq m$, and all results together simply as $(r, c) = \{(r_1, c_1), (r_2, c_2), \ldots, (r_m, c_m)\}$. By Bayes theorem

$$P(p|(r,c)) = \frac{P((r,c)|p)P(p)}{\int P((r,c)|p)P(p)dp}. \tag{5.1}$$

Various choices for the prior on p are reasonable and can be informed by previous knowledge. For the current work a simple uniform prior across all prevalence rates is assumed. It will shortly become apparent that the integral is infeasible. Hence, it is approximated via a summation over sufficiently many prevalence rates. In our case all values of $0 \leq p \leq 1$ in increments of 0.001. The uniform prior is then replaced by $P(p) = 1/1001$ for all p. It remains to find an expression for $P((r,c)|p)$.

I will first investigate each group testing run individually. To obtain an expression for $P((r_i, c_i)|p)$ it will be

helpful to introduce the number of positive samples (or defectives as often referred to in the literature) d. We obtain the expression

$$P((r_t, c_t)|p) = \sum_{d=0}^{n} P((r_t, c_t)|d)P(d|p). \tag{5.2}$$

Here, $P(d|p)$ is the probability of observing d positive individuals when sampling n total individuals given the prevalence p, which is given by the binomial distribution for n tested samples. We can then determine $P((r_t, c_t)|d)$ as follows. We first need to determine the number of ways d positive samples can result in r_t positive rows and c_t positive columns in an n sample array testing design, denoted by $k_{r_i,c_i,d}$. The number of positive rows and columns gives us the maximum number of samples that are potentially positive, $r_t c_t$. Assuming d positive samples, we have $\binom{r_i c_i}{d}$ ways of choosing them among the potential positives. However, some of the choices would lead to fewer positive row or column tests than observed. For example, if all positives were arranged in a single row, we could not explain an observation with two or more positive row tests. We can proceed in a recursive manner to discount these possibilities by subtracting from $\binom{r_i c_i}{d}$ the number of ways we could have created fewer positive tests with the same number of positives, i.e.[1]

$$k_{r_i,c_i,d} = \binom{r_t c_t}{d} - \sum_{a=0}^{r_i-1}\sum_{b=0}^{c_i-1} \binom{r_t}{a}\binom{c_t}{b} k_{a,b,d} - \binom{r_t}{r_t-1} k_{r_i-1,c_i,d} - \binom{c_t}{c_t-1} k_{r_i,c_i-1,d}, \tag{5.3}$$

where

$$k_{0,0,d} = k_{1,0,d} = k_{0,1,d} = 0. \tag{5.4}$$

Note that we introduced a factor accounting for the possible ways to choose a out of r_t rows and b out of c_t columns, i.e. how many ways are there to choose fewer rows and columns than we observed. Additionally, to arrive at $P((r_t, c_t)|d)$ we need to estimate the fraction of possible configurations of d positive samples among all potential positive samples and the number of ways of choosing r_t rows and c_t columns out of $t = \sqrt{n}$ tests. Specifically, we have $\binom{t}{r_i}$ ways of choosing r_t rows out of the t row tests, and similarly we have $\binom{t}{c_i}$ ways of choosing c_t columns out of the t column tests. This needs to be taken into account, as we have previously only determined the number of ways d positive samples can be distributed in r_t and c_t positive row and column tests. However, we could choose several sets of rows and columns and need to account for the number of possible ways to do this. Therefore,

$$P((r_t, c_t)|d) = \binom{t}{r_t}\binom{t}{c_t}\frac{k_{r_i,c_i,d}}{\binom{n}{d}}. \tag{5.5}$$

Having an expression for $P((r_t, c_t)|p)$ it is straightforward to generalize to all observations (r, c) as all

[1] This is equivalent to a different expression derived by Carlitz et al. [206] and further analyzed by Riordan and Stein [207] for the arrangements of d pieces on a r_i by c_i chessboard with no rows or columns empty.

$P((r_i, c_i)|p)$ are conditionally independent given the prevalence p.

$$P((r,c)|p) = \prod_{i=1}^{m} P((r_i, c_i)|p). \tag{5.6}$$

With this in hand we can then calculate the posterior $P(p|(r,c))$ across all prevalence rates we are interested in.

Imperfect tests

Tests used in infectious disease testing are necessarily imperfect. We have to deal with both the sensitivity (S_e, the probability that a test is positive given that the sample is positive) and specificity (S_p, the probability that a test is negative given that the sample is negative), both of which are assumed to be known. These characteristics remain high for group testing[115,116,117,118]. However, they are not perfect, and hence need to be incorporated into the estimation. I first deal with sensitivity, again aiming to estimate $P((r_i, c_i)|d)$. Here, we need to account for the possibility that given a number of positives d should with perfect testing result in a certain number of positive test results (a, b), but some of the tests actually turn out negative such that we observe (r, c) instead. We can calculate the probability of $a = r_i + x, b = c_i + y, x, y \geq 0$ positive test outcomes for d positives and x and y false negatives. We simply determine the number of possibilities for obtaining (a, b) positive tests as before and incorporate the probability for $r_i + c_i$ tests of the $a + b$ tests turning out positive, i.e.

$$P((r_i, c_i)|(a,b),d) = \binom{a+b}{r_i+c_i} S_e^{r_i+c_i}(1-S_e)^{a+b-r_i-c_i} \binom{t}{a}\binom{t}{b} \frac{k_{a,b,d}}{\binom{n}{d}}, \tag{5.7}$$

We can then again consider all possible values of d and possible test outcomes (at most d for rows and columns, respectively; this is the case when the d samples are arranged diagonally). We can then simply sum over all d as

$$P((r_i, c_i)|p) = \sum_{d=0}^{n} \sum_{a=r_i}^{\min(d,t)} \sum_{b=c_i}^{\min(d,t)} P((r_i, c_i)|(a,b),d) P(d|p). \tag{5.8}$$

We again obtain the likelihood

$$P((r,c)|p) = \prod_{i=1}^{m} P((r_i, c_i)|p). \tag{5.9}$$

For incorporating specificity we proceed analogously to above, but for observing more positive tests than

would have been caused by d positives. It is straightforward to see that

$$P((r_t, c_t)|(a,b),d) = \binom{2t-a-b}{(r_t-a)+(c_t-b)}(1-S_p)^{(r_i-a)+(c_i-b)}S_p^{2t-r_i-c_i}\binom{t}{a}\binom{t}{b}\frac{k_{a,b,d}}{\binom{n}{d}} \tag{5.10}$$

and

$$P((r_t, c_t)|p) = \sum_{d=0}^{n}\sum_{a=0}^{r_i}\sum_{b=0}^{c_i} P((r_t, c_t)|(a,b),d)P(d|p). \tag{5.11}$$

Now we need to incorporate both sensitivity and specificity. For this we can proceed in a similar manner, however, the sum of false positive and false negative identifications needs to equal the deviation of the test result (a,b) under d positives from the observed test results (r_t, c_t). We introduce the error e as the number of false positive rows and the error f for the number of false positive columns. Then given (a,b), (r_t, c_t), e, and f the number of false negative rows must be $e - r_t + b$ and the number of false negative columns $f - c_t + b$. We then need to simply account for all possible number of false positive, which can at most be $a - r_t$ and $b - c_t$, respectively, i.e. if all a are false negative, we can have at most r_t false positives to observe r_t positives. We then sum over all possibilities to observe r_t positive rows and c_t positive columns as

$$\begin{aligned}
P((r_t, c_t)|p) = &\sum_{d=0}^{n}\sum_{a=0}^{\min(d,t)}\sum_{b=0}^{\min(d,t)}\sum_{e=0}^{a-r_i}\sum_{f=0}^{b-c_i} \\
&\binom{2t-a-b}{e+f}(1-S_p)^{e+f}S_p^{2t-a-b-e-f} \\
&\binom{a+b}{(e+a-r_t)+(f+b-c_t)}(1-S_e)^{(e+a-r_i)+(f+b-c_i)}S_e^{r_i-e+c_i-f} \\
&\binom{t}{a}\binom{t}{b}\frac{k_{a,b,d}}{\binom{n}{d}}P(d|p).
\end{aligned}$$

Implementation

When implementing the above estimator some adjustments can be made to decrease the steps of computation necessary. First of all, we can pre-compute $k_{a,b,d}$ for the design size worked on if more than one group testing round is evaluated. Similarly, we can store previously encountered test outcomes (as well as their symmetric counterpart with rows and columns exchanged) to avoid having to compute them multiple times. Secondly, we can skip over several steps of the computation, as we know they will be equal to 0. If both $S_e = 1$ and $S_p = 1$ the formula simplifies to perfect tests. If $S_e = 1$, we can drop all iterations for which $e - i + a + f - j + b \neq 0$, and similarly for $S_p = 1$ and $e + f \neq 0$. The binomial coefficients that are being evaluated may get very large, especially when summed over many iterations. However, using the default 64 bit integers in Python I did not run into any issues and hence, did not implement any approaches to mitigate this. It did, however, preclude

us from using a sampling approach for the posterior estimation such as using HMC and implementing such an approach would require further research. Similarly, future research may address implementation in a different language or optimizing computation by performing calculations in log-space. We can also ignore the evidence in the formulation, $\int P((r,c)|p)P(p)dp$, which serves simply as a scaling term, but will not affect the overall shape of the posterior. This will then enable the determination of the maximum likelihood estimate (MLE) for the prevalence. Specifically, for $n = 256$, impractical run times were observed and therefore MLE estimation was implemented for this group testing size. Under the assumption that the likelihood is locally monotonous with a singular mode we can start by considering an initially reasonable guess for the prevalence. First, I chose the largest prevalence p to a precision of $1/1000$ not bigger than

$$\frac{\frac{1}{m}\sum_{i=1}^{m} r_i c_i}{4n}. \tag{5.12}$$

which empirically proved to be a good initial estimate. A step size $\alpha = 0.001$ can then be picked to compute the likelihood of p, $p+\alpha$ and $p-\alpha$. The highest one of these will be the new estimate of p. If the likelihood of p is the highest, a maximum is obtained, which is by our assumptions the MLE and the search can be terminated.

Individual Testing

For individual testing we can utilize a straightforward estimator, similar to the one described above. For a given sensitivity and specificity the likelihood is

$$P(x|p) = \sum_{d=0}^{n}\sum_{e=0}^{d-x}\binom{n-d}{e}(1-S_p)^e S_p^{2t-d-e}\binom{d}{e+d-x}(1-S_e)^{e+d-x}S_e^{x-e}P(d|p), \tag{5.13}$$

with x being the observed number of positive samples. The rest of the formulation is analogous to the one presented for array testing, with the test results (r,c) being replaced by x.

Extension to higher disjunct designs

One might want to extend the above to higher disjunct designs. However, a quick analysis of a 2-disjunct STD designs shows that the same approach cannot work. Deriving the above exact expression for a given number of positive column, row, and diagonal (bottom left to top right, as in Figure 1.5) tests fails as can

be seen by the two possible configurations (for more than 3 rows and columns)

$$\begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix} \text{ and } \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} \tag{5.14}$$

both leading to 3 positive row, column, and diagonal tests, but with a different number of total possible positive samples (6 and 5, respectively). Hence, this approach would require decoding each possible test outcome, which it is not straightforward to derive an analytical expression for.

Results

To investigate the performance of array testing for prevalence estimation several simulations were run. First, let us compare performance to that of individual testing, assuming a fixed number of 360 tests. For the group testing design I chose $n = 81$, as this will result in a group size of $t = 9$, which will still deliver reliable test results while at the same time greatly reducing the required number of tests. With 360 tests, 20 rounds of group testing can be run, equivalent to 1620 tested samples. Performance was tested at prevalence rates of 0.001, 0.0025, 0.005, 0.01, 0.025, 0.05, and 0.1. For each prevalence level 100 simulations were run. I generated all 1620 samples for each round, determining sample positivity i.i.d. given the prevalence. The first 360 samples generated constituted the individual samples. False positives and false negatives for individual testing were determined i.i.d. given sensitivity and specificity. For the group testing approach the group test outcomes were computed first and then the false positive and false negative tests were determined i.i.d. at the level of the tests. I tested four pairs of values for sensitivity and specificity:

1. $S_e = 1, S_p = 1$
2. $S_e = 0.99, S_p = 0.99$
3. $S_e = 0.95, S_p = 0.99$
4. $S_e = 0.95, S_p = 0.95$

The results are shown in Figure 5.1. Throughout, the group testing design shows better estimates, though the advantage of group testing decreases as prevalence increases. Especially at low prevalence rates individual testing shows large errors in many of the simulations. One may also consider that group testing might

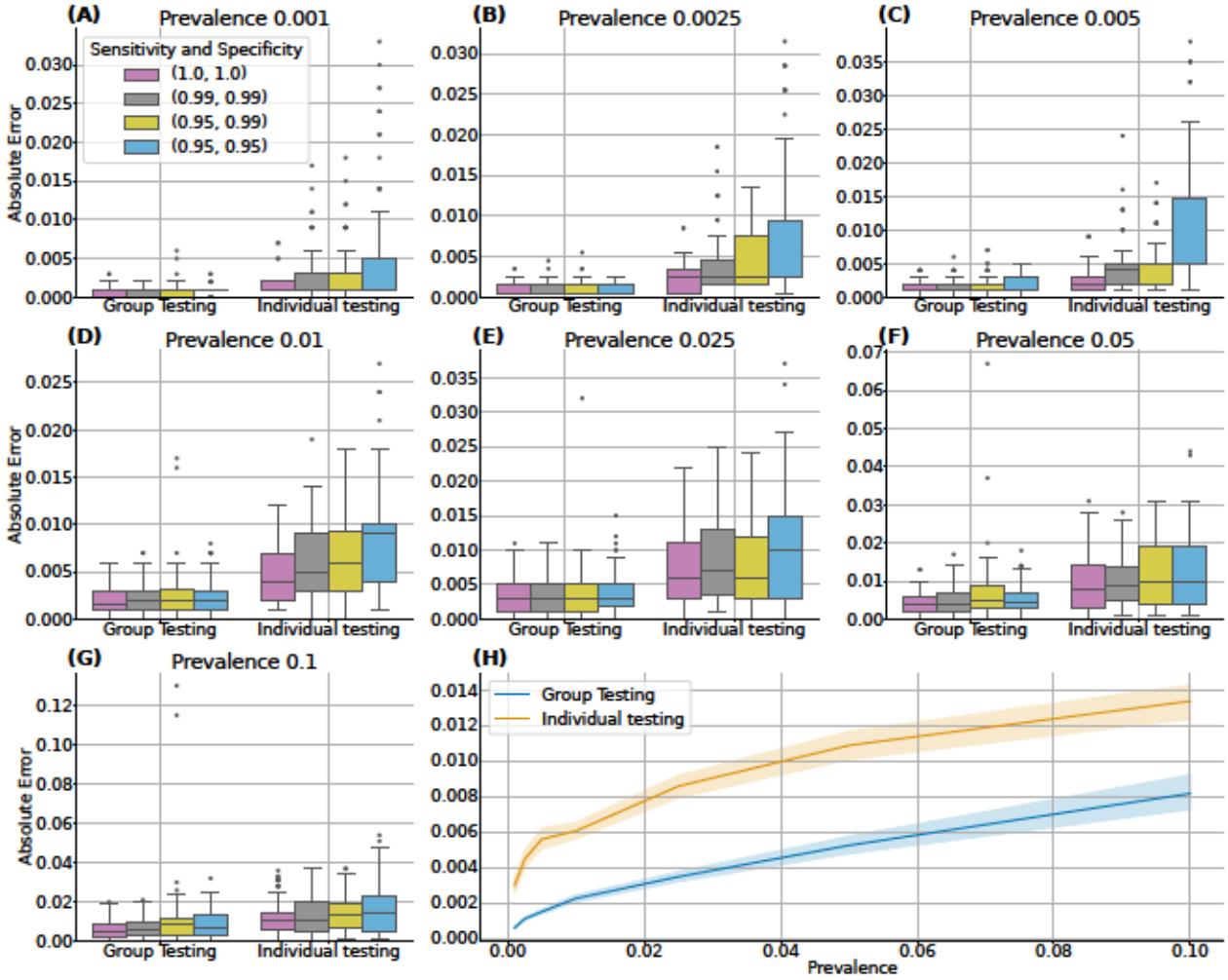


Figure 5.1: (A-G) Absolute error of estimated prevalence from true prevalence for each true prevalence for group size 9 array testing and individual testing with a total of 360 tests at varying sensitivities and specificities. Line indicates median, edges of box the 25% and 75% quartile, and whiskers extend to include the datapoint furthest from the median that is within 1.5 times the interquartile range. Dots are outliers. **(H)** Mean absolute error between true and estimated prevalence for array testing and individual testing aggregated across different sensitivities and specificities. Line shows the mean and shaded area the 95% confidence interval.

decrease the sensitivity of the tests used. Even considering this we can see that the variability in results for group testing is lower even when comparing $S_e = 0.95, S_p = 0.95$ to perfect sensitivity and specificity for individual testing. Interestingly, some outliers are observed for group testing with $S_e = 0.95, S_p = 0.99$. This may be due to the imbalanced test characteristics. However the specific reasons for this are not clear and require further investigation in the future. Lastly, it is worth noting that the chosen array testing design is only guaranteed to produce no false positive inferences (assuming perfect tests) for 1 out of 81 samples (1.2%). For more than one positive sample the probability of obtaining no false positives decreases sharply (Figure C1). Nonetheless, estimating prevalence from group tests using the presented estimator provides highly accurate results even at 10% prevalence, which corresponds to observing 8 positive samples per group

testing round on average, far beyond the design limitations of the underlying group testing scheme. This shows the impressive robustness of the present approach to prevalence estimation in different scenarios.

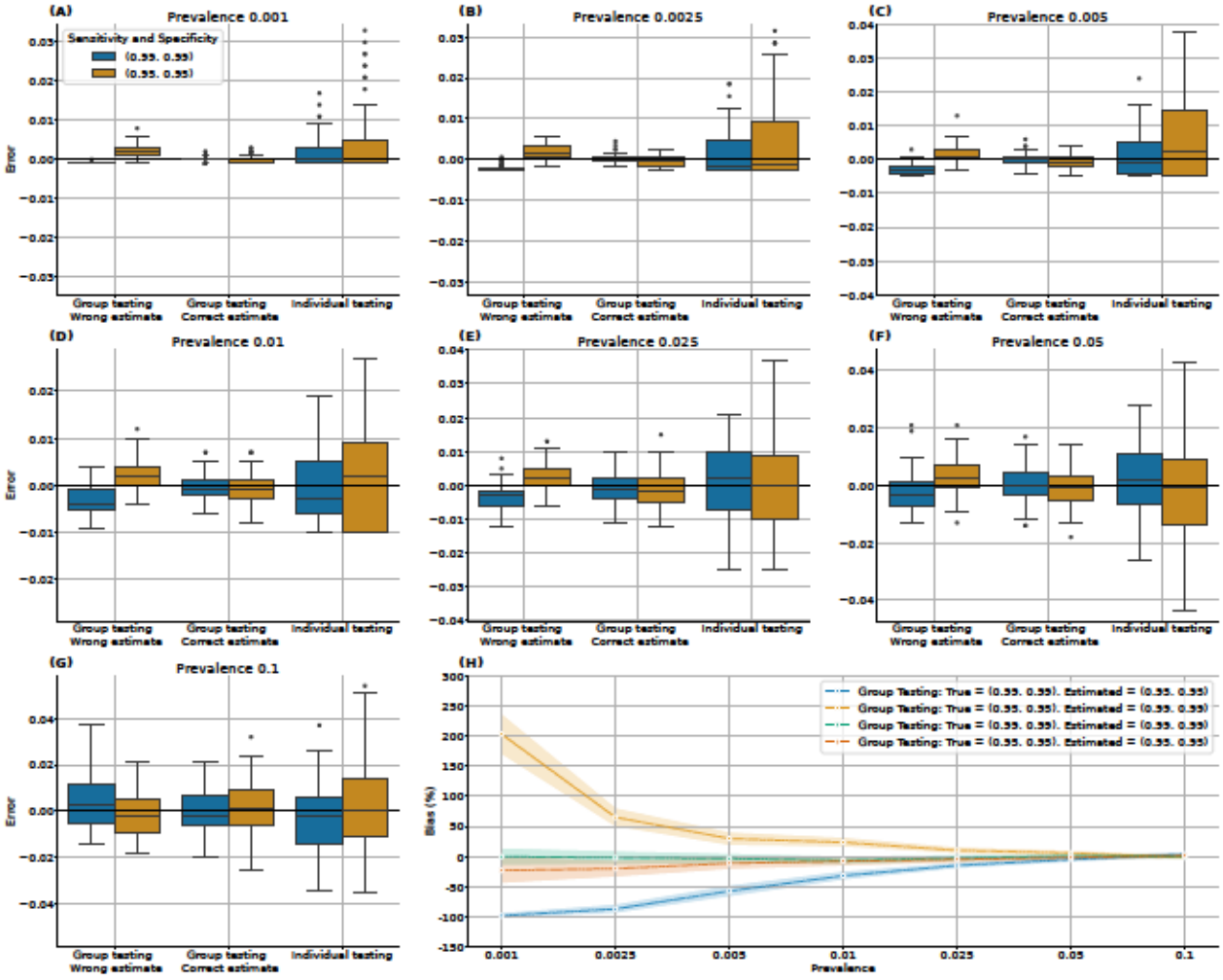


Figure 5.2: (A-G) Absolute error of estimated prevalence for group testing with wrong sensitivity and specificity estimate, as well as group testing and individual testing with correct sensitivity and specificity estimate. For the wrong estimates, estimates were $S_e = 0.95, S_p = 0.95$ for the true values $S_e = 0.99, S_p = 0.99$ and $S_e = 0.99, S_p = 0.99$ for $S_e = 0.95, S_p = 0.95$. Line indicates median, edges of box the 25% and 75% quartile, and whiskers extend to include the datapoint furthest from the median that is within 1.5 times the interquartile range. Dots are outliers. **(H)** Percent bias of group testing estimators with correct and incorrect sensitivity and specificity estimates for different underlying prevalence rates. Note that the maximum negative bias is 100% due to the lower limit of a prevalence of 0. Line shows the mean and shaded area the 95% confidence interval.

While test characteristics for PCR tests for many infectious diseases can be estimated well, in practice, it is possible to assume wrong test characteristics. To investigate this, I reran the simulations for $S_p = 0.99, S_e = 0.99$, assuming a sensitivity and specificity of 0.95. Similarly, I reran the simulations for $S_p = 0.95, S_e = 0.95$, assuming sensitivity and specificity at 0.99. The results for this simulation show the expected outcomes (Figure 5.2). When sensitivity and specificity are underestimated, then so is the prevalence, and similarly for the case when sensitivity and specificity are overestimated so is the prevalence, i.e. the estimator shows

a bias. This bias is reduced, or even absent, at higher prevalence rates. Additionally, for the number of tests presented here, the absolute deviation of group testing is still lower than for individual testing, though at a higher number of tests the deviation of individual testing will be reduced and the bias will be of primary concern. However, the shown prevalence estimates for individual testing are obtained assuming the correct sensitivity and specificity and results from individual testing will suffer a similar bias under misestimated test characteristics. Even with misestimation of the test characteristics, while a bias is observed, the estimates are still within a reasonable range of the true prevalence.

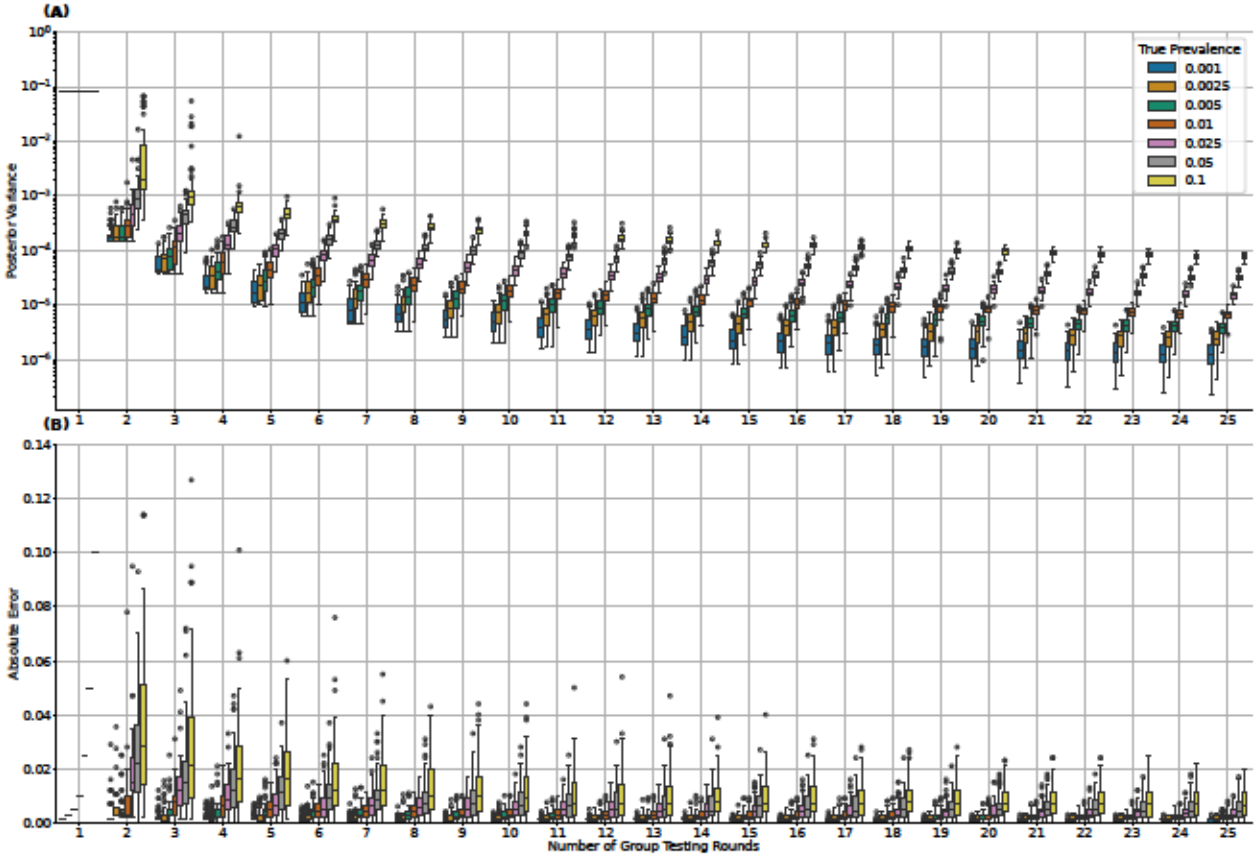


Figure 5.3: Group testing performance across different prevalence rates with different number of group testing rounds. Each group testing round was one array testing design with 18 tests and 81 samples. **(A)** Variance of the posterior distribution. **(B)** Absolute error from the true prevalence. In both plots line indicates median, edges of box the 25% and 75% quartile, and whiskers extend to include the data point furthest from the median that is within 1.5 times the interquartile range. Dots are outliers.

Furthermore, the estimator provides an estimate of the posterior distribution of the prevalence instead of simply a point estimate. To provide an idea of the evidence or number of group testing rounds required for a confident estimate I generated 25 group testing rounds, but ran the computation 25 times starting with one included group testing round and iteratively including one more data point at a time. I again repeated this procedure 100 times for each prevalence. Then, the variance around the mean for each run of the

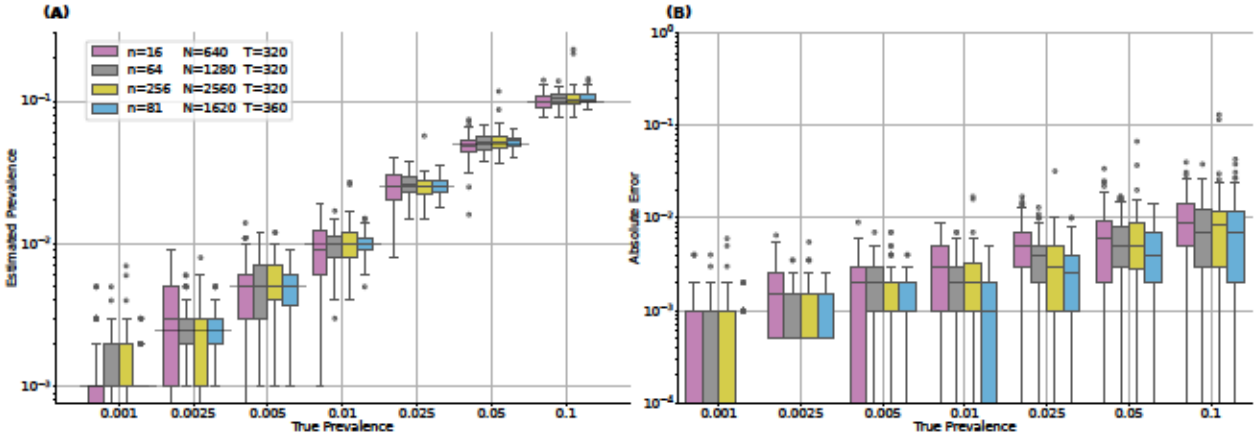


Figure 5.4: Comparison of prevalence estimation across different array testing designs of size 16, 64, and 256 with 320 tests in total. N indicates total number of samples tested. and T the number of tests. Also shown is $N = 81, T = 360, S_e = 0.95, S_p = 0.99$ for comparison of previous results. **(A)** Estimated versus true prevalence. **(B)** Absolute error between the estimated and true prevalence. In both plot line indicates median, edges of box the 25% and 75% quartile, and whiskers extend to include the datapoint furthest from the median that is within 1.5 times the interquartile range. Dots are outliers.

prevalence estimator was computed (Figure 5.3A). The variance reduces quickly with the number of included group testing rounds indicating that even with few group testing rounds we can obtain confident estimates. This is important to know, as a case may occur where group testing is implemented as prevalence estimation strategy, but fewer than expected samples are collected. Notably, variance increases with prevalence, however this is to be expected given that a large part of the array testing formalism relies on binomial distributions, which has maximum variance at a parameter of 0.5. Another measure for the accuracy of the estimates after a given number of group testing round is provided by the deviation to the true prevalence. The absolute deviation decreases quickly, with the largest decrease observed for the first few incorporated group testing rounds (Figure 5.3B). Importantly, the number of outliers decreases notably with more group testing rounds included, indicating that for 20 rounds onward (or 1620 tested samples) the estimates cluster closely around the true prevalence and can be trusted reliably. However, even at only 8 testing rounds 75% of the time the deviation at a true prevalence of 0.1 is less than 0.02 (Median = 0.01). For a true prevalence of 0.01 a median deviation of 0.003 with 10 included rounds is observed, showing a maximum deviation of 0.01 and thereby providing reliable estimates with only 810 samples tested and 180 tests used.

Lastly, a practical group testing design needs to be determined. To this end, I compared three different designs, one with 16 samples, one with 64, and one with 256, corresponding to group sizes of 4, 8, and 16. Here, I decided to give a test budget of 320 tests, i.e. 40, 20, and 10 testing rounds, testing a total of 640, 1280, and 2560 samples, respectively. However, different test characteristics for the different designs were assumed. Available research suggests a conservative estimate of around 90% sensitivity for a group size of

16[115,116,117,118]. Based on this, I chose a sensitivity of 0.99 for a group size of 4, and 0.95 for a group size of 8. 100 simulations at each prevalence were performed (Figure 5.4, also showing $n = 81$ design for comparison). For $n = 256$, only MLE estimates were computed due to extensive processing time. The estimates do improve going from $n = 16$ to $n = 64$, i.e. doubling the total amount of tested people (Figure 5.4). However, the improvements are smaller than expected, especially at higher prevalence rates. This is likely due to the inherent limitations of array testing at higher prevalence rates. Independent of the specific array testing size, a similar number of positive tests for the same number of positive samples will be observed. However, for larger designs, the number of positive samples per design will be higher. This will lead to many more possible underlying positive sample configurations being able to cause the observed test results. In turn, prevalence is potentially more likely to be overestimated, though this is balanced by the increase in tested samples. In line with this argument no increase in accuracy is observed for $n = 256$, even while again doubling the number of tested samples. However, another consideration that needs to be taken into account is that a large part of the population should be tested to obtain representative results while using as few tests as possible. Especially at low prevalence rates this may be important. Therefore, especially at low prevalence rates larger group testing designs will be preferred if accurate and reliable estimates for the test sensitivity and specificity can be obtained.

Discussion

I presented the analytical derivation of the likelihood function for prevalence estimation using array testing. To the best of my knowledge, no previous study has derived the likelihood function for array testing. Instead, simulations for estimation of prevalence rates were used for array testing designs. I show the feasibility of array testing as a means of prevalence estimation. With the $n = 81$ array design we can save almost 80% of tests compared to individual testing. I furthermore showed that with the same number of tests the estimator consistently outperforms individual testing, even when sensitivity and specificity are lower. This is important, as, while group testing has been shown to not substantially impact sensitivity[115,116,117], some researchers still report an effect that needs to be taken into account[118]. Furthermore, I show that the error in the prevalence estimate declines quickly with the inclusion of more group testing rounds. This will also make more small scale prevalence estimations, such as in local communities, feasible. Lastly, prevalence estimation is relatively unaffected by the size of the group testing design.

Several avenues for future work exist. First of all, I assumed that the sensitivity and specificity of the test are known. While this will be often the case in practice, it may be possible to estimate sensitivity and specificity

together with the prevalence within a Bayesian framework. This likely presents a fairly simple extension of the model, however, the calculation may be more readily achievable by implementing an approximation approach, such as sampling of the posterior via Markov chain Monte Carlo approaches. Furthermore, it will be worthwhile to determine the ability to track prevalence during an outbreak. This could be achieved either by simulating a pandemic akin to the study performed by Cleary et al. [208], or by subsampling test results from real data. This may give insights into the temporal dependence between estimates, and the ability of the framework to account for quick changes in prevalence rates. Such a study could also investigate the possibility of including retesting of potential positives to prevent the reporting of false positive test results to patients and its implications, and evaluate the best available group testing strategy for prevalence estimation. Extending the presented approach to multiplex testing by including co-infection probabilities is another avenue for future research. Previously, Warasi et al. [150] estimated prevalence rates in a multiplex scenario. While the approach presented here will work for multiplexed testing and may provide good results, Warasi et al. [150] also consider the probability of co-infection. Hence, their approach may be more suitable for some situations, though their method is not specialized for array testing and this needs to be evaluated in more detail in future research.

In conclusion, I demonstrate the potential of array testing for prevalence estimation. Its ease of setup without need for retesting makes it an ideal candidate for prevalence estimation, especially at low prevalence rates, where it clearly outperforms individual testing. To facilitate uptake in the research and clinical community, the estimator was made readily available as Python script (`https://github.com/lahunruh/array_prevalence`). In resource constrained settings the savings in tests will enable more accurate estimates. Estimating prevalence in these settings is of special importance, as more accurate estimates will facilitate the implementation of the correct measures to curb the spread of an infectious disease. During the Covid-19 pandemic, this was apparent across Africa, where the WHO indicated testing goal of 10 per 10000 people could not be met in many countries, inhibiting accurate surveillance and estimation of case numbers [7]. Group testing can address this issue going forward, be it for Covid-19 or other infectious diseases. There are however still a few hurdles. In general, the ethical implications of using non-adaptive group testing for identification of positive samples need to be discussed in the research community and beyond, as effective surveillance will not only deliver a prevalence estimate, but at the same time deliver test outcomes to the sampled population. Furthermore, implementation of group testing in laboratories needs to be made straightforward. This requirement makes array testing a good choice among non-adaptive group testing designs, as the grid structure is straight forward to understand and easy to set up in laboratories. Altogether, my research adds to a body of evidence clearly showing the need for policymakers to consider widespread

implementation of group testing, with the research presented here providing a strong argument for array testing to be a part of the conversation.

Chapter VI

Preliminary Study on non-binary Group Testing for Respiratory Infections

'We should resist such inertial thinking; indeed, we should urge its opposite – deep time as a radical perspective, provoking us to action not apathy. For to think in deep time can be a means not of escaping our troubled present, but rather of re-imagining it; countermanding its quick greeds and furies with older, slower stories of making and unmaking. At its best, a deep time awareness might help us see ourselves as part of a web of gift, inheritance and legacy stretching over millions of years past and millions to come, bringing us to consider what we are leaving behind for the epochs and beings that will follow us.'

– Robert Macfarlane

Here, I present a Bayesian approach combined with simulated annealing for decoding of singleplexed PCR group testing results, utilizing the quantitative nature of PCR test results. This approach aims to improve decoding over simply using binary test outcomes. Additionally, the research aims to be a basis for addressing multiplexed PCR testing for multiple respiratory infections in the future. While the current approach does not yet incorporate co-infection probabilities across different diseases, I am confident that the Bayesian model provides a potential basis that could incorporate co-infection probabilities with further development. The decoder already shows reliable results in singleplexing testing. This was validated both in silico and, together with Michael A. Crone and Paul S. Freemont, in vitro using virus-like particles of severe acute respiratory syndrome coronavirus (SARS-CoV-2). SARS-CoV-2 is a prime target of group testing validation, with SARS-CoV-2 being the causative agent of the Covid-19 pandemic. SARS-CoV-2 is a single-stranded RNA virus causing respiratory infection[209]. Since its emergence in 2019 over 7 million deaths due to Covid-19 have been reported[210]. Surveillance has been key during the Covid-19 pandemic and greatly expanded our knowledge

about the disease and its spread [211]. Especially low and middle income countries have continuously struggled with limited healthcare capacities [212] and continued surveillance will help tackle the spread of SARS-CoV-2 going forward. Group testing is a great option to address testing in settings with limited resources. Hence, performing a group testing study on SARS-CoV-2 is promising. Notably, we also conducted multiplexed experiments in vitro, which to the best of my knowledge is a first in the research literature. This validation is essential for applying group testing in a multiplexed setting in practice. However, for the discussion of the novel decoder, the multiplexed setting is not relevant and the experiments are not described herein. The interested reader can refer to Unruh et al. [1].

Methods

Reminding ourselves briefly of the formalism of group testing, we have n positive or negative samples X_j. We want to infer the binary status (positive or negative) of these samples. Each sample is split into k parts and placed into k group tests. The group tests a sample is placed in is determined by a design matrix $M \in \{0,1\}^{m \times n}$, with m the total number of group tests. If $M_{i,j} = 1$, sample X_j is placed into test i. Conversely, if $M_{i,j} = 0$ sample X_j is not placed into test i. The test outcomes are denoted by the vector Y with Y_i being the test outcome of test i, determined in binary group testing via

$$Y_i = \bigvee_{j \in [1,n]} M_{i,j} \wedge X_j. \tag{6.1}$$

According to this, if at least one sample X_j placed in test i is positive (1), the test will be positive and $Y_i = 1$. Given M and Y the goal of a decoder is to infer the status of each sample, positive or negative. For the present research, the d-disjunct shifted transversal design [120] (STD) was chosen to determine the design matrix M. The shifted transversal design expands on the simple idea of array testing described in Chapter I. Specifically, we use a version of STD described in Täufer [213] called multipool testing. While I will refer to STD throughout, as it is the more general definition of the testing design, the definition given by Täufer [213] is illustrative of some key properties. The definition, reproduced verbatim (with notation adapted to the one used in this thesis) from Täufer [213], is

Definition VI.1. *(Multipools) Let a population $(X_1, \ldots, X_n)$ of size n, a pool size g, and a multiplicity k be given, and assume that nk is a multiple of g. We call a collection of subsets/pools of $\{X_1, \ldots, X_n\}$ an (n, g, k)-multipool, or simply multipool, if all the following three conditions hold:*

(M1) Every pool consists of exactly g elements.

(M2) Every sample X_j is contained in exactly k pools.

(M3) For any two different samples X_{i_1}, X_{i_2} there exists at most one pool which contains both X_{i_1} and X_{i_2}.

The created designs will be $(k-1)$-disjunct. An STD design can then be constructed for $n = g^2$ and g a prime, but also for some other numbers of tested samples and disjunctness values, for example it is possible to create a $(81, 9, 4)$-multipool using the STD method. The actual construction can be achieved by arranging all samples in an $g \times g$ square and testing along rows, columns, and additionally $k-2$ slopes of different steepness, as illustrated in Figure 6.1. More formally, as laid out by Thierry-Mieg [120], we create M from the concatenation of k layers. Each layer is a matrix composed of n columns of size $g \times 1$ of the form

$$C_{i,j} = \pi_{s(i,j)} C_{0,0} \tag{6.2}$$

for $1 \leq j \leq k$ and $1 \leq i \leq n$, with

$$C_{0,0} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}. \tag{6.3}$$

$\pi_{s(i,j)}$ is the $s(i,j)$ times applied cyclic shift function identifying each element of $C_{0,0}$ at position i with a new position $i+1$ if $i < g$ and position 1 if $i = g$. To determine the number of shifts for each $C_{i,j}$ we have

$$s(i,j) = \sum_{c=0}^{2} j^c \left\lfloor \frac{i}{g^c} \right\rfloor \tag{6.4}$$

for $i < g$ and

$$s(n,j) = \left\lfloor \frac{i}{g^2} \right\rfloor. \tag{6.5}$$

Here, $\lfloor \frac{i}{g^2} \rfloor$ is the largest integer not bigger than $\frac{i}{g^2}$. From the g columns we create a $g \times n$ submatrix M_j for each $0 \leq j \leq k-1$. Stacking these matrices then gives us our final testing $kg \times n$ testing matrix M according to STD.

To illustrate this process, we can look at a small 2-disjunct design (i.e. $k = 3$) with $n = 9$ and $g = 3$. This

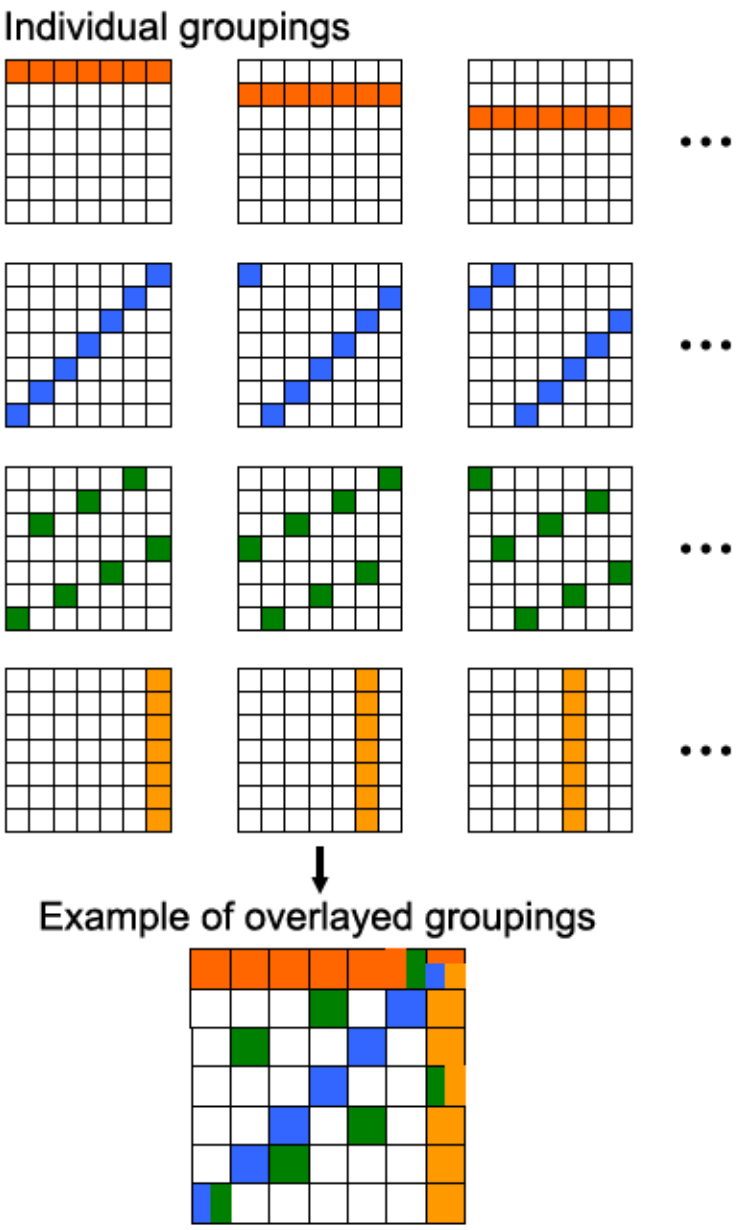


Figure 6.1: Illustration of a (49,7,4)-multipool. Each cell is one individual sample. Cells colored in one color indicate membership in the same group. Top row displays groupings along rows, with all samples in one row grouped into one group test (together, the row groupings correspond to the $g \times N$ submatrix of M for $j = 1$).Second row displays group test against the first diagonal and the third row group tests along the second diagonal. Note the action of the cyclic shift, which corresponds to the group test membership of the samples being shifted right, wrapping around to the left side of the array. The fourth row indicates grouping of samples into group tests along columns. Taking only row and column tests corresponds to array testing. Samples are grouped into one row, one column, and one of the diagonal tests each, illustrated in the example of overlaid groups. Here, each cell colored in the same color indicates the same test with multiple colors indicating membership in two or more of the illustrated tests. The other groups are created by shifting the shown groups horizontally (or vertically for the orange group), and cells wrapping around go to the first (left) column when they would pass beyond the last (right) column as illustrated in the first four rows of the figure.

means we need to create 3 submatrices. The first submatrix for $j = 0$ is created by cyclically shifting

$$C_{0,0} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}. \tag{6.6}$$

The first column stays as $C_{0,0}$. The second column $C_{1,0}$ has the cyclic shift function applied once, as $s(1,0) = \sum_{c=0}^{2} 0^c \lfloor \frac{1}{3^c} \rfloor = 1$. Similarly, $s(i,0) = i$, and when $i = g$ the cyclic shift function is applied g times and $C_{3,0} = C_{0,0}$. Therefore, applying the process is equivalent to applying the cyclic shift function $i \mod g$ times. By this process we obtain the submatrix

$$\begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}. \tag{6.7}$$

For the second submatrix ($j = 1$), $C_{0,1}$ again stays the same as $C_{0,0}$. For $C_{1,1}$ and $C_{2,1}$ we arrive at the same number of shifts as for $j = 0$. The first difference occurs at $i = 3$. Here, $s(3,1) = 1^0 \lfloor \frac{3}{3^0} \rfloor + 1^1 \lfloor \frac{3}{3^1} \rfloor + 1^2 \lfloor \frac{3}{3^2} \rfloor = 3 + 1 + 0 = 4$ (or equivalently $s(i,1) = (i \mod g) + 1$) and the column $C_{3,1}$ is given as

$$C_{0,0} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}. \tag{6.8}$$

We proceed in a similar manner. For $i \geq 6$, we effectively have $s(i,1) = (i \mod g) + 2$, so the three last columns are shifted twice compared to the first three columns. The total submatrix is then given as

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \end{bmatrix}. \tag{6.9}$$

By a similar process we obtain the columns for $j = 2$ and arrive at the complete example testing matrix

$$M = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}. \tag{6.10}$$

Here, it also becomes apparent that we need to choose n sufficiently large ($n > kg$) to achieve a saving in tests.

PCR test

In order to design a decoder that utilizes quantitative PCR test results, it is first necessary to define the nature of these results. Running a PCR test results in a Ct value, whose magnitude relates to the viral load (VL) in the tested group. Specifically, we assume that given the VL of the jth sample V_j the Ct value for the ith test is generated by

$$Y_i = K - \log_2 \left(B \sum_j \frac{M_{i,j} V_j}{g_i} \right) + error, \tag{6.11}$$

where K is a linear offset and B a scaling factor, both of which we estimated from data available in the study of Vogels et al.[124] (see in silico data generation). g_i is the size of group i, i.e. $g_i = \sum_{j=0}^{n} M_{i,j}$, and the error is a noise term necessitated by the fact that PCR test outcomes are stochastic. We assume that the error is drawn from a normal distribution $\mathcal{N}(0, e)$. If good laboratory data is available, one can also obtain an estimate ϵ for e. However, as this might not always be possible in practice we set $\epsilon = 0.5$ as an estimate in a realistic range (see Chapter I). Note the abuse of notation, where Y_i in Equation 6.1 referred to binary test outcomes, while here it corresponds to PCR test outcomes.

Simulating Group Testing Experiments

In order to tune and test the model, we need to generate simulated data. This requires a few considerations. First of all, we start with a population of n samples and determine a set number of positive samples at random. Then, we need to determine the VL V of each of the positive samples. For this, we adopt the approach from Cleary et al.[208]. Here, each positive individual starts of with a VL $V = 1$ at time 0, with the VL exponentially increasing up until time t_{max}, at which point it is at its maximum of 2^{α}. After this point, the VL decreases exponentially up until time t_{end}, at which point the VL attains it minimum of $V = 1$ again. Specifically, the VL V at time t is

$$V(t) = \begin{cases} 2^{\frac{a}{t_{max}}t} & \text{if } 0 \leq t \leq t_{max} \\ \frac{2^{\alpha}}{2^{\frac{a}{t_{end}-t_{max}}(t-t_{max})}} & \text{if } t_{max} < t \leq t_{end}. \end{cases} \tag{6.12}$$

We used the parameters presented in Cleary et al.[208], i.e. $a = 7.98$, $t_{max} = 3$, and $t_{end} = 17.62$. To obtain a VL for any sample we then pick a time point $0 < t < t_{end}$ uniformly at random. Notably, this time point t may be more restricted in a clinical sample, as people would often get tested while symptoms are showing. However, in surveillance testing the assumption of t being picked at random often holds. To convert these individual VLs to group testing Ct values we simply insert them into Equation 6.11, with the error being drawn from a normal distribution centered at 0 with standard deviation $e = 0.5$. We used $K = 37$ and $B = 100$ to generate our data, which we estimated from the data provided in Vogels et al.[124].

gtMAP decoder

Our overall goal is to estimate a status vector ω of length n. Each entry of ω corresponds to one sample, with entries 0 and 1 for negative and positive samples, respectively. To infer the status vector, we are given the testing matrix M and the group test results Y. Note that this problem is underspecified, meaning that in general there is more than one sensible solution for ω that could generate the observed Ct values. The aim is to construct a model to estimate ω that incorporates how individual VLs combined with the testing matrix and noise generate the observed test outcomes. In the model presented here, we aim to estimate ω via a simulated annealing procedure. At each step of the simulated annealing procedure (described further below) we evaluate one status vector ω. Given this status vector in which samples i for which $\omega_i = 1$ are said to be positive, we want to estimate individual Ct values for the positive samples (the Ct value that would have been obtained had we tested the sample individually instead of in a group) and evaluate how closely

the group test results estimated given these individual Ct values match the observed group testing results. We do this through maximum a posteriori (MAP) optimization.

To be specific, we aim to generate predicted Ct values for the group tests $\hat{Y}$ and compare them to the observed test results Y, and use the deviation between Y and $\hat{Y}$ as evaluation measure for the different possible states of the status vector ω. As evaluation measure the squared error is used

$$E = \sum_{t=1}^{t} (Y_t - \hat{Y}_t)^2. \tag{6.13}$$

To generate the estimated group test Ct values, Equation 6.11 is used. However, we replace the VL V_j of sample j with the individual Ct values Γ_j (the Ct value that would have been obtained had we tested the sample individually instead of in a group) of the samples, as these are estimated in our model. The individual Ct values can be transformed into VLs via $V_j = 2^{\Gamma_j}$. Therefore, the test outcome $\hat{Y}_t$ for test i is

$$\hat{Y}_t = \kappa - \log_2 \left(\beta \sum_{j=1}^{n} \frac{M_{t,j} 2^{\Gamma_j}}{g_t} \right), \tag{6.14}$$

where κ and β are estimates for K and B in Equation 6.11. How these parameters are obtained will be described below. This equation clearly states that we require the individual test outcomes Γ_j for each sample $0 \leq j \leq n$. These are obtained through a MAP optimization implemented in Stan using its interface RStan [214] in R.

The model implemented in Stan is given the data ω, M, and Y. We aim to maximize

$$P(\Gamma|Y) = P(Y|\Gamma)P(\Gamma), \tag{6.15}$$

with Γ the vector of individual Ct values. We define the likelihood $P(Y|\Gamma)$ as follows.

$$P(Y|\Gamma) = \prod_{t=0}^{t} P(Y_t|\hat{Y}_t) \tag{6.16}$$

with

$$\hat{Y}_t = \kappa - \log_2 \left(\beta \sum_{j=1}^{n} \frac{\omega_j M_{t,j} 2^{\kappa - \Gamma_j}}{g_t} \right). \tag{6.17}$$

Note the addition of ω_j compared to Equation 6.11. This is done to only consider positive samples in the group testing results, as negative samples will contribute no VL. However, $2^{\kappa - I_j}$ can never be 0, hence the inclusion of the binary indicator ω_j is required. Further, note that the individual Ct value I_j is subtracted

from the estimated Ct value baseline κ, as $2^{\kappa - I_j}$ which replaces the VL V in Equation 6.11. For any positive individual test result $I_j < \kappa$, this VL should be greater than 1. The likelihood is then obtained via a normal distribution

$$P(Y_i|\hat{Y}_i) \sim \mathcal{N}(Y_i|\hat{Y}_i, \epsilon), \tag{6.18}$$

where ϵ is a parameter that is provided to the model and an estimate for the error e around the Ct value determination in Equation 6.11. The prior over the individual Ct values I is given by

$$I_j \sim \mathcal{N}(\mu, \sigma), \tag{6.19}$$

for the j'th sample, where μ and σ can be estimated from clinical data or through a computational route (the latter was chosen here, see Parameter Tuning). Note that the prior is only added to the posterior density for samples with $\omega_j = 1$. A MAP optimization is performed using the `rstan::optimizing(stanmodel, data)` function, where `stanmodel` is the Stan implementation of the above model, and `data` is $M, Y, \omega, \mu, \sigma, \epsilon, \kappa$, and β. Note here, that it is also possible to estimate the hyperparameters $\mu, \sigma, \epsilon, \kappa$ as part of the MAP optimization. I investigated this possibility for the error ϵ with a prior $\epsilon \sim \mathcal{N}(0, 1)$, however, no notable difference to providing a fixed parameter was observed and the possibility of including these parameters in the estimation is left for future research.

Through the above MAP optimization we obtain estimates for the individual Ct values Γ given a status vector ω. We now need to consider how to estimate ω. As a first step, we permanently set all entries of ω to 0 that are definite negatives. Definite negatives are determined using COMP [121]. The remaining entries of ω are treated as potential positives and we aim to identify the true positives among them. Next, we can make use of the fact that we obtain Γ and thereby $\hat{Y}$ for any status vector through the MAP optimization. Hence, every status vector has an associated evaluation measure E (Equation 6.13). Through this we can estimate the fit of a single status vector ω via the above MAP optimization and an ideal status vector should minimize E. The question that remains is how to choose the status vectors that we want to evaluate. One avenue for choosing status vectors is to use simulated annealing (see Figure 6.2 for an illustration). Specifically, we start with a randomly initialized status vector of 0s and 1s. This status vector is then evaluated to obtain the current evaluation measure E_c. Then we propose a random change to the status vector (switching a 0 to a 1 among the potential positives, or vice versa) and evaluate the new status vector E_n. We can obtain the difference in evaluation measures as

$$\Delta E = E_n - E_c. \tag{6.20}$$

We then either accept or reject the new status vector according to an acceptance probability

$$P_{accept} = \begin{cases} 1 & \text{if } \Delta E \leq 0 \\ e^{-\Delta E/T(c)} & \text{otherwise,} \end{cases} \tag{6.21}$$

with $T(c)$ being the so-called temperature at the current temperature iteration c. In simulated annealing, this temperature determines the acceptance probability. We start of with a high temperature in the first temperature iteration $c = 1$ for several iterations over the status vector. This high temperature leads to the acceptance of nearly all proposed status vectors. In each temperature iteration c the temperature is then lowered according to a temperature decline function. This lower temperature leads to a lower acceptance probability of status vectors that would increase ΔE. In each temperature iteration we iterate over multiple proposed status vector changes - in our case 10 was chosen as the number of proposed changes iterated over - while the total number of temperature iterations was chosen as $c_{max} = 50$. These numbers of iterations were chosen to achieve a balance between performance and computational speed. We evaluated a range of frequently employed temperature decline functions with the best performing one being

$$T(\tau) = \frac{T_0}{1 + c^2\alpha}. \tag{6.22}$$

Here, T_0 is the temperature in the first temperature iteration and α is a parameter chosen such that we have $T(c_{max}) = 1$. To determine the initial temperature we evaluate 10 random changes from the initial status vector and then either double or halve the temperature until the acceptance rate for these changes is between 0.7 and 0.9.

Notably, the above optimization protocol has a random component (whether a new status vector is accepted or rejected according to the acceptance probability), and successive runs can provide different results. This is also due to the fact that the optimization landscape has multiple, potentially very similar, minima. To illustrate this, I want to give a simple example. Consider the 1-disjunct STD design (equivalent to array testing). Here, we could have a situation in which four different samples, arranged in a square have VLs of the type

$$\begin{bmatrix} a & 0 \\ 0 & a \end{bmatrix} \tag{6.23}$$

resulting in test outcomes $K - \log(a + 0)$ for the first row and column, and similarly $K - \log(0 + a)$ for the second row and column (here the equation for the Ct value of the tests is simplified to enable a clearer

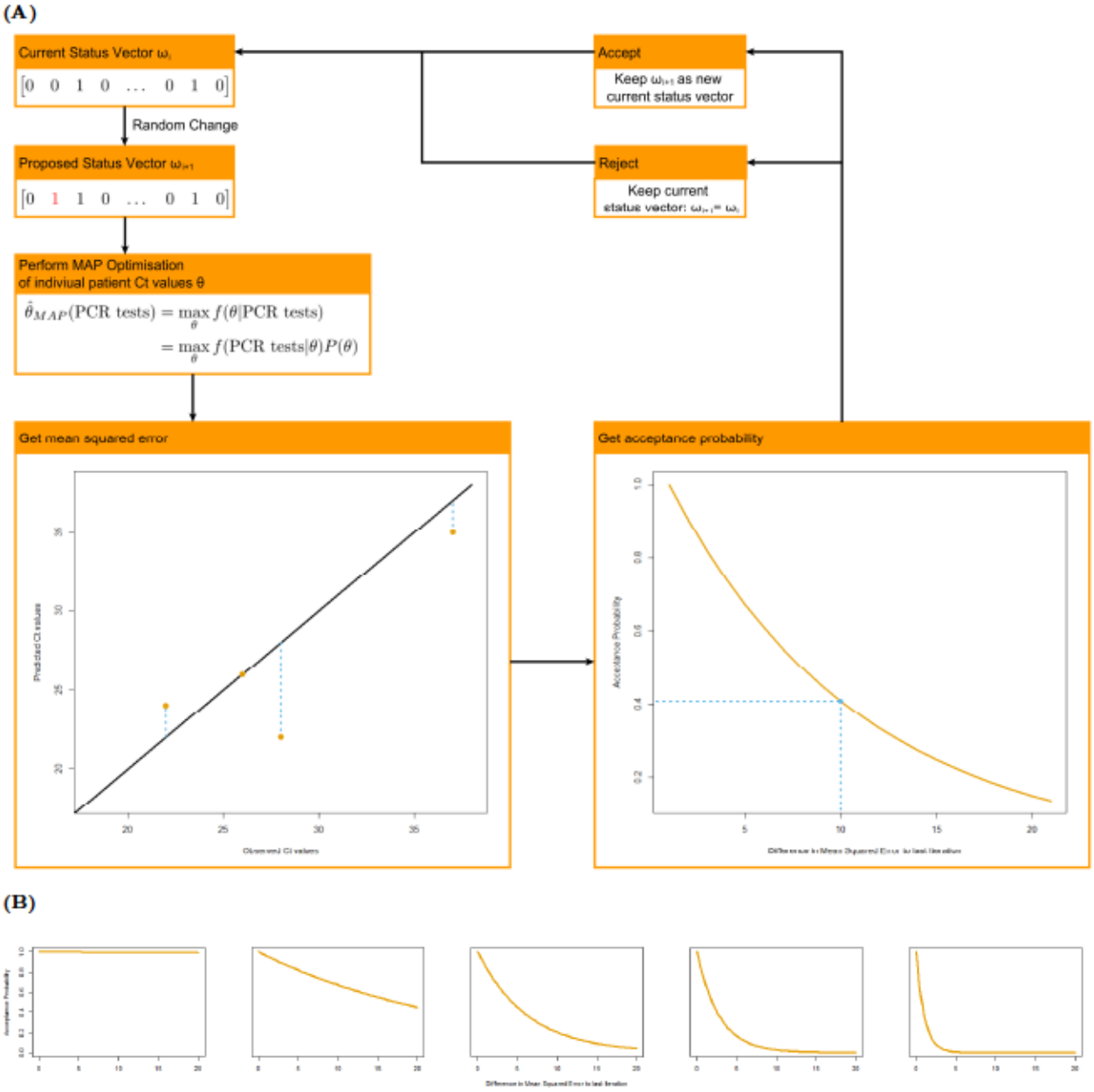


Figure 6.2: **(A)** Illustration of a decoding iteration. In each iteration i a status vector ω_i indicating positivity of the individual samples is randomly changed in one position from positive (1) to negative (0), or vice versa. The resulting status vector ω_{i+1} is used as basis for a *maximum a posteriori* optimization of the individual Ct values (here, the parameters of are denoted in the general form Θ). Based on these individual Ct values, expected group test results are predicted and we obtain a mean squared error between these expected test results and the observed ones. Based on this mean squared errors' difference to the mean squared error in the previous iteration as well as a temperature that declines with every iteration we determine the probability of accepting the proposed status vector ω_{i+1} or rejecting it and continuing with the current status vector ω_i. The declining temperature at every iteration effectively causes the function of the acceptance probability to become steeper, such that we accept almost all proposals in the beginning, but are more strict in the end, accepting only improvements in the mean squared error (negative differences, for which the function is always equal to 1). The status vector after 500 iterations (with 10 iterations at each temperature) is then used to infer the positive samples. **(B)** Change in acceptance probabilities for different temperatures with temperature decreasing from left to right plots. Figure and caption taken from Unruh et al. [1].

illustration). The same test results would also be obtained by the arrangement of positive samples

$$\begin{bmatrix} a/2 & a/2 \\ a/2 & a/2 \end{bmatrix} \tag{6.24}$$

resulting in test outcomes $K - \log(a/2 + a/2) = K - \log a$. Additionally, the Ct values obtained are noisy measurements, extending the possibilities of situations such as this one occurring. In general, the problem is underspecified, which previously inspired the application of compressed sensing techniques to group testing[122]. However, we can take the variability in the optimal solutions obtained through repeated simulated annealing runs into account and improve our inference. We make the assumption that there are some samples that are necessary to be estimated as positive. Extending this, we assume that on repeated runs the probability for any given positive sample to be identified as positive is higher than for a negative sample. We can use this assumption by running the simulated annealing a number of times, 20 in our case. We can then introduce a decision threshold, defined as the proportions of runs an item needs to be identified as positive in to be declared positive in the end. Our in silico simulations suggest that this is a sensible approach as the distributions for positive inferences on negative and positive samples clearly differ. Setting this decision threshold will enable us to control the trade-off between sensitivity and specificity.

Following the simulated annealing, we are left with a set of positive and negative samples. However, due to the logarithmic nature of Ct values samples with a low VL might be obscured if they are in tests together with samples with higher VL. Specifically, they do not decrease the Ct value sufficiently to be distinguishable from noise. We employ an approach to identify these samples inspired by Wang et al.[123]. This approach is best illustrated by example, as shown in Figure 6.3. In general, we first take the inferred test outcomes $\hat{Y}$. We then iterate over the samples declared as negative by the simulated annealing decoding process and set each one separately as positive with a minimum Ct value that is a parameter to the model. Call the sample that is set as positive sample m. We then predict the test outcomes again using Equation 6.17 based on the new individual Ct values (the ones inferred from simulated annealing and one of the other samples set to the minimum Ct value). We compare the new predicted test outcomes to the previously obtained $\hat{Y}$. If the new test outcomes deviate by less than ϵ from any of the previously obtained test outcomes in $\hat{Y}$, sample m is inferred as positive. ϵ was chosen as it is our estimate for the standard deviation of Equation 6.11. Hence, we are interested in whether test results are changed by more than one standard deviation. The intuition behind the approach here is that the other positive samples could hide sample m due to their high contribution to the group tests VL. Hence, if we can assume that sample m has some low Ct value and would not notably contribute to the group test outcomes then it may be positive, but not detectable. Therefore, we infer it as

positive to avoid introducing false negatives.

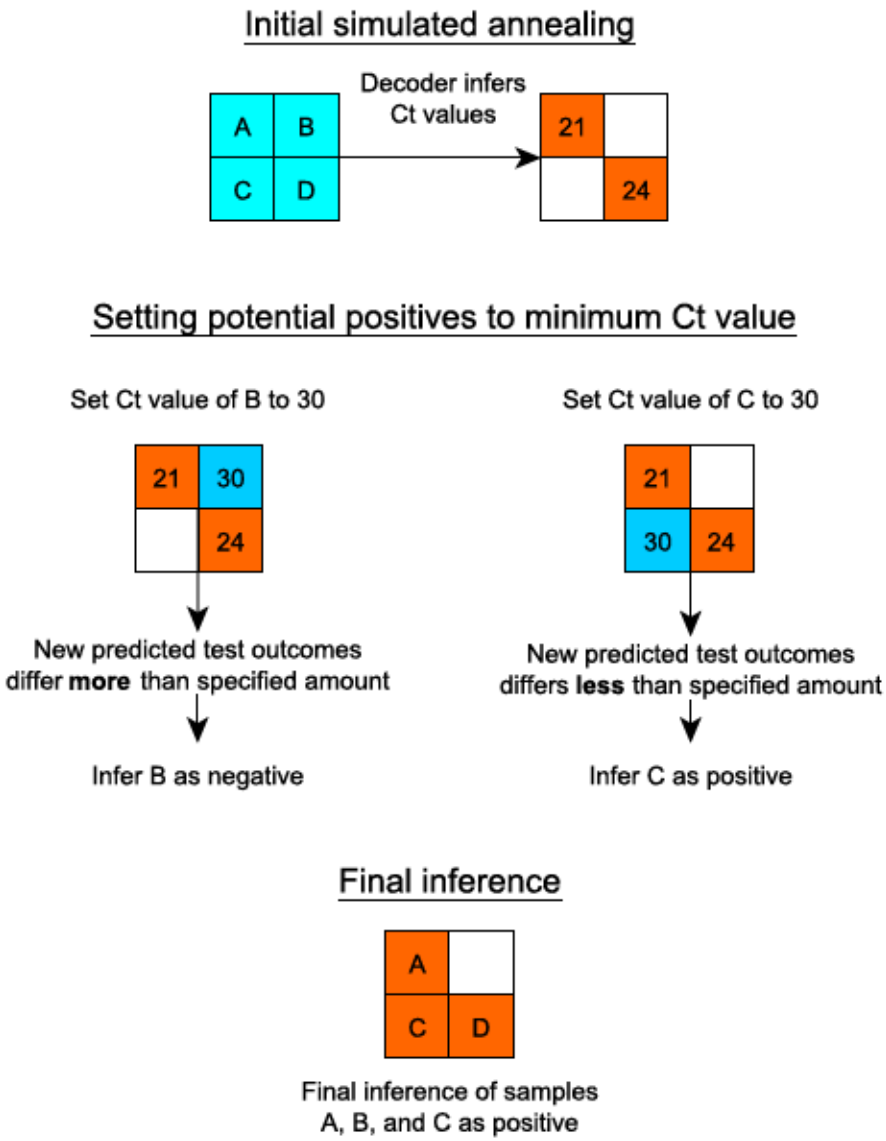


Figure 6.3: Illustration of the approach of setting each negatively inferred item to a minimum Ct value to check whether a low Ct value may be obscured by higher other Ct values. First, the simulated annealing process infers the positives from the potential positives that are in positive row and column tests (samples A, B, C, and D). In this example, sample A is inferred with an individual Ct value of 21 and sample D is inferred with an individual Ct value of 24 (marked in red). We also obtain predicted test outcomes $\hat{Y}_1$. In the next stage, the samples not inferred as positive (samples B and C) are individually set to a Ct value of 30. For example, in the left middle panel, sample B is set to a Ct value of 30. We again infer the predicted test outcomes $\hat{Y}_2$, now assuming sample B is positive with an individual Ct value of 30. Here, we observe that the new predicted test outcomes differ more than ϵ from the initially inferred ones (i.e. $|\hat{Y}_2 - \hat{Y}_1| > \epsilon$). Hence, sample B could not have been hidden by samples A and B and is inferred as negative. We again do the same for sample C, displayed in the right panel. Note that here sample B is again assumed to be negative. We again estimate the new test outcomes $\hat{Y}_3$ and observe that $|\hat{Y}_3 - \hat{Y}_1| \leq \epsilon$. Hence, we infer that if sample C had a low individual Ct value it could be hidden by the higher Ct values of A and B and we infer it as positive. Altogether, we then arrive at the final inference, where samples A, C, and D are inferred as positive, but sample B is inferred as negative.

Parameter Tuning

For tuning of the parameters involved in setting up gtMAP we ran in silico simulations that consisted of one thousand runs of a 3-disjunct STD design with 121 samples for each of 3 to 8 positive samples. We chose 121 here instead of 81 for our in silico and in vitro validation (see below) to avoid overly tuning the model to a particular group testing size, and enable testing of whether results generalize. Firstly, the model underlying gtMAP relies on choosing a temperature decline function. We arrived at the optimal one by in silico testing of three different functions[215]

$$T(c) = T_0 - \alpha c, \tag{6.25}$$

as well as

$$T(c) = a^c T_0, \tag{6.26}$$

and

$$T(c) = \frac{T_0}{1 + \alpha c^2}. \tag{6.27}$$

Function 6.27 was determined to be the best performing temperature decline function. Additionally, we also compared using a uniform or a normal prior for the individual Ct values. These were fitted to a set of 10000 Ct values generated in silico to determine their parameters. Here, the normal prior outperformed the uniform one. Note that the actual distribution will likely be close to a convolution of a normal and a uniform distribution. This can be seen from taking the log of the VL model (Equation 6.12) which will have a linear relationship with Ct values. The resulting piecewise linear function then leads to uniformly distributed Ct values. If we assume normally distributed noise around the actual Ct value we arrive at a convolution of a uniform and normal distribution as Ct value distribution. This motivated our choice of the normal and the uniform distributions as prior distributions. Using the convoluted distribution in RStan proved to be too computationally expensive.

With the basic structure of our model set, we further need to determine four other parameters. All of these parameter settings were tested on the same dataset as before ($n = 121$ in a 3-disjunct design, containing between 3 and 8 positive samples). Firstly, we needed to determine the best estimates for κ and β. We evaluated both $\kappa = 50$ and $\beta = 1$, as well as $\kappa = 37$ and $\beta = 100$ (the same parameters as used for simulated data generation). The first of these options will allow the estimated Ct values to vary over a wider range. $\kappa = 50$ and $\beta = 1$ showed the better performance and were chosen for the model. Additionally, we needed to set ϵ, the estimated error standard deviation of equation 6.11. For this we took an agnostic approach, setting it to 0.5. We further need to set the minimum Ct value for hidden sample detection. Here we tested

Ct value cutoffs of 25, 30, and 35, with 25 delivering the best results. Lastly, a decision threshold needs to be set. This can be set depending on the desired specificity-sensitivity trade-off. Based on our simulations we set the decision threshold to 0.5, which showed a good trade-off between sensitivity and specificity.

Laboratory Methods

Here, I give a brief summary of the in vitro methods. The in vitro experiments were performed by Michael A. Crone and a more detailed description of the laboratory protocols can be found in Unruh et al. [1]. Briefly, for singleplexing of SARS-CoV-2 we used MS2 virus-like particles (VLPs) packaged with the N-gene RNA of SARS-CoV-2. These can be detected with the CDC 2019-Novel Coronavirus RT-PCR diagnostic panel. Our laboratory setup for testing utilized 384-well plates. While in standard clinical practice 96-well plates are more commonly used, 384-well plates are beneficial for high-throughput testing and method validation. Nonetheless we aimed at simulating 96-well plates by subdividing each 384-well plate into 4 parts (allowing for two 36 test group testing designs to be placed in each sub-plate, see below).

Results

In Silico Validation

For testing of our finished model we used a 3-disjunct STD design with 81 samples, as we aimed to use the same setup for in vitro validation. The samples were arranged into 36 groups with 9 samples each. gtMAP's results were then compared to results obtained using COMP. For each of 3 to 8 positive samples we ran 1000 group testing rounds. The results are shown in Figure 6.4. Importantly, we improve the overall classification accuracy as well as the balanced accuracy ($\frac{specificity+sensitivity}{2}$) using gtMAP compared to COMP. Naturally, we observe lower sensitivity, as COMP is designed to create no false negatives. However, this is balanced by an increased specificity and positive predictive value (PPV). Especially the PPV is of high importance when considering group testing designs, as in population testing, most individuals are likely negative. This in turn raises the number of correct classifications overall significantly. Figure 6.5A shows the increase in false positives and false negatives depending on the number of positive samples in a group testing round. In this sensitivity analysis, we again observe the reduced number of false positives and the increased number of false negatives when using gtMAP, showing the main difference between gtMAP and COMP occurs for 6 or more positive samples.

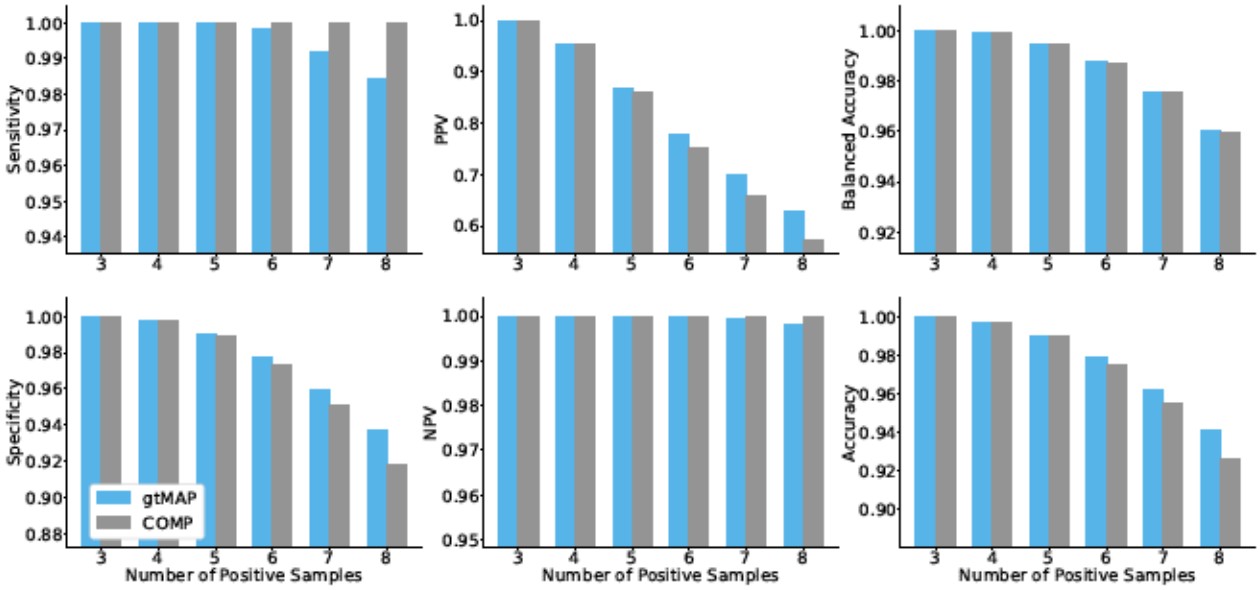


Figure 6.4: Performance characteristics of COMP and gtMAP for in silico simulations for a 3-disjunct group testing design ($n = 81$) with different numbers of positive samples. PPV is the positive predictive value ($\frac{\text{true positives}}{\text{true positives+false positives}}$) and NPV the negative predictive value ($\frac{\text{true negatives}}{\text{true negatives+false negatives}}$). Accuracy is defined as $\frac{\text{true positives+true negatives}}{\text{positives+negatives}}$, i.e. the total number of positive identifications out of all samples.

Following our parameter tuning, we chose a decision threshold of 0.5. It is important to validate that this choice is reasonable. Hence, we again ran the decoder for decision thresholds between 0.05 and 0.95 and plotted the resulting ROC curves for each number of positive samples (Figure 6.5B). Our results show that the decision threshold of 0.5 is still a good choice for a design of a different size ($n = 81$ instead of $n = 121$ used for parameter tuning). Hence, we implemented it as such in the current version of the gtMAP package available at `https://github.com/lahunruh/gtMAP`.

In Vitro Validation

For the singleplexing experiments we chose the same 3-disjunct STD design as for in silico validation. We tested 4 designs with 3, 4, 5, or 6 positive samples in random wells. These were performed with 3 technical replicates as well as 3 independent, biological ones. The VLs of the individual samples were between 0 and 14000 copies/ml chosen uniformly at random. Additionally, we constrained the individual VLs such that no group test would have a VL above 1700 copies/reaction. This was done to enable testing close to the lower limit of detection. To determine the viability of our novel decoder in vitro we again performed a comparison of gtMAP to COMP. In singleplexing the same performance was obtained by gtMAP and COMP. Notably, any design with 3 positive samples did not incur either false positives or false negatives. For more than 3 positive samples we never observed more than one false positive and no false negatives. While these are

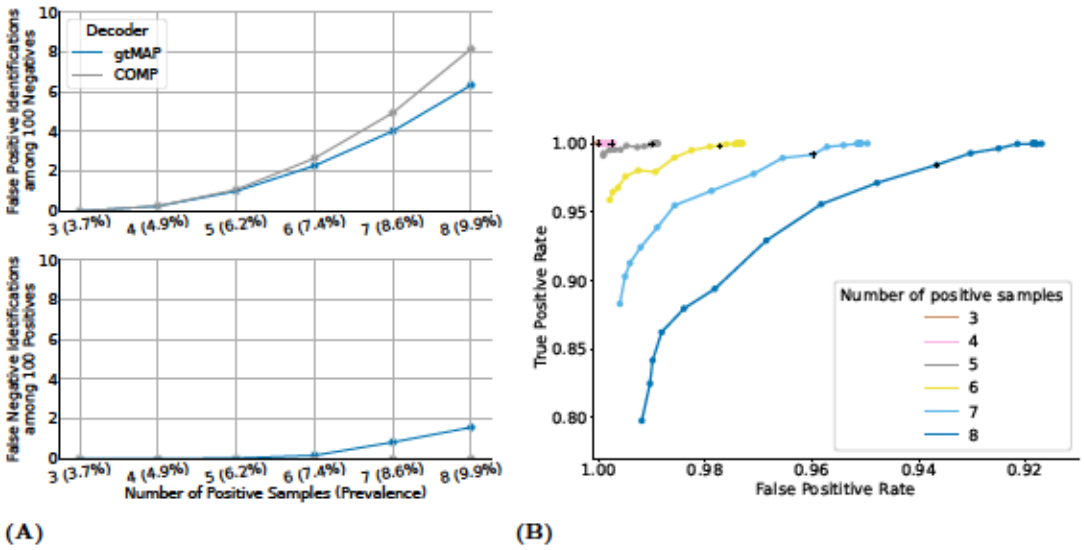


Figure 6.5: **(A)** Sensitivity analysis showing number of false positives and false negatives per 100 positive and negative samples, respectively, for COMP and gtMAP depending on the number of positive samples in a group testing round. **(B)** ROC curves for gtMAP decoding using different decision thresholds (increasing from left to right) and different number of positives. Black '+' indicate a decision threshold of 0.5.

good results, they do not allow for an in-depth evaluation of more complicated scenarios, and are likely due to favorable configurations obtained in the random placement of positive samples. Nonetheless, the results show once again that group testing is a viable strategy for testing of SARS-CoV-2 and further that gtMAP is a viable decoding strategy.

Discussion and Future Work

Our model already shows high performance in singleplexed group testing experiments both in silico and in vitro, validating further research into its development, with the eventual goal of integrating multiplexed group testing results. However, there are still many open avenues both for improving and expanding the current model, or for replacing it with a similar formulation. Firstly, more information about the outbreak that is investigated could be incorporated. A prior encompassing the infection probability or prevalence could be included, essentially a binomial distribution of the expected number of samples estimated as positive. First experiments with this incorporated were conducted, however, including the prevalence prior did not lead to substantially different results. Hence, to improve model simplicity, the prior was not included in the final model. In future models incorporation of this prior could be re-evaluated. Furthermore, test sensitivity and specificity could be included. However, the model complexity would potentially exceed reasonable computational limitations. Notably, approaches to incorporate imperfect sensitivity and specificity in group testing exist, such as NCOMP, which declares all those samples as positive for which at least all but one group test

containing them are declared positive. However, this will lead to many more false positive identifications (see Appendix Figure C1). Therefore, incorporating sensitivity and specificity into the decoding process will require more in-depth research going forward. Beyond improving the model, the estimated individual Ct values, which can be transformed to individual VLs, could be useful. Specifically, we expect to observe different distributions of Ct values at different stages of a pandemic[208]. As such, estimated VLs could provide information about the stage of an outbreak, i.e. whether infections are increasing or decreasing. Conversely, if this information is available, it could be incorporated into the model by adjusting the prior over the individual Ct values. Both approaches are worth investigating in future work. Finally, research into integrating the status vector estimation into the Bayesian model is promising. In the current work, this integration did not lead to a model that converged. However, future work could further explore this, for example by including a probit-link function that relates the individual Ct values to the group testing outcome (e.g. group Ct values above 40 are considered to be negative). This would enable the estimation of all Ct values, instead of just the ones for which $\omega_t = 1$. However, this approach will again require future work to deliver a usable model. Another approach would be to drop the status vector during estimation and instead estimate Ct values for all individual subjects, positive and negative. This approach is laid our in more detail here.

Unified Bayesian Formulation

The decoder in its current form employs a mixture of MAP optimization and simulated annealing. However, a single Bayesian model may be preferable in many situations. We can achieve this through the following model. The validation of the model is left for future research. Importantly, above a sample with an individually negative test outcome ($\omega_t = 0$) was not considered in the estimation of the likelihood. In a world without measurement noise, a negative sample would never cross the threshold used in PCR testing, and hence, would have no Ct value that could sensibly be assigned to it. However, a sufficiently large Ct value can effectively be considered as negative, and we can use this observation to generate our model. Firstly, we assume that any negative PCR test now has a Ct value that is sufficiently far above the test outcomes expected for positive tests, say for example $C_{max} = 50$. We can then incorporate negative samples into our formulation, using the likelihood

$$P(\Gamma|Y) = P(Y|\Gamma)P(\Gamma), \tag{6.28}$$

with Γ the vector of individual Ct values. We define $P(Y|\Gamma)$ as follows.

$$P(Y|\Gamma) = \prod_{t=0}^{t} P(Y_t|\hat{Y}_t) \tag{6.29}$$

with

$$\hat{Y}_i = \kappa - \log_2 \left(\beta \sum_{j=1}^{n} \frac{M_{i,j} 2^{\kappa - \Gamma_j}}{g_i} \right). \tag{6.30}$$

Note ω not appearing in this formulation. Any individual Ct value that can be considered as testing negative here will have the effect that $2^{\kappa - \Gamma_j} < 1$, and will therefore have negligible impact on the group test outcome. In this way we have incorporated negative and positive samples into a single formulation. It remains to define a prior over individual Ct values that encapsulates that most samples will be negative (i.e. they have an individual Ct value of C_{max}), with the remaining positive individual Ct values falling primarily in the range $0 \leq \Gamma_j \leq 40$. To achieve this we can employ a mixture of two normal distributions to obtain the prior

$$\Gamma_j \sim pN(\mu, \sigma) + (1-p)N(C_{max}, \sigma_2), \tag{6.31}$$

where μ and σ are the parameters for the prior over positive individual Ct values as in Equation 6.19 and σ_2 is a sufficiently small value for the standard deviation over the Ct values of negative individual samples. p is a weighting factor between the two distributions and can be chosen to equal our expected prevalence (or could also be estimated as parameter in the model). Through this model we can obtain the posterior marginal distributions for $P(\Gamma_j | Y, M)$. We can then obtain the status vector ω by setting $\omega_j = 1$ if the mean of the posterior of Γ_j is smaller than C_{max}.

Tuning and evaluation of this model is left for future work. First results suggest that one major issue is non convergent chains in the HMC estimation. This is not surprising, as a group testing problem likely has multiple valid solutions. Similar behavior is already shown in the simulated annealing process, where we obtain different results on each run. We incorporated this fact into our estimation through conducting multiple runs of the model and identifying only those samples as positive that were inferred as positive in more than a set number of runs. A similar approach could also be used here through inspection of the posterior distributions across the different chains of the individual Ct values.

Multiplexed Testing

One further promising avenue for future work is to expand this model to a multiplexed setting, considering co-infection probabilities. Such an approach has, to the best of my knowledge, not been developed. One approach to incorporate multiple viruses into the gtMAP model presented in this chapter is to estimate multiple status vectors $\omega^{[1]}, \omega^{[2]}, \ldots, \omega^{[r]}$, where r is the number if respiratory viruses considered. For example, in our in vitro validation of multiplexed group testing we considered SARS-CoV-2, RSV, Influenza A, and Influenza

B[1]. We could then add the log probabilities encompassing $P(\omega_j^{[a]}|\omega_j^{[b]})$ for diseases a and b and sample j to the posterior density. The exact formulation for this, however, remains to be investigated and validated. One approach for incorporating co-infection probabilities in the unified Bayesian model would be to use a probit-link function for obtaining $\omega^{[k]}$. In the model currently implemented in gtMAP, utilizing MAP optimization and simulated annealing, we could similarly pass the complete status vectors to the MAP optimization and again incorporate a sensible choice for $P(\omega_j^{[a]}|\omega_j^{[b]})$. Furthermore, the updating of ω in the simulated annealing procedure would need to be adapted to multiple status vectors. Here, one straightforward approach is to randomly choose one of the status vectors and change a random entry of it. Evaluation of this method is an objective of future work.

Conclusion

Overall, gtMAP, which I validated on simulated and in vitro data of SARS-CoV-2, shows reliable performance and a lot of promise for development in group testing. While naturally, any approach that is not COMP will show at least as many false negatives, gtMAP shows a higher positive predictive value. When mostly negative samples are tested, as will often be the case in practical applications such as surveillance testing, this feature will be highly valuable, and will lead to a significant reduction in false positive identifications. The application of the novel decoder to in vitro data within the workflow also further serves to showcase gtMAP's reliable performance. gtMAP leverages PCR test results, enabling usage of quantitative test outcomes as opposed to simply test positivity and negativity. This is especially useful when the group testing design's error-free capacity is exceeded, as seen in the results on simulated data for more than three positive samples in a test design.

In future research, the application of our workflow applied here to validate gtMAP in vitro and described in more detail in Unruh et al.[1] to different decoders and group testing designs will prove valuable. Especially in vitro comparison of different strategies will be vital and can have an important effect on policy going forward. Policy decisions need to take into account the viability of group testing. Discussions around implementation strategy need to focus on the best group testing design to implement. For the design chosen as basis of the current research, STD, two factors to consider when determining the specifics of the implemented design are group sizes and prevalence rates. In general, larger group sizes have the effect of reducing the proportion of tests required compared to individual tests further. However, they also increase dilutions, potentially reducing sensitivity, and this needs to be taken into account. Higher prevalence rates can increase the number of false identifications, in particular false positives. However, in STD this can be combated easily

by increasing the disjunctness of the design as prevalence increases, eliminating false identifications in many cases. Notably, strategies for prevalence estimation can help in delivering fast updates changing prevalence rates, and the strategy presented in Chapter V can be applied to any STD design, as array testing is always contained in STD. Developing a viable strategy to combine decoding and prevalence estimation will require further research, but may be a promising approach. Another hurdle to effective implementation is how to best enable group testing in laboratories. Pooling in virology labs will require clear instructions for the people setting up the tests. When automated machines are used, creating pooling instructions is straightforward. For the case where pooling is conducted by humans, close collaboration with virology labs will be essential to determine the best approach to guaranteeing error-free pooling procedures. Here, clear instructions and effective protocols are essential to reduce human error. However, among non-adaptive approaches, STD is a prime candidate for easy implementation due to its underlying grid structure.

The work done here could be expanded to other diseases and multiplexed testing. Lastly, it may be valuable to investigate the ideal implementation strategy for multiplexed group testing. For example, Deckert et al. [216] propose context sensitive pooling of samples. This work could be expanded to multiplexed testing with multiple circulating respiratory pathogens by considering the individual prevalence rates and the probability of co-occurring infections. Finally, it is worth emphasizing that a quantitative decoder like gtMAP also returns individual Ct values, which can be transformed into individual VLs. Utilizing this information may prove valuable, as during outbreaks the distribution of VLs in infected individuals depends on the stage of the outbreak[208]. Group testing utilizing quantitative PCR test results may provide insights into the stage of an outbreak going forward. Similarly, if the stage of an outbreak is known, this information could potentially be incorporated into the decoder, for example via a changed prior over Ct values.

In conclusion, group testing can increase testing capacity by requiring far fewer resources than individual testing. At the same time, testing accuracy is not majorly reduced. One major area of application is testing for representative sampling surveillance. The Coronavirus (COVID-19) Infection Survey[217] put into effect by the Office for National Statistics had costs of around £390 million. This large cost makes it prohibitively expensive for resource-constrained applications, but group testing can reduce these costs significantly. Another area of application is genomic surveillance, which has representative sampling as an important pillar[139], and where a large gap can be observed comparing high to low income countries[140]. Again, group testing will reduce this gap by expanding the capacity for tests in low income countries. Multiplexed testing can also increase our capacities by facilitating testing during parallel outbreaks of more than one infectious agents. The rising demand on health systems due to multiple co-circulating respiratory viruses can be alleviated by the application of multiplexed group testing. Furthermore, together with my collaborators we showed the

viability of multiplexed group testing in vitro [1]. Hence, developing a decoder for multiplexed group testing is a prime target for research going forward. Finally, while group testing saves a lot of resources, there are ethical issues surrounding it, for example when reporting a patient as positive even though they are negative. The increased false positive rate of group testing compared to individual testing may be an issue here. However, the STD design can be adapted to higher prevalence rates and when the number of inferred positives exceeds the disjunctness of the design we know that an error may have occured. This in turn could prompt second round re-testing if required. Overall, policy makers need to weigh up the overall costs and benefits, both ethically and focused on population health and healthcare cost, to implement clear guidelines and regulations going forward. This will benefit the targeted development of group testing methods fulfilling the set criteria.

Chapter VII

Discussion

'Da steh´ ich nun, ich armer Tor! Und bin so klug, als wie zuvor.'

– Johann Wolfgang von Goethe

This work started out with the aim to develop novel approaches to improve diagnosis and surveillance of different pathogens. I presented novel tools for high-resolution bacterial identification and group testing, both of which show high quality results, with one of the latter methods being validated in vitro. Still, as is always the case in science, a lot of work remains to be done and the presented research has highlighted several open issues in the field. Here, I will first give a brief synopsis of the main results, but more importantly, this section will serve to put the findings into the broader context of research in the field. I take a wider view and discuss metagenomic methods, high-resolution taxonomic research, and group testing now and in the future, and highlight applications as well as their meaning for research and policy.

Synopsis of Results

I started out by evaluating different available tools for strain-level taxonomic classification in metagenomics. Of the tools I analyzed, two clearly stood out in performance, StrainGE and StrainScan. While both have a lot of potential to be applied in metagenomic analysis, the presented research shows that the choice for a specific tool depends on both the species that is to be investigated and the desired taxonomic resolution. I highlight the need to perform a benchmark prior to application of either of these tools, and am currently working to make an easy to use version of the benchmarking pipeline available to the research community. My investigation also highlighted some shortcomings of previously developed approaches. Of course, accuracy is always a main concern, but more importantly, the available high performing methods lack interpretability and are indifferent to the analytical goals of the researcher employing them. I aimed to address these shortcomings by developing a novel tool, utilizing the benefits of mixed integer linear programming. Mixed integer linear programming has the advantage that it is fairly straightforward to develop interpretable models for complex biological problems, with efficient solvers being readily available. First, I developed several models addressing strain identification with a focus on the core genome. While a lot of time and resources were invested into developing a model based on read to strain assignment, this approach did not show promising results. Similarly, de novo estimation of nucleotide sequences did not bear fruit. On the other hand, a much simpler approach, based on estimating a mixture of reference strains via nucleotide proportions, showed very promising results and good accuracy even on a whole genome level considering that only the core genome was taken into account. The potential for further expansion by extension of the model to consider information beyond the shared genome of a species became clear and I proceeded by developing two models taking into consideration accessory genes. The final APM model I presented in detail in the end shows state-of-the-art performance on several metagenomic datasets. Additionally, by relying on a gene rather than a k-mer basis the estimation of genes not previously found in the identified reference strains is enabled. While there are several desirable improvements to this functionality, it can facilitate in-depth analysis of the genetic makeup of bacterial strains found in microbiomes. Moreover, the model I presented has several tunable parameters with clear interpretations, namely the desired sparsity as well as the importance of the core and the accessory genome in the model. As no tool is perfect for every species of interest, this tunability will be valuable when applying the tool to a new species. Furthermore, the tunability will serve researchers when investigating specific questions. For example, some analysis may be better served by only considering the core genome. One example where this may be true, but remains to be tested, is the inference of highly recombined strains. A large recombination event may preclude precise analysis using other tools. However, the core genome may still reflect the original lineage of the strain well and as such enable better inference. Finally, complete

genes were the main focus of the presented research. However, there is also a place for within gene variation. Especially the identification of conserved indels within genes could be valuable in inferring phylogenies from metagenomes. Conserved indels are indels within a gene present in some individuals but not in others that are in between conserved regions. These indels can provide indications towards when two subgroups of organisms diverged [218]. The approach presented here enables the exploration of conserved indels with little further work by identifying conserved indels in the gene MSAs and it is definitely an avenue worth exploring. Altogether, my research on strain identification in metagenomes suggests that a multi-method approach will be best suited for extracting useful information. None of the available tools are perfect, but different approaches, including the one developed here, provide different benefits and angles of analysis, which all may have their part depending on the research interests of anyone employing these tools.

Next, I moved on to group testing as another important tool in the surveillance of infectious diseases. Our work here presents almost an antithesis to the work on metagenomics, with PCR testing being both highly sensitive and specific for distinct pieces of RNA. In Chapter V, I addressed the prevalence estimation problem in group testing. I carefully developed a highly accurate theoretical framework for estimation of prevalence from array testing results. Given the ease of implementation of array testing and the absence of a need for retesting, it is a group testing design worthy of strong consideration for application in surveillance testing. Again, especially low resource settings will benefit greatly from the savings in tests, and it will open up the possibility to expand testing to a larger portion of the population. In Chapter VI, I presented a novel method for decoding of group testing results based on quantitative PCR test outcomes. This decoder provides highly accurate results, reducing the number of false positive identifications compared to decoding based on binary test results. It highlights the potential for further developments in this area, including designing a decoder that incorporates co-infection probabilities in a multiplexed group testing setting. Notwithstanding the need for further research, the application of our approach, with its validity shown in in vitro experiments, could already help save large quantities of testing resources, a hugely important factor in measuring, and hence, controlling the spread of infectious diseases, especially in low resource settings.

Metagenomics in Context

Environmental sampling has become more widespread and wastewater sampling is a common practice that has led to research that directly influenced my research aims [10,53,54]. The MetaSUB consortium has constantly been pushing the boundaries in sampling of the urban microbiome. As of 2024, their database contains already over 200000 samples from all five continents (`https://portal.geoseeq.com/`). Several impactful works

of research are already based on this data. In a landmark paper, Danko et al.[219] described the urban microbial landscape in 60 countries and provide an analysis of antibiotic resistance across the different sample sites. Further, Zhang et al.[220] attempted to recover sample origin based on microbial signatures. These studies highlight the insights we can gain from metagenomic data from various locations and environments and are a promising step towards the utilization of metagenomics in surveillance strategies. Identification and tracking of specific microbes, and with that precision metagenomics, will play an important part in enabling such surveillance. Moreover, while there have been attempts to assign antibiotic resistance genes to bacterial genera[54] these are still limited in the information they provide. In an ideal world we would be able to assign antibiotic resistance genes to the specific organism carrying it. Take for example *E. coli*. If we discover an antibiotic resistance gene that we could assign to a member of *E. coli*, then this could still mean that it is carried by a commensal or by a pathogenic member of that species. The implications of either are certainly different and highlight the importance of exact determination of the host of an antibiotic resistance gene. With a view towards surveillance, it may be possible for example to predict species or strain specific resistance profiles in a specific area and adapt our antibiotic prescription strategies accordingly. Such an approach is closely aligned with the ideas of antibiotic stewardship[221] and collateral sensitivity[222,223] and would undoubtedly have a significant impact on public health. It would also enable us to understand better the transmission of bacterial strains and antibiotic resistance genes between environments. However, an understanding of how to best conduct environmental surveillance of antibiotic resistance is still lacking. Different studies point to high or low correlations of ARGs in wastewater with clinical resistance prevalence[52,11,53]. Strain-level analyses can help determining which analysis are most helpful in our efforts to surveil antibiotic resistance. Horizontal gene transfer is another important factor to keep in mind, and antibiotic resistance genes that are horizontally transferred may be harder to assign to specific strains. Many ARGs are carried on mobile genetic elements[224], making this a central issue to solve. Nonetheless, there is still specificity in HGT. For example, plasmids are more likely to be shared within a species or between closely related ones[45,46]. Furthermore, relative abundance information can help in assigning even horizontally transferred genes, and figuring out the best way to utilize this information, especially when a gene is present in very similar forms in multiple organisms, is a promising goal for future research. Lastly, we may not require 100% accurate inferences, depending our goals. As such, improved inference of potential carriers of an ARG may still be beneficial compared to essentially no knowledge.

Beyond this, metagenomics could help in outbreak tracking and determination of reservoirs of certain bacteria. Research has already shown the relevance of strain type based techniques such as MLST in outbreak tracking and epidemiological surveillance[69,70,71,72,73]. With a look towards the future, routine surveillance sequencing

via metagenomics can benefit the tracking of outbreaks even when we do not know where to look. Similarly, reservoirs of potentially infectious bacteria could be monitored. Finally, a well set-up surveillance system would allow analyses of outbreaks post-hoc, and samples that were collected without a specific goal in mind could become highly relevant later on. Of course, when considering these potential applications need to keep in mind practical limitations that require to be tackled to implement effective metagenomic surveillance. The first factor is that of cost. While metagenomic surveillance is estimated to be potentially much cheaper than culture-based surveillance[8], it is still a major undertaking. Especially, if we aim for a routine level surveillance structure, costs will play a vital role. Sequencing will likely continue to get cheaper, however, a detailed cost-benefit analysis will be required. Such a study should also establish the areas with the greatest potential, such that not only more resources can be allocated to it, but so that we can also establish aims in regard to what information the employed technology needs to be able to deliver, and whether we can achieve this. Secondly, while bioinformatic tools are constantly progressing, there are still a great many things that are not possible. Here, further development of bioinformatic methods is essential. Developments in sequencing technology can also help address these limitations. Long-read sequencing, even though it currently does not achieve the same sequencing depth as short-read metagenomics, will transform research going forward. Especially high resolution metagenomic studies will be easier to conduct as we obtain longer reads that are easier to assemble. Even though chimeric assemblies may still provide issues, this will make strain level analyses much easier. Furthermore, assigning genes to species or strains will be made more straight forward by easier assemblies as well. Techniques for assigning genes based on flanking regions such as presented in Munk et al.[54] will be made possible at a higher resolution and with more confidence. Long-read sequencing may well completely replace short-read sequencing in the future. In the meantime, the integration of the two approaches has many promises, as showcased by Bertrand et al.[225] who present a hybrid assembler utilizing the benefits of parallel short- and long-read sequencing. However, employing both short- and long-read metagenomic sequencing to the same sample is likely beyond the financial resources of many laboratories. Developing new methods for short-read metagenomics therefore still has an important place in research. Moreover, it will retain that place to some degree, even as technology advances. I already briefly addressed the MetaSUB project and the following has been a topic of intense discussion at the 2024 MetaSUB consortium meeting. Projects like MetaSUB create vast amounts of data, data that will still be relevant in the future, especially when it can be reanalyzed with new approaches and new knowledge. New angles may be taken to approach analyses that can provide new valuable insights. Of course, we cannot foresee some of these approaches, which brings me to my final consideration on metagenomics, the importance of metadata and multiple modality datasets. As just one example, while genetic sequences can provide brilliant insights, it is a very hard problem to infer a phenotype from a genotype. Metadata and multi-omics provide vital information that can help address

this and other issues and the integration of multi-omic data has been widely discussed[226,227]. The present work constitutes an important contribution in bringing us closer to connecting genes and their hosts in metagenomes. Knowledge on this frontier may provide new ways of integrating multi-omics data. Here, an interesting area of research is to utilize metagenomics and other -omics approaches in tandem to infer genes, their hosts, and gene expression. Finally, I strongly believe that general metadata on sample collection, such as location, surface, temperature, pH, time of day, etc. should be similarly integrated into metagenomic and multi-omic analysis. This metadata will further help future re-analysis of the data. However, in my experience, detailed metadata is often missing for metagenomic samples. More focus should be put on the collection of this data, and a standardized set of necessary metadata measures should be developed and widely adopted. Finally, fine-grained metadata will also serve to support surveillance efforts. The lack of fine-grained metadata also highlights data quality more generally, and overall it may be more important to have higher quality data than large amounts of data. In this vein, especially with a view towards basic research, it may be worth collecting fewer samples, but using multiple -omics technologies and collecting high quality metadata, instead of gathering large amounts of lower quality data.

Metagenomic investigations of other environments will play a crucial part in future research and health care and strain level analysis will play a vital part here. Little needs to be said about the potential in metagenomics in the human microbiome and its relation to healthcare. At the time of writing, already two microbiome therapies are approved by the FDA[228] and I project that this is only the beginning of rapid advancements and new medicines. Analysis of human microbiome dynamics and an understanding how these are impacted by pathogens and antibiotics is essential. It is well understood that antibiotic treatment causes dysbiosis and there is strong evidence that this creates an ecological niche for infections to emerge[229]. Furthermore, reinfections with microbes from the microbiome of a patient are indicated by Stracy et al.[230]. Precision metagenomics, i.e. high resolution, high quality information on the members of a bacterial community and their genetic and functional characteristics, can help enormously in tackling these issues. Being able to identify members of a patients microbiome that pose a risk for antibiotically resistant reinfections will influence our prescription strategies. A hurdle to this could be the relatively long processing time of metagenomic samples. Nonetheless, a big benefit is that metagenomics do not require an a priori differential diagnosis to identify a potential infectious organism as opposed to culture based methods[62] and advances in sequencing technologies may enable faster processing in the future. Similarly, high resolution metagenomics could help in understanding specific properties of infectious bacteria that help it thrive in a patient and ultimately lead to infection, though this is mere speculation at this point. Finally, the human microbiome constitutes a reservoir of antibiotic resistance genes[231,232]. These antibiotic resistance genes

might be transferred to pathogenic bacteria via horizontal gene transfer. Hence, identification of which bacteria within humans carry which antibiotic resistance genes may help our understanding on transferred resistance.

To emphasize this point, it is worth stepping back and expanding our view. Many different environments contain the same species of bacteria. Take for example *A. baumannii* or *E. coli*. Then again, there exists substantial variability within bacterial species and it is unclear how the members of a species found in different microbial communities interrelate. This is important in the context of the one health approach. Strain resolution metagenomics could provide novel insights in this area. For example, it may be possible to track the spread of bacterial strains through interconnected environments, enabling improved insights into various questions. How does the microbiome in a subway relate to the microbiome of the humans using it? How do both of these relate to hospital environments and wastewater? When are strains transferred? What is the shape of the gene sharing network in different environments? Finding satisfactory answers to these questions will no doubt require much more research and technological developments, as well as a better understanding of the rules governing microbiome dynamics. Precision metagenomics will be an important pillar of research in this direction. The present work makes a contribution towards enabling research in all of these directions by developing a high precision metagenomics tool aimed at resolving strain-gene relations and I am certain further work in this direction will be of great value. Research could eventually be turned into diagnostic and surveillance tools that enable the tracking of pathogens, antibiotic resistance genes, plasmids, and virulence factors. Personalized medicine is another important area of application that will benefit greatly from precision metagenomics[62]. We are still a way off, but there are few limits to human ingenuity, and few limits on the capacity of knowledge to change worlds, as expressed beautifully by David Deutsch[233]

> *'Like an explosive awaiting a spark, unimaginably numerous environments in the universe are waiting out there, for aeons on end, doing nothing at all or blindly generating evidence and storing it up or pouring it out into space. Almost any of them would, if the right knowledge ever reached it, instantly and irrevocably burst into a radically different type of physical activity: intense knowledge-creation, displaying all the various kinds of complexity, universality and reach that are inherent in the laws of nature, and transforming that environment from what is typical today into what could become typical in the future. If we want to, we could be that spark.'*

Taking a wider view, the considerations we have made call the species concept into question. It is most certainly a useful heuristic, as shown by examples such as the conversed inlA-inlB locus and the Listeria

Pathogenicity Island 1 in almost all members of *L. monocytogenes*[44], or the higher likelihood of plasmid sharing within species[45,46]. However, species such as *E. coli* contain both a wide range of pathogenic and commensal members[152] and plasmids do cross species boundaries[203]. Redondo-Salvo et al.[46] present results that suggest that some clusters of related plasmid are shared across an entire phylum and they find members of most clusters are present across families frequently. Gonzalez et al.[234] even go as far as to postulate a genetic continuum. On the other hand, vertical inheritance still outweighs horizontal gene transfer[46], which lends itself to a discrete interpretation of the genomic landscape of bacteria. In light of this, how should we approach the species concept, and should we talk about bacterial species at all? In favor of the species concept is that species level diagnostics help doctors treat patients effectively and higher resolution approaches are not readily available yet. Further, the species concept can help in developing explanatory theories. In research, we indeed observe regularities at lower taxonomic levels. For example, Goldford et al.[205] have shown that different microbial communities converge toward a stable equilibrium at which the same families of bacteria are present at similar relative abundances. Then again, it is important to acknowledge that the way we classify data shapes the way we think about it and in turn the way we chunk and analyze complex systems. Consider for example the field of multi-omics, which deals with the integration of multiple -omics data modalities such as genomics, proteomics, or metabolomics. Oftentimes, integration of multi-omics data is based on component-sample matrices[227]. The components corresponding to metagenomic data here are often species abundances. Similarly, classification into taxonomic units is oftentimes a first step in metagenomic analysis[235]. Such a summarization step, or even going beyond this to broader taxonomic groups[235], is oftentimes necessary to reduce the dimensionality of the data. Naturally, this a priori classification of the data will have large impacts on the results and their interpretation. While research may often provide valuable insights, and bacterial species do seem to cluster together on measures such as ANI[236], we may also overlook important aspects. Definitions of a particular species are often the result of older classification approaches and our understanding of the structure within a species can change drastically over time[153]. In many cases the genetic sequences grouped together as a species include members from distinct ecosystems[237] and we sometimes observe large diversity in these species. However, Caro-Quintero and Konstantinidis[237] suggest that sequence diversity is much lower among members from ecologically homogeneous environments. This suggests that there may be potentially better species groupings than those regularly employed. These may include grouping by ecosystem, groupings of organisms that show horizontal gene transfer between each other, or other similar approaches. Returning to the concept of the core genome, it may similarly be advantageous to determine the core genome of groups of organisms and clustering them by their core genome types instead of the reverse approach were the shared genome is inferred in a set of organisms determined to be a single species. Most importantly, we should not just rely on default taxonomic databases, for example

the one provided by NCBI used in many tools such as Kraken2[82]. Similarly, while conceptually useful we should not blindly apply the species concept in the interpretations of results, as it is a patchwork (defined by Wilson[238] as an *'extension of linguistic application'*) and new patches such as a species concept based on new measure will inherit some of the concepts associated with the core concept[239]. The assumptions that are carried over should always be questioned. Hanage[240] even hypothesizes on how different our outlook on bacteria could have been had a bacterium that is naturally able to take up DNA from its environment, such as *Streptococcus pneumoniae*, been our model organism instead of *E. coli*. My work is of course no exception to using the predefined term of a species, especially when we discussed clusters of related strains as well as differences in the analysis of different species. However, the current work does focus at its heart on biological units that are much easier to discriminate, single strains identified with unique assemblies, and single genes, which are clearly demarcated in the genome by start and stop codons (though their expression is of course not guaranteed by their presence). Additionally, the model developed in this thesis will work with various underlying databases of strains, with the selection of these strains being the main driver of the interpretations we make. Of course, this selection should be done carefully. Altogether, the species concept is a useful heuristic and clearly represents real structure in biological systems. Nonetheless, one should not blindly apply the species concept or categorize samples by species without putting thought into the implications of the criteria used for species demarcation and the implications of the species definition used should always be considered when interpreting results. Finally, the use of metagenomics, especially with high resolution methods such as the one I presented here may provide novel insights into biological networks that may upend our classical view of bacterial species and lead us to new useful demarcations between microbes. In conclusion, I encourage more active discussion of the issues surrounding the species concept in the research community.

Group Testing in Context

The Covid-19 pandemic has showcased just how essential surveillance testing is in the context of a pandemic, especially in low- and middle-income countries where there are often not enough testing resources available[6,7]. It has been noted that this lack of testing availability can lead to an underestimation of the impact of Covid-19[241]. Furthermore, the need for testing of co-circulating infections in several African countries and the burden this places on the healthcare system have been pointed out[7]. Group testing has received a surge in interest in the research community during the Covid-19 pandemic. Yet, implementations of group testing are few and far between and are mainly limited to classical Dorfman testing despite group testing having

much larger potential than is currently exploited within infectious diseases. In our work we aimed to make the uptake of group testing in clinical practice easier. Beyond developing clear pathways to go from theory and computational experiments to in vitro validation and finally practical application, policy makers also need other information and incentives for implementing group testing. Cleary et al.[208], Brault et al.[242], and Yu et al.[243] all estimate the benefit of group testing during pandemics, modeling the viral load of infected individuals and providing results for expected group testing benefits during different stages of a pandemic. Zhang and Heath[244] further consider contact networks among individuals in a population. All of this research is valuable, but the evaluated testing designs are still oftentimes limited. An exploration of more varied group testing designs will be beneficial in providing more information into the optimal group testing strategy under various practical considerations, constraints, and goals. Furthermore, only Cleary et al.[208] consider prevalence estimation, and with potentially unrealistic group sizes. More research has to be devoted to how effective prevalence estimation can help during a pandemic, how we can save costs, and what the best strategy is in different situations. A central question that can be addressed by simulations is that of how prevalence testing with subsequent prevalence based interventions can lower the overall spread of an infectious disease. This is true especially after the Covid-19 pandemic, as we gained many novel insights into different infection control measures[245]. Potentially, identification at the same time as estimation may bring a lot of benefits. I believe array testing is a great candidate for such an approach. However, more research is required to provide a clear verdict here. Furthermore, prevalence estimation necessarily needs to be combined with the identification of positive samples, which in turn requires the discussion of ethical concerns around false positive identifications. Addressing these concerns will be central going forward, and in-depth research of group testing strategies both for estimation of prevalence and identification of positive samples can convince policy makers to initiate these essential discussions going forward.

The focus in group testing research should therefore clearly be on driving it towards effective application. Nonetheless, there is still room for design improvements. In our work, we addressed the identification of positive samples in group testing using Ct values instead of binary test outcomes. Beyond testing for infectious diseases using non-binary test outcomes may have other applications as well. Zabeti et al.[246] realized the similarity of the problem of identifying SNPs conferring antibiotic resistance to group testing. Here, the testing matrix indicates whether a given bacterial strain carries a SNP and the test outcomes are whether or not a strain is resistant. Importantly, minimum inhibitory concentrations, a measure of the level of antibiotic resistance a bacterium displays, are logarithmic measures. Hence, it is within the realm of possibilities to apply gtMAP to this problem with only slight modifications. This may provide information about resistance carrying SNPs and with some modifications possibly even the amount by which resistance is conferred by

these SNPs. Furthermore, there is still potential for improving the decoding methods used. Researchers have previously addressed how contact structures should be considered in group testing[244]. One could take this information further and include proxies for contact probability in the decoding process, where higher contact probability would imply higher likelihood of both samples being infected simultaneously. Importantly, the decoding process is not the only place where the quantitative nature of Ct values should be considered. The design matrix M can be modified to include different proportions of different samples. Cleverly designed, such a matrix could contain entries that grow quadratically (due to the logarithmic relation of Ct values to viral load), i.e. either one, two, four, or eight parts of one sample are entered into a pool. Beyond an initial design, such an approach would of course also need to be validated in vitro, especially due to its higher demands on the precise splitting of samples. Nonetheless, I believe that an investigation in this area would provide valuable insights. Inferring individual Ct values (and by extension viral loads) during the decoding process can have yet another benefit. The distribution of viral loads in a population varies over the course of a pandemic[208]. Information on individual Ct values could thereby inform us on the current state of an outbreak in the area we are investigating and could reveal whether the incidence is rising or falling. This in turn can provide information for resource allocation, where more resources should be diverted towards areas with rising incidence rates.

Conclusion

In this thesis, I have explored the themes of high resolution taxonomic classification in metagenomics and the use of group testing for resource saving identification of infected individuals as well as for prevalence estimation. I developed multiple novel methods addressing current issues from new angles. The methods developed herein can be employed to gain new insights into microbiomes and help in effectively curbing the spread of infectious diseases. Surveillance of spreading infectious diseases will play an essential part in protecting human health in our highly connected world. Altogether, the developed methods lay a strong foundation for future research that opens up new avenues in surveillance and public health.

Bibliography

[1] Leonhardt AH Unruh, Michael A Crone, Paul S Freemont, and Leonid Chindelevitch. A standardised, high-throughput approach to diagnostic group testing method validation. *medRxiv*, pages 2024–04, 2024.

[2] Kevin S Ikuta, Lucien R Swetschinski, Gisela Robles Aguilar, Fablina Sharara, Tomislav Mestrovic, Authia P Gray, Nicole Davis Weaver, Eve E Wool, Chieh Han, Anna Gershberg Hayoon, et al. Global mortality associated with 33 bacterial pathogens in 2019: a systematic analysis for the global burden of disease study 2019. *The Lancet*, 400(10369):2221–2248, 2022.

[3] Christopher JL Murray, Kevin Shunji Ikuta, Fablina Sharara, Lucien Swetschinski, Gisela Robles Aguilar, Authia Gray, Chieh Han, Catherine Bisignano, Puja Rao, Eve Wool, et al. Global burden of bacterial antimicrobial resistance in 2019: a systematic analysis. *The lancet*, 399(10325):629–655, 2022.

[4] Mohammad Ahmad and Asad U Khan. Global economic impact of antibiotic resistance: A review. *Journal of global antimicrobial resistance*, 19:313–316, 2019.

[5] WHO. Global surveillance for covid-19 caused by human infection with covid-19 virus: Interim guidance, 2020.

[6] Nahla Khamis Ibrahim. Epidemiologic surveillance for controlling covid-19 pandemic: types, challenges and implications. *Journal of infection and public health*, 13(11):1630–1638, 2020.

[7] Yusuff Adebayo Adebisi, Adrian Rabe, and Don Eliseo Lucero-Prisno III. Covid-19 surveillance systems in african countries. *Health promotion perspectives*, 11(4):382, 2021.

[8] Frank M Aarestrup and Mark EJ Woolhouse. Using sewage for surveillance of antimicrobial resistance. *Science*, 367(6478):630–632, 2020.

[9] Rene S Hendriksen, Oksana Lukjancenko, Patrick Munk, Mathis H Hjelmsø, Jennifer R Verani, Eric Ng'eno, Godfrey Bigogo, Samuel Kiplangat, Traoré Oumar, Lasse Bergmark, et al. Pathogen surveil-

lance in the informal settlement, kibera, kenya, using a metagenomics approach. *PLoS One*, 14(10): e0222531, 2019.

[10] Rene S Hendriksen, Patrick Munk, Patrick Njage, Bram Van Bunnik, Luke McNally, Oksana Lukjancenko, Timo Röder, David Nieuwenhuijse, Susanne Karlsmose Pedersen, Jette Kjeldgaard, et al. Global monitoring of antimicrobial resistance based on metagenomics analyses of urban sewage. *Nature communications*, 10(1):1124, 2019.

[11] Marion Hutinel, Patricia Maria Catharina Huijbers, Jerker Fick, Christina Åhrén, Dan Göran Joakim Larsson, and Carl-Fredrik Flach. Population-level surveillance of antibiotic resistance in escherichia coli through sewage analysis. *Eurosurveillance*, 24(37):1800497, 2019.

[12] Christopher Quince, Alan W Walker, Jared T Simpson, Nicholas J Loman, and Nicola Segata. Shotgun metagenomics, from sampling to analysis. *Nature biotechnology*, 35(9):833–844, 2017.

[13] John C Wooley, Adam Godzik, and Iddo Friedberg. A primer on metagenomics. *PLoS computational biology*, 6(2):e1000667, 2010.

[14] Eric L Delwart. Viral metagenomics. *Reviews in medical virology*, 17(2):115–131, 2007.

[15] Jan-Niklas Macher, Vera Marie Alida Zizka, Alexander Martin Weigand, and Florian Leese. A simple centrifugation protocol for metagenomic studies increases mitochondrial dna yield by two orders of magnitude. *Methods in Ecology and Evolution*, 9(4):1070–1074, 2018.

[16] Zachery W Dickson, Dirk Hackenberger, Melanie Kuch, Art Marzok, Arinjay Banerjee, Laura Rossi, Jennifer Ann Klowak, Alison Fox-Robichaud, Karen Mossmann, Matthew S Miller, et al. Probe design for simultaneous, targeted capture of diverse metagenomic targets. *Cell reports methods*, 1(6), 2021.

[17] Melanie Schirmer, Rosalinda D'Amore, Umer Z Ijaz, Neil Hall, and Christopher Quince. Illumina error profiles: resolving fine-scale variation in metagenomic sequencing data. *BMC bioinformatics*, 17:1–15, 2016.

[18] Michael A Tanner, Brett M Goebel, Michael A Dojka, and Norman R Pace. Specific ribosomal dna sequences from diverse environmental settings correlate with experimental contaminants. *Applied and environmental microbiology*, 64(8):3110–3113, 1998.

[19] Anthony M Bolger, Marc Lohse, and Bjoern Usadel. Trimmomatic: a flexible trimmer for illumina sequence data. *Bioinformatics*, 30(15):2114–2120, 2014.

[20] S. Andrews. Fastqc: A quality control tool for high throughput sequence data, 2010. URL `http://www.bioinformatics.babraham.ac.uk/projects/fastqc/`.

[21] Inc. Illumina. Illumina adapter portfolio, 2025. URL `https://knowledge.illumina.com/library-preparation/general/library-preparation-general-reference_material-list/000003275`.

[22] Martin Ayling, Matthew D Clark, and Richard M Leggett. New approaches for metagenome assembly with short reads. *Briefings in bioinformatics*, 21(2):584–594, 2020.

[23] Daniel R Mende, Alison S Waller, Shinichi Sunagawa, Aino I Järvelin, Michelle M Chan, Manimozhiyan Arumugam, Jeroen Raes, and Peer Bork. Assessment of metagenomic assembly using simulated next generation sequencing data. *PloS one*, 7(2):e31386, 2012.

[24] João C Setubal. Metagenome-assembled genomes: concepts, analogies, and challenges. *Biophysical reviews*, 13(6):905–909, 2021.

[25] Dongwan D Kang, Feng Li, Edward Kirton, Ashleigh Thomas, Rob Egan, Hong An, and Zhong Wang. Metabat 2: an adaptive binning algorithm for robust and efficient genome reconstruction from metagenome assemblies. *PeerJ*, 7:e7359, 2019.

[26] Hiromi Nishida. Genome dna sequence variation, evolution, and function in bacteria and archaea. *Current Issues in Molecular Biology*, 15(1):19–24, 2013.

[27] Lucio Aliperti, Ariel A Aptekmann, Gonzalo Farfañuk, Luciana L Couso, Alfonso Soler-Bistué, and Ignacio E Sánchez. r/k selection of gc content in prokaryotes. *Environmental Microbiology*, 25(12): 3255–3268, 2023.

[28] Christopher Quince, Tom O Delmont, Sébastien Raguideau, Johannes Alneberg, Aaron E Darling, Gavin Collins, and A Murat Eren. Desman: a new tool for de novo extraction of strains from metagenomes. *Genome biology*, 18:1–22, 2017.

[29] Chao Yang, Debajyoti Chowdhury, Zhenmiao Zhang, William K Cheung, Aiping Lu, Zhaoxiang Bian, and Lu Zhang. A review of computational tools for generating metagenome-assembled genomes from metagenomic sequencing data. *Computational and Structural Biotechnology Journal*, 19:6301–6314, 2021.

[30] Martial Marbouty and Romain Koszul. Metagenomes binning using proximity-ligation data. *Hi-C Data Analysis: Methods and Protocols*, pages 163–181, 2022.

[31] Martial Marbouty and Romain Koszul. Metagenome analysis exploiting high-throughput chromosome conformation capture (3c) data. *Trends in Genetics*, 31(12):673–682, 2015.

[32] Christopher W Beitel, Lutz Froenicke, Jenna M Lang, Ian F Korf, Richard W Michelmore, Jonathan A Eisen, and Aaron E Darling. Strain-and plasmid-level deconvolution of a synthetic metagenome by sequencing proximity ligation products. *PeerJ*, 2:e415, 2014.

[33] Joshua N Burton, Ivan Liachko, Maitreya J Dunham, and Jay Shendure. Species-level deconvolution of metagenome assemblies with hi-c–based contact probability maps. *G3: Genes, Genomes, Genetics*, 4(7):1339–1346, 2014.

[34] Juliane C Dohm, Philipp Peters, Nancy Stralis-Pavese, and Heinz Himmelbauer. Benchmarking of long-read correction methods. *NAR Genomics and Bioinformatics*, 2(2):lqaa037, 2020.

[35] Chankyung Kim, Monnat Pongpanich, and Thantrira Porntaveetus. Unraveling metagenomics through long-read sequencing: A comprehensive review. *Journal of Translational Medicine*, 22(1):111, 2024.

[36] Peter E Warburton and Robert P Sebra. Long-read dna sequencing: recent advances and remaining challenges. *Annual review of genomics and human genetics*, 24(1):109–132, 2023.

[37] Weichun Huang, Leping Li, Jason R Myers, and Gabor T Marth. Art: a next-generation sequencing read simulator. *Bioinformatics*, 28(4):593–594, 2012.

[38] E Michael Valencia, Katherine A Maki, Jennifer N Dootz, and Jennifer J Barb. Mock community taxonomic classification performance of publicly available shotgun metagenomics pipelines. *Scientific Data*, 11(1):81, 2024.

[39] Pieter-Jan Van Camp and Aleksey Porollo. Seq2mgs: an effective tool for generating realistic artificial metagenomes from the existing sequencing data. *NAR Genomics and Bioinformatics*, 4(3):lqac050, 2022.

[40] Robert J Baker and Robert D Bradley. Speciation in mammals and the genetic species concept. *Journal of mammalogy*, 87(4):643–662, 2006.

[41] EaBMG Stackebrandt and Brett M Goebel. Taxonomic note: a place for dna-dna reassociation and 16s rrna sequence analysis in the present species definition in bacteriology. *International journal of systematic and evolutionary microbiology*, 44(4):846–849, 1994.

[42] Jessica E Cooper and Edward J Feil. Multilocus sequence typing–what is resolved? *Trends in microbiology*, 12(8):373–377, 2004.

[43] Konstantinos T Konstantinidis and James M Tiedje. Genomic insights that advance the species definition for prokaryotes. *Proceedings of the National Academy of Sciences*, 102(7):2567–2572, 2005.

[44] Juan J Quereda, Alvaro Morón-García, Carla Palacios-Gorba, Charlotte Dessaux, Francisco García-del Portillo, M Graciela Pucciarelli, and Alvaro D Ortega. Pathogenicity and virulence of listeria monocytogenes: A trip from environmental to medical microbiology. *Virulence*, 12(1):2509–2545, 2021.

[45] In-Geol Choi and Sung-Hou Kim. Global extent of horizontal gene transfer. *Proceedings of the National Academy of Sciences*, 104(11):4489–4494, 2007.

[46] Santiago Redondo-Salvo, Raúl Fernández-López, Raúl Ruiz, Luis Vielva, María de Toro, Eduardo PC Rocha, M Pilar Garcillán-Barcia, and Fernando de la Cruz. Pathways for horizontal gene transfer in bacteria revealed by a global map of their plasmids. *Nature communications*, 11(1):3602, 2020.

[47] Telleasha L Greay, Alexander W Gofton, Alireza Zahedi, Andrea Paparini, Kathryn L Linge, Cynthia A Joll, and Una M Ryan. Evaluation of 16s next-generation sequencing of hypervariable region 4 in wastewater samples: An unsuitable approach for bacterial enteric pathogen identification. *Science of the total environment*, 670:1111–1124, 2019.

[48] Jerald Conrad Ibal, Huy Quang Pham, Chang Eon Park, and Jae-Ho Shin. Information about variations in multiple copies of bacterial 16s rrna genes may aid in species identification. *PLoS One*, 14(2): e0212090, 2019.

[49] Juan Jovel, Jordan Patterson, Weiwei Wang, Naomi Hotte, Sandra O'Keefe, Troy Mitchel, Troy Perry, Dina Kao, Andrew L Mason, Karen L Madsen, et al. Characterization of the gut microbiome using 16s or shotgun metagenomics. *Frontiers in microbiology*, 7:459, 2016.

[50] Leonid Chindelevitch, Elita Jauneikaite, Nicole E Wheeler, Kasim Allel, Bede Yaw Ansiri-Asafoakaa, Wireko A Awuah, Denis C Bauer, Stephan Beisken, Kara Fan, Gary Grant, et al. Applying data technologies to combat amr: current status, challenges, and opportunities on the way forward. *arXiv preprint arXiv:2208.04683*, 2022.

[51] About one health. URL `https://www.cdc.gov/one-health/about/index.html`. Accessed: 30-09-2024.

[52] Katariina MM Pärnänen, Carlos Narciso-da Rocha, David Kneis, Thomas U Berendonk, Damiano Cacace, Thi Thuy Do, Christian Elpers, Despo Fatta-Kassinos, Isabel Henriques, Thomas Jaeger, et al. Antibiotic resistance in european wastewater treatment plants mirrors the pattern of clinical antibiotic resistance prevalence. *Science advances*, 5(3):eaau9124, 2019.

[53] Antti Karkman, Fanny Berglund, Carl-Fredrik Flach, Erik Kristiansson, and DG Joakim Larsson. Predicting clinical resistance prevalence using sewage metagenomic data. *Communications Biology*, 3 (1):711, 2020.

[54] Patrick Munk, Christian Brinch, Frederik Duus Møller, Thomas N Petersen, Rene S Hendriksen, Anne Mette Seyfarth, Jette S Kjeldgaard, Christina Aaby Svendsen, Bram van Bunnik, Fanny Berglund, et al. Genomic analysis of sewage from 101 countries reveals global landscape of antimicrobial resistance. *Nature Communications*, 13(1):7251, 2022.

[55] OneHealthTrust. Resistancemap, 2024. URL `https://resistancemap.onehealthtrust.org/AntibioticResistance.php`.

[56] Finlay Maguire, Baofeng Jia, Kristen L Gray, Wing Yin Venus Lau, Robert G Beiko, and Fiona SL Brinkman. Metagenome-assembled genome binning methods with short reads disproportionately fail for plasmids and genomic islands. *Microbial genomics*, 6(10):e000436, 2020.

[57] Hirokazu Yano, Masaki Shintani, Masaru Tomita, Haruo Suzuki, and Taku Oshima. Reconsidering plasmid maintenance factors for computational plasmid design. *Computational and Structural Biotechnology Journal*, 17:70–81, 2019.

[58] Patricia MC Huijbers, Carl-Fredrik Flach, and DG Joakim Larsson. A conceptual framework for the environmental surveillance of antibiotics and antibiotic resistance. *Environment international*, 130: 104880, 2019.

[59] Abby P Douglas, Caroline Marshall, Sarah L Baines, David Ritchie, Jeff Szer, Victoria Madigan, Hiu Tat Chan, Susan A Ballard, Benjamin P Howden, Kirsty Buising, et al. Utilizing genomic analyses to investigate the first outbreak of van a vancomycin-resistant enterococcus in australia with emergence of daptomycin non-susceptibility. *Journal of Medical Microbiology*, 68(3):303–308, 2019.

[60] Ryan C Johnson, Clay Deming, Sean Conlan, Caroline J Zellmer, Angela V Michelin, ShihQueen Lee-Lin, Pamela J Thomas, Morgan Park, Rebecca A Weingarten, John Less, et al. Investigation of a cluster of sphingomonas koreensis infections. *New England Journal of Medicine*, 379(26):2529–2539, 2018.

[61] Brian D Ondov, Todd J Treangen, Páll Melsted, Adam B Mallonee, Nicholas H Bergman, Sergey Koren, and Adam M Phillippy. Mash: fast genome and metagenome distance estimation using minhash. *Genome biology*, 17:1–14, 2016.

[62] Charles Y Chiu and Steven A Miller. Clinical metagenomics. *Nature Reviews Genetics*, 20(6):341–355, 2019.

[63] Kern Rei Chng, Chenhao Li, Denis Bertrand, Amanda Hui Qi Ng, Junmei Samantha Kwah, Hwee Meng Low, Chengxuan Tong, Maanasa Natrajan, Michael Hongjie Zhang, Licheng Xu, et al. Cartography of opportunistic pathogens and antibiotic resistance genes in a tertiary hospital environment. *Nature medicine*, 26(6):941–951, 2020.

[64] Martin CJ Maiden. Multilocus sequence typing of bacteria. *Annu. Rev. Microbiol.*, 60:561–588, 2006.

[65] KA Jolley, J Kalmusova, EJ Feil, S Gupta, M Musilek, P Kriz, and MCJ Maiden. Carried meningococci in the czech republic: a diverse recombining population. *Journal of clinical microbiology*, 38(12):4492–4498, 2000.

[66] Wieger L Homan, David Tribe, Simone Poznanski, Mei Li, Geoff Hogg, Emile Spalburg, Jan DA Van Embden, and Rob JL Willems. Multilocus sequence typing scheme for enterococcus faecium. *Journal of clinical microbiology*, 40(6):1963–1971, 2002.

[67] Edward J Feil, Jessica E Cooper, Hajo Grundmann, D Ashley Robinson, Mark C Enright, Tony Berendt, Sharon J Peacock, John Maynard Smith, Michael Murphy, Brian G Spratt, et al. How clonal is staphylococcus aureus? *Journal of bacteriology*, 185(11):3307–3316, 2003.

[68] AP Johnson, CL Sheppard, SJ Harnett, A Birtles, Timothy Harrison, NP Brenwald, MJ Gill, RA Walker, DM Livermore, and RC George. Emergence of a fluoroquinolone-resistant strain of streptococcus pneumoniae in england. *Journal of Antimicrobial Chemotherapy*, 52(6):953–960, 2003.

[69] Edmond J Byrnes III, Wenjun Li, Yonathan Lewit, Hansong Ma, Kerstin Voelz, Ping Ren, Dee A Carter, Vishnu Chaturvedi, Robert J Bildfell, Robin C May, et al. Emergence and pathogenicity of highly virulent cryptococcus gattii genotypes in the northwest united states. *PLoS pathogens*, 6(4): e1000850, 2010.

[70] Izabel Cristina Vanzato Palazzo, A Pitondo-Silva, Carlos E Levy, and AL da Costa Darini. Changes in vancomycin-resistant enterococcus faecium causing outbreaks in brazil. *Journal of Hospital Infection*, 79(1):70–74, 2011.

[71] Otto G Vanderkooi, Deirdre L Church, Judy MacDonald, Franziska Zucol, and James D Kellner. Community-based outbreaks in vulnerable populations of invasive infections caused by streptococcus pneumoniae serotypes 5 and 8 in calgary, canada. *PloS one*, 6(12):e28547, 2011.

[72] Carina Brehony, Keith A Jolley, and Martin CJ Maiden. Multilocus sequence typing for global surveillance of meningococcal disease. *FEMS microbiology reviews*, 31(1):15–26, 2007.

[73] Caroline L Trotter, Manosree Chandra, Rosa Cano, Amparo Larrauri, Mary E Ramsay, Carina Brehony, Keith A Jolley, Martin CJ Maiden, Sigrid Heuberger, and Matthias Frosch. A surveillance network for meningococcal disease in europe. *FEMS microbiology reviews*, 31(1):27–36, 2007.

[74] Keith A Jolley, James E Bray, and Martin CJ Maiden. Open-access bacterial population genomics: Bigsdb software, the pubmlst. org website and their applications. *Wellcome open research*, 3, 2018.

[75] Christopher Quince, Tom O Delmont, Sébastien Raguideau, Johannes Alneberg, Aaron E Darling, Gavin Collins, and A Murat Eren. Desman: a new tool for de novo extraction of strains from metagenomes. *Genome biology*, 18:1–22, 2017.

[76] Chengwei Luo, Rob Knight, Heli Siljander, Mikael Knip, Ramnik J Xavier, and Dirk Gevers. Constrains identifies microbial strains in metagenomic datasets. *Nature biotechnology*, 33(10):1045–1052, 2015.

[77] Nicola Segata, Levi Waldron, Annalisa Ballarini, Vagheesh Narasimhan, Olivier Jousson, and Curtis Huttenhower. Metagenomic microbial community profiling using unique clade-specific marker genes. *Nature methods*, 9(8):811–814, 2012.

[78] Sergio Pulido-Tamayo, Aminael Sánchez-Rodríguez, Toon Swings, Bram Van den Bergh, Akanksha Dubey, Hans Steenackers, Jan Michiels, Jan Fostier, and Kathleen Marchal. Frequency-based haplotype reconstruction from deep sequencing data of bacterial populations. *Nucleic acids research*, 43(16):e105–e105, 2015.

[79] Sergio Andreu-Sánchez, Lianmin Chen, Daoming Wang, Hannah E Augustijn, Alexandra Zhernakova, and Jingyuan Fu. A benchmark of genetic variant calling pipelines using metagenomic short-read sequencing. *Frontiers in genetics*, 12:648229, 2021.

[80] Balamurugan Jagadeesan, Peter Gerner-Smidt, Marc W Allard, Sébastien Leuillet, Anett Winkler, Yinghua Xiao, Samuel Chaffron, Jos Van Der Vossen, Silin Tang, Mitsuru Katase, et al. The use of next generation sequencing for improving food safety: Translation into practice. *Food microbiology*, 79: 96–115, 2019.

[81] Carlos Valiente-Mullor, Beatriz Beamud, Iván Ansari, Carlos Francés-Cuesta, Neris García-González, Lorena Mejía, Paula Ruiz-Hueso, and Fernando González-Candelas. One is not enough: on the effects of reference genome for the mapping and subsequent analyses of short-reads. *PLoS computational biology*, 17(1):e1008678, 2021.

[82] Derrick E Wood, Jennifer Lu, and Ben Langmead. Improved metagenomic analysis with kraken 2. *Genome biology*, 20:1–13, 2019.

[83] Moreno Zolfo, Adrian Tett, Olivier Jousson, Claudio Donati, and Nicola Segata. Metamlst: multi-locus strain-level bacterial typing from metagenomic samples. *Nucleic acids research*, 45(2):e7–e7, 2017.

[84] Davide Albanese and Claudio Donati. Strain profiling and epidemiology of bacterial species from metagenomic sequencing. *Nature communications*, 8(1):2260, 2017.

[85] Derrick E Wood and Steven L Salzberg. Kraken: ultrafast metagenomic sequence classification using exact alignments. *Genome biology*, 15:1–12, 2014.

[86] Florian P Breitwieser, Daniel N Baker, and Steven L Salzberg. Krakenuniq: confident and fast metagenomics classification using unique k-mer counts. *Genome biology*, 19:1–10, 2018.

[87] Meriam Guellil, Marcel Keller, Jenna M Dittmar, Sarah A Inskip, Craig Cessford, Anu Solnik, Toomas Kivisild, Mait Metspalu, John E Robb, and Christiana L Scheib. An invasive haemophilus influenzae serotype b infection in an anglo-saxon plague victim. *Genome biology*, 23(1):22, 2022.

[88] Zhi Ruan, Shengmei Zou, Zeyu Wang, Luhan Zhang, Hangfei Chen, Yuye Wu, Huiqiong Jia, Mohamed S Draz, and Ye Feng. Toward accurate diagnosis and surveillance of bacterial infections using enhanced strain-level metagenomic next-generation sequencing of infected body fluids. *Briefings in Bioinformatics*, 23(2):bbac004, 2022.

[89] Lucas R van Dijk, Bruce J Walker, Timothy J Straub, Colin J Worby, Alexandra Grote, Henry L Schreiber IV, Christine Anyansi, Amy J Pickering, Scott J Hultgren, Abigail L Manson, et al. Strainge: a toolkit to track and characterize low-abundance strains in complex microbial communities. *Genome biology*, 23(1):74, 2022.

[90] Tae-Hyuk Ahn, Juanjuan Chai, and Chongle Pan. Sigma: strain-level inference of genomes from metagenomic analysis for biosurveillance. *Bioinformatics*, 31(2):170–177, 2015.

[91] Herui Liao, Yongxin Ji, and Yanni Sun. High-resolution strain-level microbiome composition analysis from short reads. *Microbiome*, 11(1):183, 2023.

[92] Daniel N Baker and Ben Langmead. Dashing: fast and accurate genomic distances with hyperloglog. *Genome biology*, 20:1–12, 2019.

[93] The Human Microbiome Project Consortium. Structure, function and diversity of the healthy human microbiome. *nature*, 486(7402):207–214, 2012.

[94] Giuseppe Lancia. Mathematical programming in computational biology: an annotated bibliography. *Algorithms*, 1(2):100–129, 2008.

[95] Robert J. Vanderbei. *Linear Programming: Foundations and Extensions*. Springer New York, New York, 2013.

[96] Giacomo Zambelli. Michele Conforti, Gérard Cornuéjols. *Integer Programming*. Springer Cham, Switzerland, 2013. doi: 10.1007/978-3-319-11008-0.

[97] J Erickson. Integer programming reduction, 2015. URL `https://web.archive.org/web/20150518072946/https://courses.engr.illinois.edu/cs498dl1/sp2015/solutions/hw10sol.pdf`.

[98] David G Luenberger, Yinyu Ye, et al. *Linear and nonlinear programming*, volume 2. Springer, 1984.

[99] Hae-Young Kim and Michael G Hudgens. Three-dimensional array-based group testing algorithms. *Biometrics*, 65(3):903–910, 2009. doi: https://doi.org/10.1111/j.1541-0420.2008.01158.x.

[100] Desmond C. Ong. Statistics and analytics for the social and computing sciences, 2022. URL `https://desmond-ong.github.io/stats-notes/`. Accessed: 30-09-2024.

[101] Ana Guasque and Patricia Balbastre. Evaluation and comparison of integer programming solvers for hard real-time scheduling. *IEICE TRANSACTIONS on Information and Systems*, 105(10):1726–1733, 2022.

[102] Arno Luppold, Dominic Oehlert, and Heiko Falk. Evaluating the performance of solvers for integer-linear programming. Technical report, 2018.

[103] Robert Dorfman. The detection of defective members of large populations. *The Annals of Mathematical Statistics*, 14(4):436–440, 1943.

[104] Noam Shental, Shlomia Levy, Vered Wuvshet, Shosh Skorniakov, Bar Shalem, Aner Ottolenghi, Yariv Greenshpan, Rachel Steinberg, Avishay Edri, Roni Gillis, Michal Goldhirsh, Khen Moscovici, Sinai Sachren, Lilach M. Friedman, Lior Nesher, Yonat Shemer-Avni, Angel Porgador, and Tomer Hertz. Efficient high-throughput sars-cov-2 testing to detect asymptomatic carriers. *Science Advances*, 6(37), 2020. doi: 10.1126/sciadv.abc5961.

[105] Jin-Taek Seong. Group testing-based robust algorithm for diagnosis of covid-19. *Diagnostics*, 10(6): 396, 2020. doi: https://doi.org/10.3390/diagnostics10060396.

[106] Hae-Young Kim, Michael G Hudgens, Jonathan M Dreyfuss, Daniel J Westreich, and Christopher D Pilcher. Comparison of group testing algorithms for case identification in the presence of test error. *Biometrics*, 63(4):1152–1163, 2007. doi: 10.1111/j.1541-0420.2007.00817.x.

[107] Noam Shental, Amnon Amir, and Or Zuk. Identification of rare alleles and their carriers using compressed se(que)nsing. *Nucleic Acids Research*, 38(19):e179–e179, 08 2010. ISSN 0305-1048. doi: 10.1093/nar/gkq675. URL `https://doi.org/10.1093/nar/gkq675`.

[108] Michael T. Goodrich, Mikhail J. Atallah, and Roberto Tamassia. Indexing information for data forensics. In John Ioannidis, Angelos Keromytis, and Moti Yung, editors, *Applied Cryptography and Network Security*, pages 206–221, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg. ISBN 978-3-540-31542-1.

[109] Matthew Aldridge, Oliver Johnson, and Jonathan Scarlett. Group testing: An information theory perspective. *Foundations and Trends® in Communications and Information Theory*, 15(3-4):196–392, 2019. doi: 10.1561/0100000099.

[110] Claudio M Verdun, Tim Fuchs, Pavol Harar, Dennis Elbrächter, David S Fischer, Julius Berner, Philipp Grohs, Fabian J Theis, and Felix Krahmer. Group testing for sars-cov-2 allows for up to 10-fold efficiency increase across realistic scenarios and testing strategies. *medRxiv*, 2020. doi: 10.1101/2020.04.30.20085290.

[111] Matthew Aldridge. Conservative two-stage group testing. *arXiv*, arXiv:2005.06617, 2020.

[112] Paul Fischer, Norbert Klasner, and Ingo Wegenera. On the cut-off point for combinatorial group testing. *Discrete Applied Mathematics*, 91(1-3):83–92, 1999.

[113] Matthew Aldridge. Rates of adaptive group testing in the linear regime. In *2019 IEEE international symposium on information theory (ISIT)*, pages 236–240. IEEE, 2019.

[114] Matthew Aldridge. Individual testing is optimal for nonadaptive group testing in the linear regime. *IEEE Transactions on Information Theory*, 65(4):2058–2061, 2018. doi: 10.1109/TIT.2018.2873136.

[115] Idan Yelin, Noga Aharony, Einat Shaer Tamar, Amir Argoetti, Esther Messer, Dina Berenbaum, Einat Shafran, Areen Kuzli, Nagham Gandali, Omer Shkedi, Tamar Hashimshony, Yael Mandel-Gutfreund, Michael Halberthal, Yuval Geffen, Moran Szwarcwort-Cohen, and Roy Kishony. Evaluation of COVID-19 RT-qPCR Test in Multi sample Pools. *Clinical Infectious Diseases*, 2020. doi: 10.1093/cid/ciaa531.

[116] Roni Ben-Ami, Agnes Klochendler, Matan Seidel, Tal Sido, Ori Gurel-Gurevich, Moran Yassour, Eran Meshorer, Gil Benedek, Irit Fogel, Esther Oiknine-Djian, et al. Large-scale implementation of pooled

rna extraction and rt-pcr for sars-cov-2 detection. *Clinical Microbiology and Infection*, 26(9):1248–1253, 2020. doi: https://doi.org/10.1016/j.cmi.2020.06.009.

[117] So Yeon Kim, Jaehyeon Lee, Heungsup Sung, Hyukmin Lee, Myung Guk Han, Cheon Kwon Yoo, Sang Won Lee, and Ki Ho Hong. Pooling upper respiratory specimens for rapid mass screening of covid-19 by real-time rt-pcr. *Emerging infectious diseases*, 26(10):2469, 2020. doi: 10.3201/eid2610.201955.

[118] Allen C. Bateman, Shanna Mueller, Kyley Guenther, and Peter Shult. Assessing the dilution effect of specimen pooling on the sensitivity of sars-cov-2 pcr tests. *Journal of Medical Virology*, 93(3): 1568–1572, 2020. doi: 10.1002/jmv.26519.

[119] Leon Mutesa, Pacifique Ndishimye, Yvan Butera, Jacob Souopgui, Annette Uwineza, Robert Rutayisire, Ella Larissa Ndoricimpaye, Emile Musoni, Nadine Rujeni, Thierry Nyatanyi, et al. A pooled testing strategy for identifying sars-cov-2 at low prevalence. *Nature*, pages 1–8, 2020. doi: 10.1038/s41586-020-2885-5.

[120] Nicolas Thierry-Mieg. A new pooling strategy for high-throughput screening: the shifted transversal design. *BMC bioinformatics*, 7(1):28, 2006. doi: 10.1186/1471-2105-7-28.

[121] C. L. Chan, P. H. Che, S. Jaggi, and V. Saligrama. Non-adaptive probabilistic group testing with noisy measurements: Near-optimal bounds with efficient algorithms. In *2011 49th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1832–1839, 2011. doi: 10.1109/Allerton.2011.6120391.

[122] Sabyasachi Ghosh, Rishi Agarwal, Mohammad Ali Rehan, Shreya Pathak, Pratyush Agarwal, Yash Gupta, Sarthak Consul, Nimay Gupta, Ritesh Goenka, Ajit Rajwade, et al. A compressed sensing approach to pooled rt-pcr testing for covid-19 detection. *IEEE Open Journal of Signal Processing*, 2: 248–264, 2021.

[123] Hsin-Po Wang, Ryan Gabrys, and Alexander Vardy. Tropical group testing. *IEEE Transactions on Information Theory*, 2023.

[124] Chantal BF Vogels, Anderson F Brito, Anne L Wyllie, Joseph R Fauver, Isabel M Ott, Chaney C Kalinich, Mary E Petrone, Arnau Casanovas-Massana, M Catherine Muenker, Adam J Moore, et al. Analytical sensitivity and efficiency comparisons of sars-cov-2 rt–qpcr primer–probe sets. *Nature microbiology*, 5(10):1299–1305, 2020.

[125] David P Clark and Nanette J Pazdernik. *Molecular biology*. Elsevier, 2012.

[126] Sabyasachi Ghosh, Rishi Agarwal, Mohammad Ali Rehan, Shreya Pathak, Pratyush Agrawal, Yash Gupta, Sarthak Consul, Nimay Gupta, Ritika Goyal, Ajit Rajwade, et al. A compressed sensing approach to group-testing for covid-19 detection. *arXiv*, arXiv:2005.07895, 2020.

[127] Sonia N Rao, Davide Manissero, Victoria R Steele, and Josep Pareja. A narrative systematic review of the clinical utility of cycle threshold values in the context of covid-19. *Infectious diseases and therapy*, pages 1–14, 2020. doi: 10.1007/s40121-020-00324-3.

[128] Renan Lyra Miranda, Alexandro Guterres, Carlos Henrique de Azeredo Lima, Paulo Niemeyer Filho, and Mônica R. Gadelha. Misinterpretation of viral load in covid-19 clinical outcomes. *Virus Research*, 296:198340, 2021. doi: https://doi.org/10.1016/j.virusres.2021.198340.

[129] Kok-Siong Poon and Nancy Wen-Sim Tee. Caveats of Reporting Cycle Threshold Values from Severe Acute Respiratory Syndrome Coronavirus 2 (SARS-CoV-2) Qualitative Polymerase Chain Reaction Assays: A Molecular Diagnostic Laboratory Perspective. *Clinical Infectious Diseases*, 2020. doi: 10.1093/cid/ciaa1399.

[130] Quest Diagnostics Incorporated. Sars-cov-2 rna, qualitative real-time rt-pcr (test code 39433), 2023. URL `https://www.fda.gov/media/136231/download`.

[131] W. K. Roth, M. P. Busch, A. Schuller, S. Ismay, A. Cheng, C. R. Seed, C. Jungbauer, P. M. Minsk, D. Sondag-Thull, S. Wendel, J. E. Levi, M. Fearon, G. Delage, Y. Xie, I. Jukic, P. Turek, H. Ullum, V. Tefanova, M. Tilk, R. Reimal, J. Castren, M. Naukkarinen, A. Assal, C. Jork, M. K. Hourfar, P. Michel, R. Offergeld, L. Pichl, M. Schmidt, V. Schottstedt, E. Seifried, F. Wagner, M. Weber-Schehl, C. Politis, C. K. Lin, W. C. Tsoi, J. O'Riordan, A. Gottreich, E. Shinar, V. Yahalom, C. Velati, M. Satake, N. Sanad, I. Sisene, A. H. Bon, M. Koppelmann, P. Flanagan, O. Flesland, E. Brojer, M. Letowska, F. Nascimento, E. Zhiburt, S. S. Chua, D. Teo, S. Levicnik Stezinar, M. Vermeulen, R. Reddy, Q. Park, E. Castro, A. Eiras, I. Gonzales Fraile, P. Torres, B. Ekermo, C. Niederhauser, H. Chen, S. Oota, L. J. Brant, R. Eglin, L. Jarvis, L. Mohabir, J. Brodsky, G. Foster, C. Jennings, E. Notari, S. Stramer, D. Kessler, C. Hillyer, H. Kamel, L. Katz, C. Taylor, S. Panzer, and H.W. Reesink. International survey on NAT testing of blood donations: expanding implementation and yield from 1999 to 2009. *Vox Sanguinis*, 102(1):82–90, 2012. ISSN 0042-9007. doi: 10.1111/j.1423-0410.2011.01506.x.

[132] Antonio Aguilera, Sara Pereira, Ana Fuentes, Adolfo de Salazar, Rocío Trastoy, Daniel Navarro, Camila A Picchio, Jeffrey V Lazarus, and Federico García. Pooling samples for hepatitis C RNA

detection. *The Lancet Gastroenterology & Hepatology*, 6(8):608–609, 2021. ISSN 2468-1253. doi: 10.1016/s2468-1253(21)00217-x.

[133] Ashfaq Ahmed. Covid-19: Pakistan starts 'pool testing' for coronavirus in a bid to expedite the process. 2020. URL https://gulfnews.com/world/asia/pakistan/covid-19-pakistan-starts-pool-testing-for-coronavirus-in-a-bid-to-expedite-the-process-1.71457097.

[134] Nicoletta Lanese. Wuhan tested millions of people for covid-19 in just days. could us cities do the same? 2020. URL https://www.livescience.com/pooled-sampling-covid19-in-wuhan-and-us-cities.html.

[135] Leon Mutesa. Rwanda's covid-19 pool testing: a savvy option where there's low viral prevalence., 2020. URL https://theconversation.com/rwandas-covid-19-pool-testing-a-savvy-option-where-theres-low-viral-prevalence-141704.

[136] Kwak Sung-sun. Korea considers testing pooled samples for vulnerable groups, 2020. URL https://www.koreabiomed.com/news/articleView.html?idxno=7966.

[137] PHE. Standard operating procedure: Pooling of asymptomatic SARS COV-2 COVID-19 samples for (PCR/or other) testing., 9 2020. URL https://www.england.nhs.uk/coronavirus/wp-content/uploads/sites/52/2020/09/C0777-sample-pooling-sop-v1.pdf.

[138] CDC. Interim Guidance for Use of Pooling Procedures in SARS-CoV-2 Diagnostic, Screening, and Surveillance Testing, 10 2020. URL https://www.cdc.gov/coronavirus/2019-ncov/lab/pooling-procedures.html.

[139] Kirstin I Oliveira Roster, Stephen M Kissler, Enoma Omoregie, Jade C Wang, Helly S Amin, Steve Di Lonardo, Scott Hughes, and Yonatan Grad. Surveillance strategies for the detection of new SARS-CoV-2 variants across epidemiological contexts. *medRxiv*, May 2023. doi: 10.1101/2023.05.09.23289744.

[140] Alvin X Han, Amy Toporowski, Jilian A Sacks, Mark D Perkins, Sylvie Briand, Maria van Kerkhove, Emma Hannay, Sergio Carmona, Bill Rodriguez, Edyth Parker, et al. SARS-CoV-2 diagnostic testing rates determine the sensitivity of genomic surveillance programs. *Nature Genetics*, 55(1):26–33, 2023. doi: 10.1038/s41588-022-01267-w.

[141] Jacqueline M Hughes-Oliver and William H Swallow. A two-stage adaptive group-testing procedure for estimating small proportions. *Journal of the American Statistical Association*, 89(427):982–993, 1994.

[142] Pritha Guha, Apratim Guha, and Tathagata Bandyopadhyay. Application of pooled testing in estimating the prevalence of covid-19. *Health Services and Outcomes Research Methodology*, 22(2):163–191, 2022.

[143] Ngoc T Nguyen, Ebru K Bish, and Douglas R Bish. Optimal pooled testing design for prevalence estimation under resource constraints. *Omega*, 105:102504, 2021.

[144] M Hung and William H Swallow. Robustness of group testing in the estimation of proportions. *Biometrics*, 55(1):231–237, 1999.

[145] JP Martins, R Santos, and M Felgueiras. Estimation of prevalence in rare disease using pooled samples. In *Journal of Physics: Conference Series*, volume 1564, page 012028. IOP Publishing, 2020.

[146] Joao Paulo Martins, Miguel Felgueiras, and Rui Santos. Estimation through array-based group tests. *REVSTAT-Statistical Journal*, 15(4):487–500, 2017.

[147] João Paulo Martins, Rui Santos, and Miguel Felgueiras. A maximum likelihood estimator for the prevalence rate using pooled sample tests. In *Theory and Practice of Risk Assessment: ICRA 5, Tomar, Portugal, 2013*, pages 99–110. Springer, 2015.

[148] João Paulo Martins, Miguel Felgueiras, and Rui Santos. Three-dimensional array-based group testing algorithms with one-stage. In *AIP Conference Proceedings*, volume 1702. AIP Publishing, 2015.

[149] Md S Warasi, Laura L Hungerford, and Kevin Lahmers. Optimizing pooled testing for estimating the prevalence of multiple diseases. *Journal of Agricultural, Biological and Environmental Statistics*, 27(4): 713–727, 2022.

[150] Md S Warasi, Joshua M Tebbs, Christopher S McMahan, and Christopher R Bilder. Estimating the prevalence of two or more diseases using outcomes from multiplex group testing. *Biometrical Journal*, 65(7):2200270, 2023.

[151] Mireia Coscolla and Sebastien Gagneux. Consequences of genomic diversity in mycobacterium tuberculosis. In *Seminars in immunology*, volume 26, pages 431–444. Elsevier, 2014.

[152] Olivier Tenaillon, David Skurnik, Bertrand Picard, and Erick Denamur. The population genetics of commensal escherichia coli. *Nature reviews microbiology*, 8(3):207–217, 2010.

[153] Lee W Riley. Distinguishing pathovars from nonpathovars: Escherichia coli. *Microbiology spectrum*, 8 (4):10–1128, 2020.

[154] Ismael L Hernández-González, Valeria Mateo-Estrada, and Santiago Castillo-Ramirez. The promiscuous and highly mobile resistome of acinetobacter baumannii. *Microbial Genomics*, 8(1):000762, 2022.

[155] Jeong Ho Jeon, Kyung-Min Jang, Jung Hun Lee, Lin-Woo Kang, and Sang Hee Lee. Transmission of antibiotic resistance genes through mobile genetic elements in acinetobacter baumannii and gene-transfer prevention. *Science of the Total Environment*, 857:159497, 2023.

[156] World Health Organization. Who publishes list of bacteria for which new antibiotics are urgently needed. `https://www.who.int/news/item/27-02-2017-who-publishes-list-of-bacteria-for-which-new-antibiotics-are-urgently-needed`. Accessed: 11-09-2023.

[157] Ellis L Torrance, Corey Burton, Awa Diop, and Louis-Marie Bobay. Evolution of homologous recombination rates across bacteria. *Proceedings of the National Academy of Sciences*, 121(18):e2316302121, 2024.

[158] Frederick R Blattner, Guy Plunkett III, Craig A Bloch, Nicole T Perna, Valerie Burland, Monica Riley, Julio Collado-Vides, Jeremy D Glasner, Christopher K Rode, George F Mayhew, et al. The complete genome sequence of escherichia coli k-12. *science*, 277(5331):1453–1462, 1997.

[159] Duccio Medini, Claudio Donati, Hervé Tettelin, Vega Masignani, and Rino Rappuoli. The microbial pan-genome. *Current opinion in genetics & development*, 15(6):589–594, 2005.

[160] Michael A Savageau. Escherichia coli habitats, cell types, and molecular mechanisms of gene control. *The american naturalist*, 122(6):732–744, 1983.

[161] Nicole T Perna, Guy Plunkett III, Valerie Burland, Bob Mau, Jeremy D Glasner, Debra J Rose, George F Mayhew, Peter S Evans, Jason Gregor, Heather A Kirkpatrick, et al. Genome sequence of enterohaemorrhagic escherichia coli o157: H7. *Nature*, 409(6819):529–533, 2001.

[162] David A Rasko, MJ Rosovitz, Garry SA Myers, Emmanuel F Mongodin, W Florian Fricke, Pawel Gajer, Jonathan Crabtree, Mohammed Sebaihia, Nicholas R Thomson, Roy Chaudhuri, et al. The pangenome structure of escherichia coli: comparative genomic analysis of e. coli commensal and pathogenic isolates. *Journal of bacteriology*, 190(20):6881–6893, 2008.

[163] Rolf S Kaas, Carsten Friis, David W Ussery, and Frank M Aarestrup. Estimating variation within the genes and inferring the phylogeny of 186 sequenced diverse escherichia coli genomes. *BMC genomics*, 13:1–13, 2012.

[164] Zhi-Kai Yang, Hao Luo, Yanming Zhang, Baijing Wang, and Feng Gao. Pan-genomic analysis provides novel insights into the association of e. coli with human host and its minimal genome. *Bioinformatics*, 35(12):1987–1991, 2019.

[165] T Seeman. mlst. URL `https://github.com/tseemann/mlst`. Accessed: 13-12-2024.

[166] Thomas Madden. The blast sequence analysis tool. *The NCBI handbook*, 2(5):425–436, 2013.

[167] Sebastien Gagneux. Ecology and evolution of mycobacterium tuberculosis. *Nature Reviews Microbiology*, 16(4):202–213, 2018.

[168] World Health Organization. Global tuberculosis report 2023, 2023. URL `https://www.who.int/publications/i/item/9789240083851`.

[169] Daniela Brites and Sebastien Gagneux. Old and new selective pressures on mycobacterium tuberculosis. *Infection, genetics and evolution*, 12(4):678–685, 2012.

[170] Mark Achtman. Evolution, population structure, and phylogeography of genetically monomorphic bacterial pathogens. *Annu. Rev. Microbiol.*, 62(1):53–70, 2008.

[171] Rosalino Vázquez-López, Sandra Georgina Solano-Gálvez, Juan José Juárez Vignon-Whaley, Jorge Andrés Abello Vaamonde, Luis Andrés Padró Alonzo, Andrés Rivera Reséndiz, Mauricio Muleiro Álvarez, Eunice Nabil Vega López, Giorgio Franyuti-Kelly, Diego Abelardo Álvarez-Hernández, et al. Acinetobacter baumannii resistance: a real challenge for clinicians. *Antibiotics*, 9(4):205, 2020.

[172] Amy K Cain and Mehrad Hamidian. Portrait of a killer: Uncovering resistance mechanisms and global spread of acinetobacter baumannii. *PLoS Pathogens*, 19(8):e1011520, 2023.

[173] Lucia Graña-Miraglia, Benjamin A Evans, Luis E López-Jácome, Melissa Hernández-Durán, Claudia Adriana Colín-Castro, Patricia Volkow-Fernández, Miguel A Cevallos, Rafael Franco-Cendejas, and Santiago Castillo-Ramírez. Origin of oxa-23 variant oxa-239 from a recently emerged lineage of acinetobacter baumannii international clone v. *Msphere*, 5(1):10–1128, 2020.

[174] Santiago Castillo-Ramírez. The importance of acinetobacter baumannii from non-human sources. *The Lancet Microbe*, 4(10):e761–e762, 2023.

[175] Sergio G Bartual, Harald Seifert, Corinna Hippler, M Angeles Domínguez Luzon, Hilmar Wisplinghoff, and Francisco Rodríguez-Valera. Development of a multilocus sequence typing scheme for characterization of clinical isolates of acinetobacter baumannii. *Journal of clinical microbiology*, 43(9):4382–4390, 2005.

[176] Gregory T Jeffers, James L Bruce, Patrick L McDonough, Janet Scarlett, Kathryn J Boor, and Martin Wiedmann. Comparative genetic characterization of listeria monocytogenes isolates from human and animal listeriosis cases. *Microbiology*, 147(5):1095–1104, 2001.

[177] Mylène M Maury, Yu-Huan Tsai, Caroline Charlier, Marie Touchon, Viviane Chenal-Francisque, Alexandre Leclercq, Alexis Criscuolo, Charlotte Gaultier, Sophie Roussel, Anne Brisabois, et al. Uncovering listeria monocytogenes hypervirulence by harnessing its biodiversity. *Nature genetics*, 48(3): 308–313, 2016.

[178] Mylène M Maury, Hélène Bracq-Dieye, Lei Huang, Guillaume Vales, Morgane Lavina, Pierre Thouvenot, Olivier Disson, Alexandre Leclercq, Sylvain Brisse, and Marc Lecuit. Hypervirulent listeria monocytogenes clones' adaption to mammalian gut accounts for their association with dairy products. *Nature Communications*, 10(1):2488, 2019.

[179] Michael J Gray, Ruth N Zadoks, Esther D Fortes, Belgin Dogan, Steven Cai, Yuhuan Chen, Virginia N Scott, David E Gombas, Kathryn J Boor, and Martin Wiedmann. Listeria monocytogenes isolates from foods and humans form distinct but overlapping populations. *Applied and environmental microbiology*, 70(10):5833–5841, 2004.

[180] V Ferreira, M Wiedmann, P Teixeira, and MJ Stasiewicz. Listeria monocytogenes persistence in food-associated environments: epidemiology, strain characteristics, and implications for public health. *Journal of food protection*, 77(1):150–170, 2014.

[181] Patricia Hingston, Jessica Chen, Kevin Allen, Lisbeth Truelstrup Hansen, and Siyun Wang. Strand specific rna-sequencing and membrane lipid profiling reveals growth phase-dependent cold stress response mechanisms in listeria monocytogenes. *PLoS One*, 12(6):e0180123, 2017.

[182] Françoise Jaureguy, Luce Landraud, Virginie Passet, Laure Diancourt, Eric Frapy, Ghislaine Guigon, Etienne Carbonnelle, Olivier Lortholary, Olivier Clermont, Erick Denamur, et al. Phylogenetic and genomic diversity of human bacteremic escherichia coli strains. *BMC genomics*, 9:1–14, 2008.

[183] E Michael Valencia, Katherine A Maki, Jennifer N Dootz, and Jennifer J Barb. Mock community taxonomic classification performance of publicly available shotgun metagenomics pipelines. *Scientific Data*, 11(1):81, 2024.

[184] Leonid N Wasserstein. Markov processes on countable product space describing large systems of automata. *Probl. Pered. Inform*, 5:64–73, 1969.

[185] Roland L Dobrushin. Prescribing a system of random variables by conditional distributions. *Theory of Probability & Its Applications*, 15(3):458–486, 1970.

[186] W James Murdoch, Chandan Singh, Karl Kumbier, Reza Abbasi-Asl, and Bin Yu. Interpretable machine learning: definitions, methods, and applications. *arXiv preprint arXiv:1901.04592*, 2019.

[187] Xiaowen Yang, Yajie Li, Juan Zang, Yexia Li, Pengfei Bie, Yanli Lu, and Qingmin Wu. Analysis of pan-genome to identify the core genes and essential genes of brucella spp. *Molecular genetics and genomics*, 291:905–912, 2016.

[188] Stephen J Callister, Lee Ann McCue, Joshua E Turse, Matthew E Monroe, Kenneth J Auberry, Richard D Smith, Joshua N Adkins, and Mary S Lipton. Comparative bacterial proteomics: analysis of the core genome concept. *PLoS One*, 3(2):e1542, 2008.

[189] Michael A Brockhurst, Ellie Harrison, James PJ Hall, Thomas Richards, Alan McNally, and Craig MacLean. The ecology and evolution of pangenomes. *Current Biology*, 29(20):R1094–R1103, 2019.

[190] Torsten Seemann. Prokka: rapid prokaryotic genome annotation. *Bioinformatics*, 30(14):2068–2069, 2014.

[191] Kazutaka Katoh, Kazuharu Misawa, Kei-ichi Kuma, and Takashi Miyata. Mafft: a novel method for rapid multiple sequence alignment based on fast fourier transform. *Nucleic acids research*, 30(14): 3059–3066, 2002.

[192] Gerry Tonkin-Hill, Neil MacAlasdair, Christopher Ruis, Aaron Weimann, Gal Horesh, John A Lees, Rebecca A Gladstone, Stephanie Lo, Christopher Beaudoin, R Andres Floto, et al. Producing polished prokaryotic pangenomes with the panaroo pipeline. *Genome biology*, 21:1–21, 2020.

[193] Weizhong Li and Adam Godzik. Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences. *Bioinformatics*, 22(13):1658–1659, 2006.

[194] Gerry Tonkin-Hill, John A Lees, Stephen D Bentley, Simon DW Frost, and Jukka Corander. Rhierbaps: an r implementation of the population clustering algorithm hierbaps. *Wellcome open research*, 3, 2018.

[195] Ben Langmead and Steven L Salzberg. Fast gapped-read alignment with bowtie 2. *Nature methods*, 9 (4):357–359, 2012.

[196] François Margot. Symmetry in integer linear programming. *50 Years of Integer Programming 1958-2008: From the Early Years to the State-of-the-Art*, pages 647–686, 2009.

[197] Anil Chaturvedi, Paul E Green, and J Douglas Caroll. K-modes clustering. *Journal of classification*, 18:35–55, 2001.

[198] Chirag Jain, Luis M Rodriguez-R, Adam M Phillippy, Konstantinos T Konstantinidis, and Srinivas Aluru. High throughput ani analysis of 90k prokaryotic genomes reveals clear species boundaries. *Nature communications*, 9(1):5114, 2018.

[199] Caroline M Stott and Louis-Marie Bobay. Impact of homologous recombination on core genome phylogenies. *BMC genomics*, 21:1–10, 2020.

[200] Dai Yoshimura, Rei Kajitani, Yasuhiro Gotoh, Katsuyuki Katahira, Miki Okuno, Yoshitoshi Ogura, Tetsuya Hayashi, and Takehiko Itoh. Evaluation of snp calling methods for closely related bacterial isolates and a novel high-accuracy pipeline: Bactsnp. *Microbial genomics*, 5(5):e000261, 2019.

[201] T Seemann. Snippy: rapid haploid variant calling and core snp phylogeny. *GitHub. Available at: github. com/tseemann/snippy*, 2015.

[202] Mario Juhas, Leo Eberl, and John I Glass. Essence of life: essential genes of minimal genomes. *Trends in cell biology*, 21(10):562–568, 2011.

[203] Marco Fondi and Renato Fani. The horizontal flow of the plasmid resistome: clues from inter-generic similarity networks. *Environmental microbiology*, 12(12):3228–3242, 2010.

[204] Alexander Koeppel, Elizabeth B Perry, Johannes Sikorski, Danny Krizanc, Andrew Warner, David M Ward, Alejandro P Rooney, Evelyne Brambilla, Nora Connor, Rodney M Ratcliff, et al. Identifying the fundamental units of bacterial diversity: a paradigm shift to incorporate ecology into bacterial systematics. *Proceedings of the National Academy of Sciences*, 105(7):2504–2509, 2008.

[205] Joshua E Goldford, Nanxi Lu, Djordje Bajić, Sylvie Estrela, Mikhail Tikhonov, Alicia Sanchez-Gorostiaga, Daniel Segrè, Pankaj Mehta, and Alvaro Sanchez. Emergent simplicity in microbial community assembly. *Science*, 361(6401):469–474, 2018.

[206] Leonard Carlitz, David P Roselle, and Richard A Scoville. Permutations and sequences with repetitions by number of increases. *Journal of Combinatorial Theory*, 1(3):350–374, 1966.

[207] John Riordan and Paul R Stein. Arrangements on chessboards. *Journal of Combinatorial Theory, Series A*, 12(1):72–80, 1972.

[208] Brian Cleary, James A Hay, Brendan Blumenstiel, Maegan Harden, Michelle Cipicchio, Jon Bezney, Brooke Simonton, David Hong, Madikay Senghore, Abdul K Sesay, et al. Using viral load and epidemic

dynamics to optimize pooled testing in resource-constrained settings. *Science translational medicine*, 13(589):eabf1568, 2021.

[209] Jatin Machhi, Jonathan Herskovitz, Ahmed M Senan, Debashis Dutta, Barnali Nath, Maxim D Oleynikov, Wilson R Blomberg, Douglas D Meigs, Mahmudul Hasan, Milankumar Patel, et al. The natural history, pathobiology, and clinical manifestations of sars-cov-2 infections. *Journal of neuroimmune pharmacology*, 15:359–386, 2020.

[210] WHO. Who covid-19 dashboard, 2024. URL `https://data.who.int/dashboards/covid19/deaths`.

[211] Zhiyuan Chen, Andrew S Azman, Xinhua Chen, Junyi Zou, Yuyang Tian, Ruijia Sun, Xiangyanyu Xu, Yani Wu, Wanying Lu, Shijia Ge, et al. Global landscape of sars-cov-2 genomic surveillance and data sharing. *Nature genetics*, 54(4):499–507, 2022.

[212] Patrick GT Walker, Charles Whittaker, Oliver J Watson, Marc Baguelin, Peter Winskill, Arran Hamlet, Bimandra A Djafaara, Zulma Cucunubá, Daniela Olivera Mesa, Will Green, et al. The impact of covid-19 and strategies for mitigation and suppression in low-and middle-income countries. *Science*, 369(6502):413–422, 2020.

[213] Matthias Täufer. Rapid, large-scale, and effective detection of covid-19 via non-adaptive testing. *Journal of Theoretical Biology*, 506:110450, 2020. doi: https://doi.org/10.1016/j.jtbi.2020.110450.

[214] Stan Development Team. RStan: the R interface to Stan. URL `https://mc-stan.org/`. R package version 2.18.1.

[215] R Dahmani, S Boogmans, A Meijs, and D Van den Berg. Paintings-from-polygons: simulated annealing. In *International Conference on Computational Creativity (ICCC 2020)*, 2020.

[216] Andreas Deckert, Till Bärnighausen, and Nicholas NA Kyei. Simulation of pooled-sample analysis strategies for covid-19 mass testing. *Bulletin of the World Health Organization*, 98(9):590, 2020.

[217] Koen B Pouwels, Thomas House, Emma Pritchard, Julie V Robotham, Paul J Birrell, Andrew Gelman, Karina-Doris Vihta, Nikola Bowers, Ian Boreham, Heledd Thomas, et al. Community prevalence of SARS-CoV-2 in England from April to November, 2020: results from the ONS Coronavirus Infection Survey. *The Lancet Public Health*, 6(1):e30–e38, 2021. doi: 10.1016/S2468-2667(20)30282-6.

[218] Radhey S Gupta. Identification of conserved indels that are useful for classification and evolutionary studies. In *Methods in microbiology*, volume 41, pages 153–182. Elsevier, 2014.

[219] David Danko, Daniela Bezdan, Evan E Afshin, Sofia Ahsanuddin, Chandrima Bhattacharya, Daniel J Butler, Kern Rei Chng, Daisy Donnellan, Jochen Hecht, Katelyn Jackson, et al. A global metagenomic map of urban microbiomes and antimicrobial resistance. *Cell*, 184(13):3376–3393, 2021.

[220] Runzhi Zhang, Alejandro R Walker, and Susmita Datta. Unraveling city-specific signature and identifying sample origin locations for the data from camda metasub challenge. *Biology Direct*, 16:1–14, 2021.

[221] Kieran Hand. Antibiotic stewardship. *Clinical Medicine*, 13(5):499, 2013.

[222] Csaba Pál, Balázs Papp, and Viktória Lázár. Collateral sensitivity of antibiotic-resistant microbes. *Trends in microbiology*, 23(7):401–407, 2015.

[223] Daniel Nichol, Joseph Rutter, Christopher Bryant, Andrea M Hujer, Sai Lek, Mark D Adams, Peter Jeavons, Alexander RA Anderson, Robert A Bonomo, and Jacob G Scott. Antibiotic collateral sensitivity is contingent on the repeatability of evolution. *Nature communications*, 10(1):334, 2019.

[224] Sally R Partridge, Stephen M Kwong, Neville Firth, and Slade O Jensen. Mobile genetic elements associated with antimicrobial resistance. *Clinical microbiology reviews*, 31(4):10–1128, 2018.

[225] Denis Bertrand, Jim Shaw, Manesh Kalathiyappan, Amanda Hui Qi Ng, M Senthil Kumar, Chenhao Li, Mirta Dvornicic, Janja Paliska Soldo, Jia Yu Koh, Chengxuan Tong, et al. Hybrid metagenomic assembly enables high-resolution analysis of resistance determinants and mobile elements in human microbiomes. *Nature biotechnology*, 37(8):937–944, 2019.

[226] Jingwen Yan, Shannon L Risacher, Li Shen, and Andrew J Saykin. Network approaches to systems biology analysis of complex disease: integrative methods for multi-omics data. *Briefings in bioinformatics*, 19(6):1370–1381, 2018.

[227] Matteo Bersanelli, Ettore Mosca, Daniel Remondini, Enrico Giampieri, Claudia Sala, Gastone Castellani, and Luciano Milanesi. Methods for the integration of multi-omics data: mathematical aspects. *BMC bioinformatics*, 17:167–177, 2016.

[228] Asher Mullard. Fda approves second microbiome-based c. difficile therapy. *Nature reviews. Drug Discovery*, 2023.

[229] Richard R Stein, Vanni Bucci, Nora C Toussaint, Charlie G Buffie, Gunnar Rätsch, Eric G Pamer, Chris Sander, and Joao B Xavier. Ecological modeling from time-series inference: insight into dynamics and stability of intestinal microbiota. *PLoS computational biology*, 9(12):e1003388, 2013.

[230] Mathew Stracy, Olga Snitser, Idan Yelin, Yara Amer, Miriam Parizade, Rachel Katz, Galit Rimler, Tamar Wolf, Esma Herzel, Gideon Koren, et al. Minimizing treatment-induced emergence of antibiotic resistance in bacterial infections. *Science*, 375(6583):889–894, 2022.

[231] John Penders, Ellen E Stobberingh, Paul HM Savelkoul, and Petra Wolffs. The human microbiome as a reservoir of antimicrobial resistance. *Frontiers in microbiology*, 4:87, 2013.

[232] Gerard D Wright. The antibiotic resistome. *Expert opinion on drug discovery*, 5(8):779–788, 2010.

[233] David Deutsch. *The beginning of infinity: Explanations that transform the world*. penguin uK, 2011.

[234] Juan M Gonzalez, Elena Puerta-Fernández, Margarida M Santana, and Bhagwan Rekadwad. On a non-discrete concept of prokaryotic species. *Microorganisms*, 8(11):1723, 2020.

[235] M Luz Calle. Statistical analysis of metagenomics data. *Genomics & informatics*, 17(1), 2019.

[236] Chirag Jain, Luis M Rodriguez-R, Adam M Phillippy, Konstantinos T Konstantinidis, and Srinivas Aluru. High throughput ani analysis of 90k prokaryotic genomes reveals clear species boundaries. *Nature communications*, 9(1):5114, 2018.

[237] Alejandro Caro-Quintero and Konstantinos T Konstantinidis. Bacterial species may exist, metagenomics reveal. *Environmental microbiology*, 14(2):347–355, 2012.

[238] Mark Wilson. *Wandering significance: An essay on conceptual behavior*. Oxford University Press, 2006.

[239] Aaron Novick and W Ford Doolittle. 'species' without species. *Studies in History and Philosophy of Science Part A*, 87:72–80, 2021.

[240] William P Hanage. Not so simple after all: bacteria, their population genetics, and recombination. *Cold Spring Harbor perspectives in biology*, 8(7):a018069, 2016.

[241] Nisreen A Alwan. Surveillance is underestimating the burden of the covid-19 pandemic. *Lancet (London, England)*, 396(10252):e24, 2020.

[242] Vincent Brault, Bastien Mallein, and Jean-François Rupprecht. Group testing as a strategy for covid-19 epidemiological monitoring and community surveillance. *PLoS computational biology*, 17(3):e1008726, 2021.

[243] Jiali Yu, Yiduo Huang, and Zuo-Jun Shen. Optimizing and evaluating pcr-based pooled screening during covid-19 pandemics. *Scientific reports*, 11(1):21460, 2021.

[244] Jingyi Zhang and Lenwood S Heath. Adaptive group testing strategy for infectious diseases using social contact graph partitions. *Scientific Reports*, 13(1):12102, 2023.

[245] Jan M Brauner, Sören Mindermann, Mrinank Sharma, David Johnston, John Salvatier, Tomáš Gavenčiak, Anna B Stephenson, Gavin Leech, George Altman, Vladimir Mikulik, et al. Inferring the effectiveness of government interventions against covid-19. *Science*, 371(6531):eabd9338, 2021.

[246] Hooman Zabeti, Nick Dexter, Amir Hosein Safari, Nafiseh Sedaghat, Maxwell Libbrecht, and Leonid Chindelevitch. Ingot-dr: an interpretable classifier for predicting drug resistance in m. tuberculosis. *Algorithms for Molecular Biology*, 16(1):17, 2021.

[247] Katherine A Innamorati, Joshua P Earl, Surya D Aggarwal, Garth D Ehrlich, and N Luisa Hiller. The bacterial guide to designing a diversified gene portfolio. *The pangenome: Diversity, dynamics and evolution of genomes*, pages 51–87, 2020.

Appendix A

Supplementary Tables and Figures for Chapter II

Sample Nr	E. coli Sample Reads	A. baumannii Sample Reads	M. tuberculosis Sample Reads	L. monocytogenes Sample Reads	Number of Strains	Coverage	Relative Abundance Profile	Same or Different Cluster	Strains in Reference Database
1	18722	13378	14643	10002	1	1	Equal	Same	Yes
2	17110	12812	14689	9760	1	1	Equal	Same	Yes
3	17099	11993	14673	10305	1	1	Equal	Same	Yes
4	15262	13882	14643	9718	1	1	Equal	Same	Yes
5	16606	13990	14679	9869	1	1	Equal	Same	Yes
6	92518	67486	72498	49586	1	5	Equal	Same	Yes
7	72628	65167	72780	47829	1	5	Equal	Same	Yes
8	81048	67184	70363	47824	1	5	Equal	Same	Yes
9	81048	67976	73389	48235	1	5	Equal	Same	Yes
10	90421	64651	73213	50162	1	5	Equal	Same	Yes
11	145255	125315	139966	100380	1	10	Equal	Same	Yes
12	171090	133875	124663	97465	1	10	Equal	Same	Yes
13	153716	132020	140220	103615	1	10	Equal	Same	Yes
14	152615	127054	140725	95455	1	10	Equal	Same	Yes
15	171090	129236	145560	99160	1	10	Equal	Same	Yes
16	370070	260660	291000	198650	1	20	Equal	Same	Yes
17	305230	257183	288380	192540	1	20	Equal	Same	Yes
18	370070	239146	292850	194950	1	20	Equal	Same	Yes
19	342180	252595	293542	195400	1	20	Equal	Same	Yes
20	353650	273648	292850	197040	1	20	Equal	Same	Yes
21	72743	64821	72354	49253	2	5	Equal	Same	Yes
22	89569	64450	72620	49329	2	5	Equal	Same	Yes
23	85459	61679	72015	49993	2	5	Equal	Same	Yes
24	72743	66992	72473	49581	2	5	Equal	Same	Yes
25	87777	67343	73650	50378	2	5	Equal	Same	Yes
26	82902	64010	71849	50205	2	5	Equal	Different	Yes
27	77511	66439	72765	47788	2	5	Equal	Different	Yes

Sample Nr	E. coli Sample Reads	A. baumannii Sample Reads	M. tuberculosis Sample Reads	L. monocytogenes Sample Reads	Number of Strains	Coverage	Relative Abundance Profile	Same or Different Cluster	Strains in Reference Database
28	80941	64717	71865	48991	2	5	Equal	Different	Yes
29	79201	65462	73019	50564	2	5	Equal	Different	Yes
30	78201	67593	73170	50933	2	5	Equal	Different	Yes
31	87633	66926	72324	50129	2	5	One Dominant Strain	Same	Yes
32	84577	69118	72507	50560	2	5	One Dominant Strain	Same	Yes
33	78985	67958	72234	49501	2	5	One Dominant Strain	Same	Yes
34	78183	69081	71750	50947	2	5	One Dominant Strain	Same	Yes
35	94411	66180	73362	48993	2	5	One Dominant Strain	Same	Yes
36	89500	64600	72649	50785	2	5	One Dominant Strain	Different	Yes
37	88516	67716	72826	49814	2	5	One Dominant Strain	Different	Yes
38	84638	68704	73445	48283	2	5	One Dominant Strain	Different	Yes
39	81970	65916	70644	50983	2	5	One Dominant Strain	Different	Yes
40	94214	64670	73152	50751	2	5	One Dominant Strain	Different	Yes
41	170916	134647	147075	99189	2	10	Equal	Same	Yes
42	152158	119041	146690	101279	2	10	Equal	Same	Yes
43	182885	135667	142085	98599	2	10	Equal	Same	Yes
44	188609	129528	146713	100664	2	10	Equal	Same	Yes
45	188608	129148	145178	101100	2	10	Equal	Same	Yes
46	179324	131403	146440	98228	2	10	Equal	Different	Yes
47	170554	126219	142635	101137	2	10	Equal	Different	Yes
48	157356	127148	143576	96494	2	10	Equal	Different	Yes
49	160959	136921	146152	99558	2	10	Equal	Different	Yes
50	167114	130284	142638	101305	2	10	Equal	Different	Yes
51	170321	137354	145544	96603	2	10	One Dominant Strain	Same	Yes
52	166892	132529	147193	96189	2	10	One Dominant Strain	Same	Yes
53	145347	130575	145197	101145	2	10	One Dominant Strain	Same	Yes
54	186472	143645	146112	97034	2	10	One Dominant Strain	Same	Yes
55	149633	128634	145414	98664	2	10	One Dominant Strain	Same	Yes

Sample Nr	E. coli Sample Reads	A. baumannii Sample Reads	M. tuberculosis Sample Reads	L. monocytogenes Sample Reads	Number of Strains	Coverage	Relative Abundance Profile	Same or Different Cluster	Strains in Reference Database
56	171974	128984	144842	97075	2	10	One Dominant Strain	Different	Yes
57	171833	136181	145453	97681	2	10	One Dominant Strain	Different	Yes
58	181756	135612	141461	98805	2	10	One Dominant Strain	Different	Yes
59	150790	131030	145733	99285	2	10	One Dominant Strain	Different	Yes
60	172981	129696	141758	100804	2	10	One Dominant Strain	Different	Yes
61	343615	278618	293425	198285	2	20	Equal	Same	Yes
62	304315	267295	287172	197280	2	20	Equal	Same	Yes
63	290970	255613	277861	202230	2	20	Equal	Same	Yes
64	329325	264971	284070	195645	2	20	Equal	Same	Yes
65	290970	263004	288196	202180	2	20	Equal	Same	Yes
66	366775	273583	290325	205580	2	20	Equal	Different	Yes
67	343325	271459	291985	202935	2	20	Equal	Different	Yes
68	314910	262028	284455	196335	2	20	Equal	Different	Yes
69	337650	266179	293155	201700	2	20	Equal	Different	Yes
70	327455	266664	292782	200200	2	20	Equal	Different	Yes
71	337624	267074	293823	195390	2	20	One Dominant Strain	Same	Yes
72	291246	268850	279897	197242	2	20	One Dominant Strain	Same	Yes
73	335744	271995	288342	192338	2	20	One Dominant Strain	Same	Yes
74	344340	253642	290810	195982	2	20	One Dominant Strain	Same	Yes
75	359888	249040	291002	196294	2	20	One Dominant Strain	Same	Yes
76	333994	269623	291429	203092	2	20	One Dominant Strain	Different	Yes
77	316050	258303	282922	204514	2	20	One Dominant Strain	Different	Yes
78	303364	260958	292260	192804	2	20	One Dominant Strain	Different	Yes
79	331970	272766	290579	202210	2	20	One Dominant Strain	Different	Yes
80	340519	277122	288984	207724	2	20	One Dominant Strain	Different	Yes
81	89466	66938	71064	48794	3	5	Equal	Same	Yes
82	87235	61671	73680	50735	3	5	Equal	Same	Yes
83	87206	69274	73303	49219	3	5	Equal	Same	Yes

Sample Nr	E. coli Sample Reads	A. baumannii Sample Reads	M. tuberculosis Sample Reads	L. monocytogenes Sample Reads	Number of Strains	Coverage	Relative Abundance Profile	Same or Different Cluster	Strains in Reference Database
84	89471	67893	72195	49099	3	5	Equal	Same	Yes
85	89851	67666	69602	48679	3	5	Equal	Same	Yes
86	84317	70352	71332	48677	3	5	Equal	Different	Yes
87	80054	66067	72023	50944	3	5	Equal	Different	Yes
88	83210	66412	71803	49900	3	5	Equal	Different	Yes
89	84828	64864	73085	49985	3	5	Equal	Different	Yes
90	81075	67317	72161	48842	3	5	Equal	Different	Yes
91	89840	63392	72640	48976	3	5	One Dominant Strain	Same	Yes
92	81226	64278	72117	49058	3	5	One Dominant Strain	Same	Yes
93	81875	65077	68944	48677	3	5	One Dominant Strain	Same	Yes
94	90257	66400	72542	51474	3	5	One Dominant Strain	Same	Yes
95	93462	66780	72541	49785	3	5	One Dominant Strain	Same	Yes
96	92490	64818	72352	51097	3	5	One Dominant Strain	Different	Yes
97	90675	68792	72271	48984	3	5	One Dominant Strain	Different	Yes
98	85876	69182	73184	48995	3	5	One Dominant Strain	Different	Yes
99	84082	66188	70386	48341	3	5	One Dominant Strain	Different	Yes
100	83519	67414	72372	49752	3	5	One Dominant Strain	Different	Yes
101	186989	130596	144579	97279	3	10	Equal	Same	Yes
102	165393	135090	146219	95379	3	10	Equal	Same	Yes
103	180025	131876	145803	100039	3	10	Equal	Same	Yes
104	172725	129524	145010	97640	3	10	Equal	Same	Yes
105	171793	122760	146767	100938	3	10	Equal	Same	Yes
106	164041	127583	145278	97748	3	10	Equal	Different	Yes
107	176805	132635	145630	99554	3	10	Equal	Different	Yes
108	178629	128106	146040	99648	3	10	Equal	Different	Yes
109	169510	131457	144169	102304	3	10	Equal	Different	Yes
110	173831	136395	144950	99179	3	10	Equal	Different	Yes
111	155524	120612	144130	97418	3	10	One Dominant Strain	Same	Yes

Sample Nr	E. coli Sample Reads	A. baumannii Sample Reads	M. tuberculosis Sample Reads	L. monocytogenes Sample Reads	Number of Strains	Coverage	Relative Abundance Profile	Same or Different Cluster	Strains in Reference Database
112	155524	135712	146189	99931	3	10	One Dominant Strain	Same	Yes
113	155524	135959	144727	100373	3	10	One Dominant Strain	Same	Yes
114	179275	132535	145742	99833	3	10	One Dominant Strain	Same	Yes
115	165120	131581	143688	97856	3	10	One Dominant Strain	Same	Yes
116	158277	127231	145658	98803	3	10	One Dominant Strain	Different	Yes
117	171596	128166	144632	97056	3	10	One Dominant Strain	Different	Yes
118	183912	132699	146331	102349	3	10	One Dominant Strain	Different	Yes
119	178471	136122	146081	97561	3	10	One Dominant Strain	Different	Yes
120	170043	137408	146164	97872	3	10	One Dominant Strain	Different	Yes
121	344061	259443	291030	198399	3	20	Equal	Same	Yes
122	354620	260080	289200	201759	3	20	Equal	Same	Yes
123	307117	257904	291054	191827	3	20	Equal	Same	Yes
124	307117	263386	294793	200480	3	20	Equal	Same	Yes
125	343194	252641	293717	203275	3	20	Equal	Same	Yes
126	359186	269733	287455	198803	3	20	Equal	Different	Yes
127	352861	266858	288023	197652	3	20	Equal	Different	Yes
128	335108	259797	291083	200807	3	20	Equal	Different	Yes
129	311775	255348	287277	197823	3	20	Equal	Different	Yes
130	316003	263609	291029	202552	3	20	Equal	Different	Yes
131	311047	281875	289999	203410	3	20	One Dominant Strain	Same	Yes
132	311047	283911	294226	199743	3	20	One Dominant Strain	Same	Yes
133	332791	272705	290400	204167	3	20	One Dominant Strain	Same	Yes
134	316611	269616	294465	196127	3	20	One Dominant Strain	Same	Yes
135	323438	270906	288797	203440	3	20	One Dominant Strain	Same	Yes
136	355470	252931	292946	197028	3	20	One Dominant Strain	Different	Yes
137	328306	250212	289262	204093	3	20	One Dominant Strain	Different	Yes
138	310820	264712	283438	193805	3	20	One Dominant Strain	Different	Yes
139	345360	267814	292695	202419	3	20	One Dominant Strain	Different	Yes

Sample Nr	E. coli Sample Reads	A. baumannii Sample Reads	M. tuberculosis Sample Reads	L. monocytogenes Sample Reads	Number of Strains	Coverage	Relative Abundance Profile	Same or Different Cluster	Strains in Reference Database
140	347303	260428	278276	197508	3	20	One Dominant Strain	Different	Yes
141	86647	67233	72270	49256	4	5	Equal	Same	Yes
142	86312	68913	73137	50029	4	5	Equal	Same	Yes
143	85755	66588	71622	50505	4	5	Equal	Same	Yes
144	84276	66397	71358	49942	4	5	Equal	Same	Yes
145	87526	64561	73397	50337	4	5	Equal	Same	Yes
146	88726	66002	71674	49034	4	5	Equal	Different	Yes
147	84741	67779	72353	49492	4	5	Equal	Different	Yes
148	86606	66526	71427	50635	4	5	Equal	Different	Yes
149	80831	65892	72786	50759	4	5	Equal	Different	Yes
150	89740	64530	72227	50106	4	5	Equal	Different	Yes
151	87580	66984	73564	49456	4	5	One Dominant Strain	Same	Yes
152	88024	67086	72268	50011	4	5	One Dominant Strain	Same	Yes
153	88300	66522	73508	48776	4	5	One Dominant Strain	Same	Yes
154	88737	68892	72084	50227	4	5	One Dominant Strain	Same	Yes
155	91847	64802	68869	51539	4	5	One Dominant Strain	Same	Yes
156	82240	64322	72705	50124	4	5	One Dominant Strain	Different	Yes
157	88656	65880	70631	49391	4	5	One Dominant Strain	Different	Yes
158	81502	63288	73248	48467	4	5	One Dominant Strain	Different	Yes
159	77582	69680	72004	48945	4	5	One Dominant Strain	Different	Yes
160	86780	67368	71378	50271	4	5	One Dominant Strain	Different	Yes
161	187773	129226	145610	101495	4	10	Equal	Same	Yes
162	173194	135116	145191	100838	4	10	Equal	Same	Yes
163	167921	134837	142176	102605	4	10	Equal	Same	Yes
164	182676	127402	146290	101623	4	10	Equal	Same	Yes
165	173629	128651	142052	101962	4	10	Equal	Same	Yes
166	160442	136645	144251	98375	4	10	Equal	Different	Yes
167	164021	133375	144644	98587	4	10	Equal	Different	Yes

Sample Nr	E. coli Sample Reads	A. baumannii Sample Reads	M. tuberculosis Sample Reads	L. monocytogenes Sample Reads	Number of Strains	Coverage	Relative Abundance Profile	Same or Different Cluster	Strains in Reference Database
168	159019	131834	144225	101084	4	10	Equal	Different	Yes
169	162470	134463	144936	99076	4	10	Equal	Different	Yes
170	158351	134745	144356	99236	4	10	Equal	Different	Yes
171	187511	130986	138862	101666	4	10	One Dominant Strain	Same	Yes
172	154789	125050	145305	98926	4	10	One Dominant Strain	Same	Yes
173	173304	131324	139144	101025	4	10	One Dominant Strain	Same	Yes
174	183370	129697	144061	99839	4	10	One Dominant Strain	Same	Yes
175	177120	131223	144423	103688	4	10	One Dominant Strain	Same	Yes
176	158504	129160	145238	100498	4	10	One Dominant Strain	Different	Yes
177	155174	130556	146187	102436	4	10	One Dominant Strain	Different	Yes
178	169031	134154	144956	97092	4	10	One Dominant Strain	Different	Yes
179	149609	129861	145867	99548	4	10	One Dominant Strain	Different	Yes
180	159916	139459	147023	99516	4	10	One Dominant Strain	Different	Yes
181	350443	268151	291253	201202	4	20	Equal	Same	Yes
182	350790	269128	288403	194155	4	20	Equal	Same	Yes
183	366488	263701	289691	197680	4	20	Equal	Same	Yes
184	362206	248863	289467	206449	4	20	Equal	Same	Yes
185	344768	267442	290139	194351	4	20	Equal	Same	Yes
186	335221	256556	293297	196958	4	20	Equal	Different	Yes
187	331673	270228	286463	195794	4	20	Equal	Different	Yes
188	334975	255863	289616	196444	4	20	Equal	Different	Yes
189	330472	267579	290151	198104	4	20	Equal	Different	Yes
190	327533	271925	291091	200005	4	20	Equal	Different	Yes
191	305472	268713	294143	201901	4	20	One Dominant Strain	Same	Yes
192	326575	285678	293797	191853	4	20	One Dominant Strain	Same	Yes
193	339424	264240	292278	195803	4	20	One Dominant Strain	Same	Yes
194	353197	266912	281819	196149	4	20	One Dominant Strain	Same	Yes
195	340591	261798	294681	198410	4	20	One Dominant Strain	Same	Yes

Sample Nr	E. coli Sample Reads	A. baumannii Sample Reads	M. tuberculosis Sample Reads	L. monocytogenes Sample Reads	Number of Strains	Coverage	Relative Abundance Profile	Same or Different Cluster	Strains in Reference Database
196	327262	266803	285891	193173	4	20	One Dominant Strain	Different	Yes
197	310063	265346	282898	199213	4	20	One Dominant Strain	Different	Yes
198	348444	244029	282818	205019	4	20	One Dominant Strain	Different	Yes
199	361411	265112	282698	200376	4	20	One Dominant Strain	Different	Yes
200	364486	276794	282077	199069	4	20	One Dominant Strain	Different	Yes
201	174710	135482	143856	97105	1	10	Equal	Different	No
202	180845	139093	144974	100330	1	10	Equal	Different	No
203	155060	132674	146210	101515	1	10	Equal	Different	No
204	183210	133599	145950	96835	1	10	Equal	Different	No
205	162100	126118	145155	95810	1	10	Equal	Different	No
206	178715	133641	146616	102091	2	10	Equal	Different	No
207	155656	133554	143473	100119	2	10	Equal	Different	No
208	182599	131880	145103	99459	2	10	Equal	Different	No
209	182915	135771	145376	97406	2	10	Equal	Different	No
210	180051	134279	144823	100122	2	10	Equal	Different	No

Table A1: Details of the 840 created samples. Each row corresponds to four samples, one for *E. coli*, one for *A. baumannii*, one for *M. tuberculosis*, and one for *L. monocytogenes*, that were created with the same parameters. The sample reads column of each species indicates the total number of paired end reads in the sample. Coverage indicates the total coverage across all strains of the corresponding species in a sample. Relative abundance profile column indicates whether all strains in a sample had equal abundance (Equal) or whether one strain had 0.8 relative abundance with the remaining strains having equal parts of the remaining 0.2 relative abundance (One Dominant Strain). The same or different cluster cluster column describes whether all the strains in a sample were sampled from the same cluster at the 100 cluster level, or from different ones. The strains in reference database column indicates whether the strains in the sample were also in the reference database provided to the tools (Yes) or not (No).

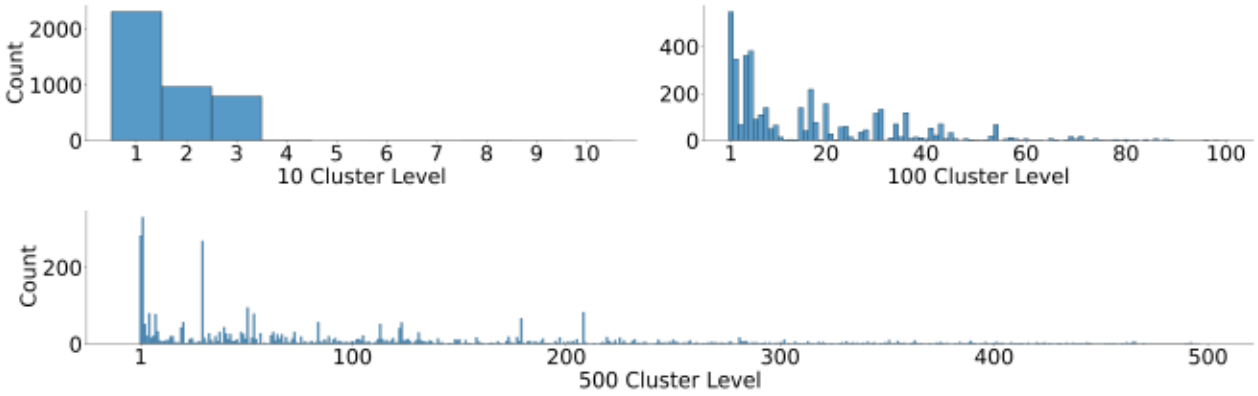


Figure A1: Count of assemblies of *E. coli* assigned to each cluster at the three clustering levels (assemblies clustered into 10, 100, and 500 clusters, labeled as 10 Cluster Level, 100 Cluster Level, and 500 Cluster Level, respectively). Cluster numbers are arbitrary and numbering increases left to right on the x-axis (some x-axis ticks were left out for visibility).

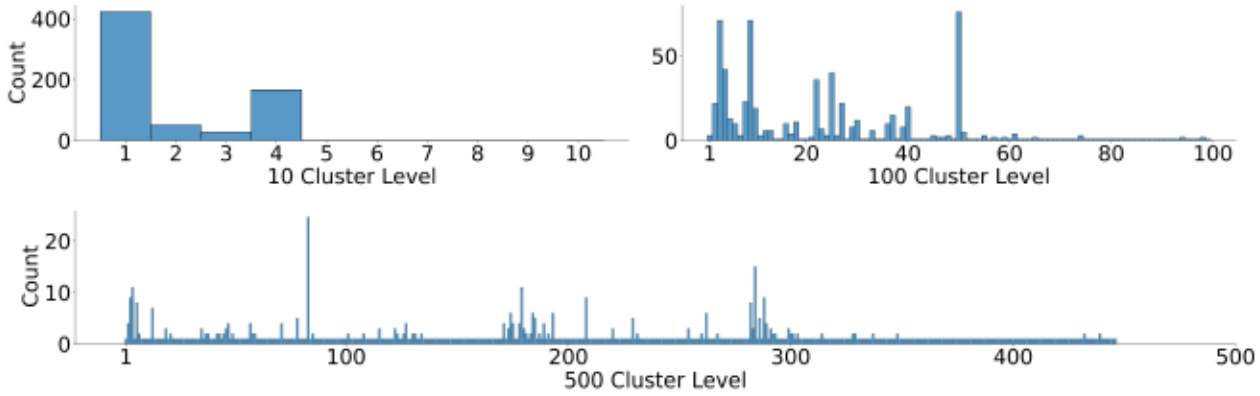


Figure A2: Count of assemblies of *M. tuberculosis* assigned to each cluster at the three clustering levels (assemblies clustered into 10, 100, and 500 clusters, labeled as 10 Cluster Level, 100 Cluster Level, and 500 Cluster Level, respectively). Cluster numbers are arbitrary and numbering increases left to right on the x-axis (some x-axis ticks were left out for visibility).

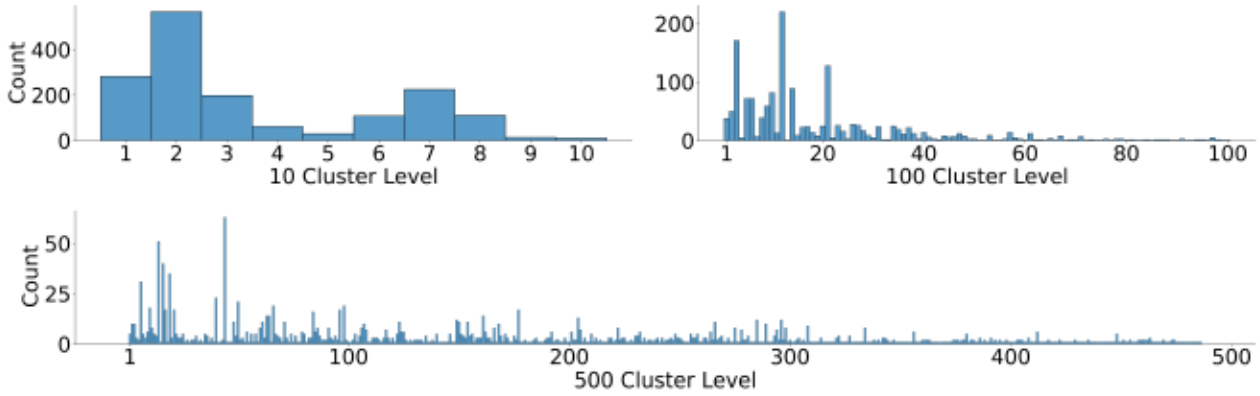


Figure A3: Count of assemblies of *L. monocytogenes* assigned to each cluster at the three clustering levels (assemblies clustered into 10, 100, and 500 clusters, labeled as 10 Cluster Level, 100 Cluster Level, and 500 Cluster Level, respectively). Cluster numbers are arbitrary and numbering increases left to right on the x-axis (some x-axis ticks were left out for visibility).

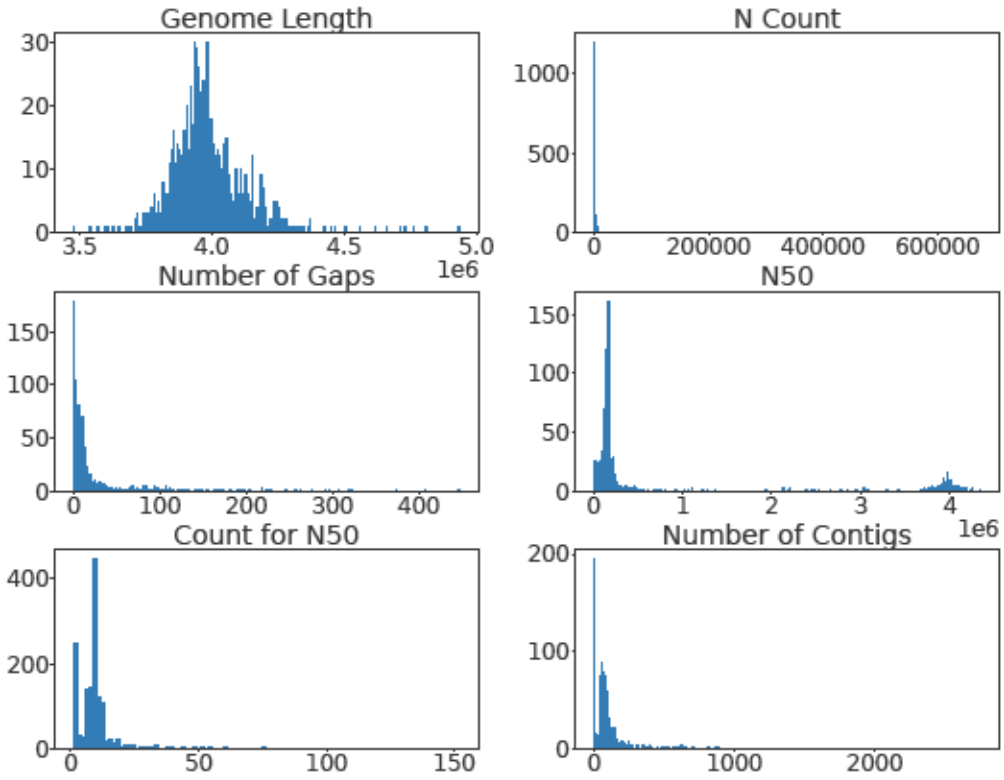


Figure A4: Assembly statistics for *A. baumannii*. Genome Length is the length of the genomes in nucleotides. N count is the number of ambiguous bases in the assemblies. Number of Gaps is the number of gaps (-) in the assemblies. N50 is the shortest contig that needs to be included to cover over 50% of the genome length and N50 count the number of contigs that need to be included to reach the N50. Number of Contigs is the number of contigs the assemblies consist of.

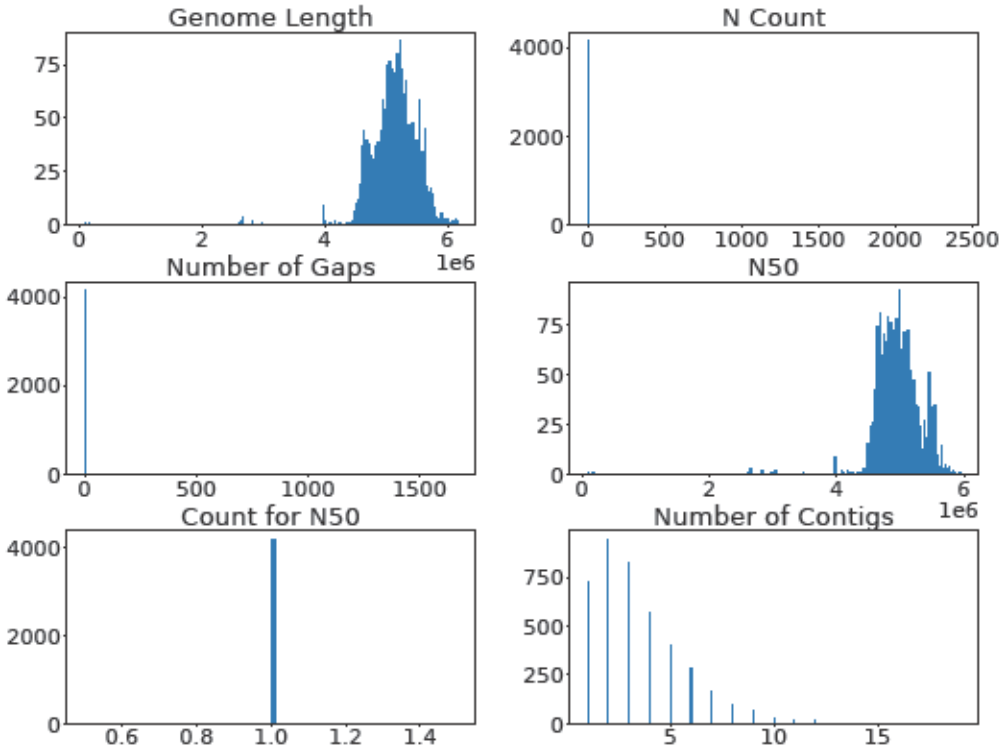


Figure A5: Assembly statistics for *E. coli*. Genome Length is the length of the genomes in nucleotides. N count is the number of ambiguous bases in the assemblies. Number of Gaps is the number of gaps (-) in the assemblies. N50 is the shortest contig that needs to be included to cover over 50% of the genome length and N50 count the number of contigs that need to be included to reach the N50. Number of Contigs is the number of contigs the assemblies consist of.

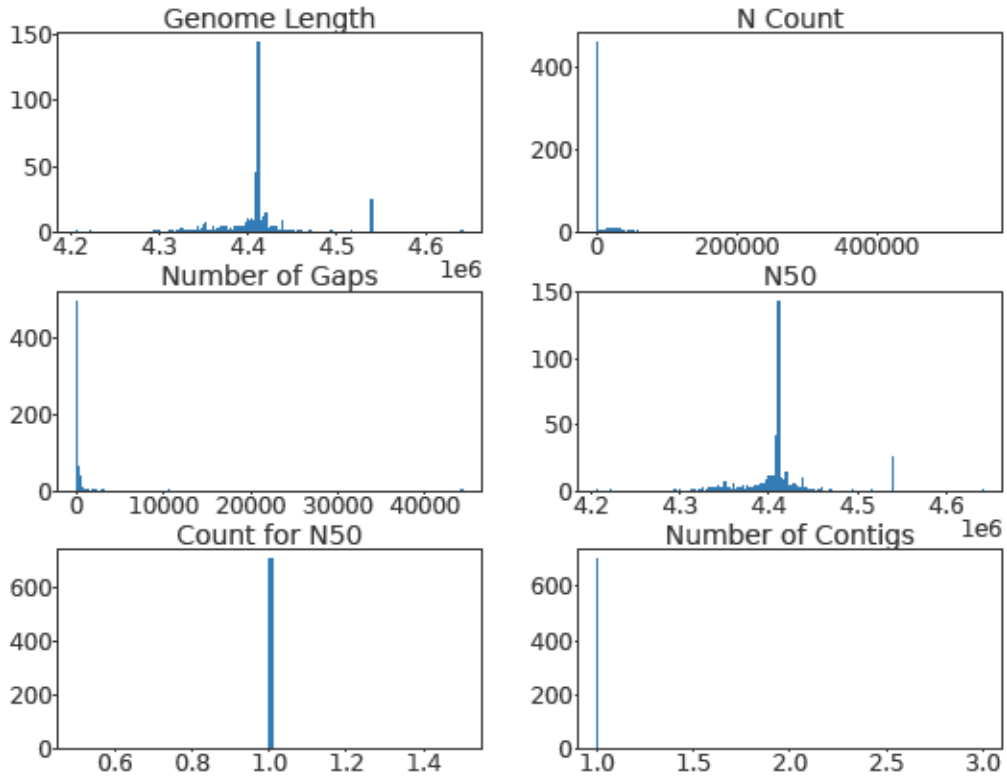


Figure A6: Assembly statistics for *M. tuberculosis*. Genome Length is the length of the genomes in nucleotides. N count is the number of ambiguous bases in the assemblies. Number of Gaps is the number of gaps (-) in the assemblies. N50 is the shortest contig that needs to be included to cover over 50% of the genome length and N50 count the number of contigs that need to be included to reach the N50. Number of Contigs is the number of contigs the assemblies consist of.

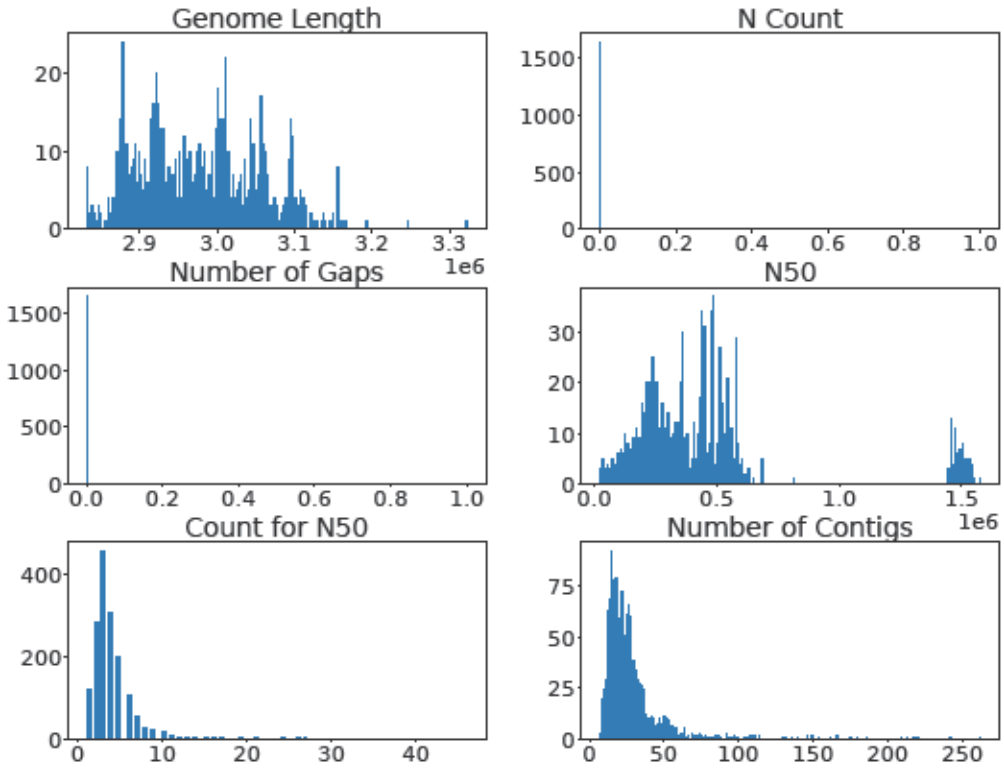


Figure A7: Assembly statistics for *L. monocytogenes*. Genome Length is the length of the genomes in nucleotides. N count is the number of ambiguous bases in the assemblies. Number of Gaps is the number of gaps (-) in the assemblies. N50 is the shortest contig that needs to be included to cover over 50% of the genome length and N50 count the number of contigs that need to be included to reach the N50. Number of Contigs is the number of contigs the assemblies consist of.

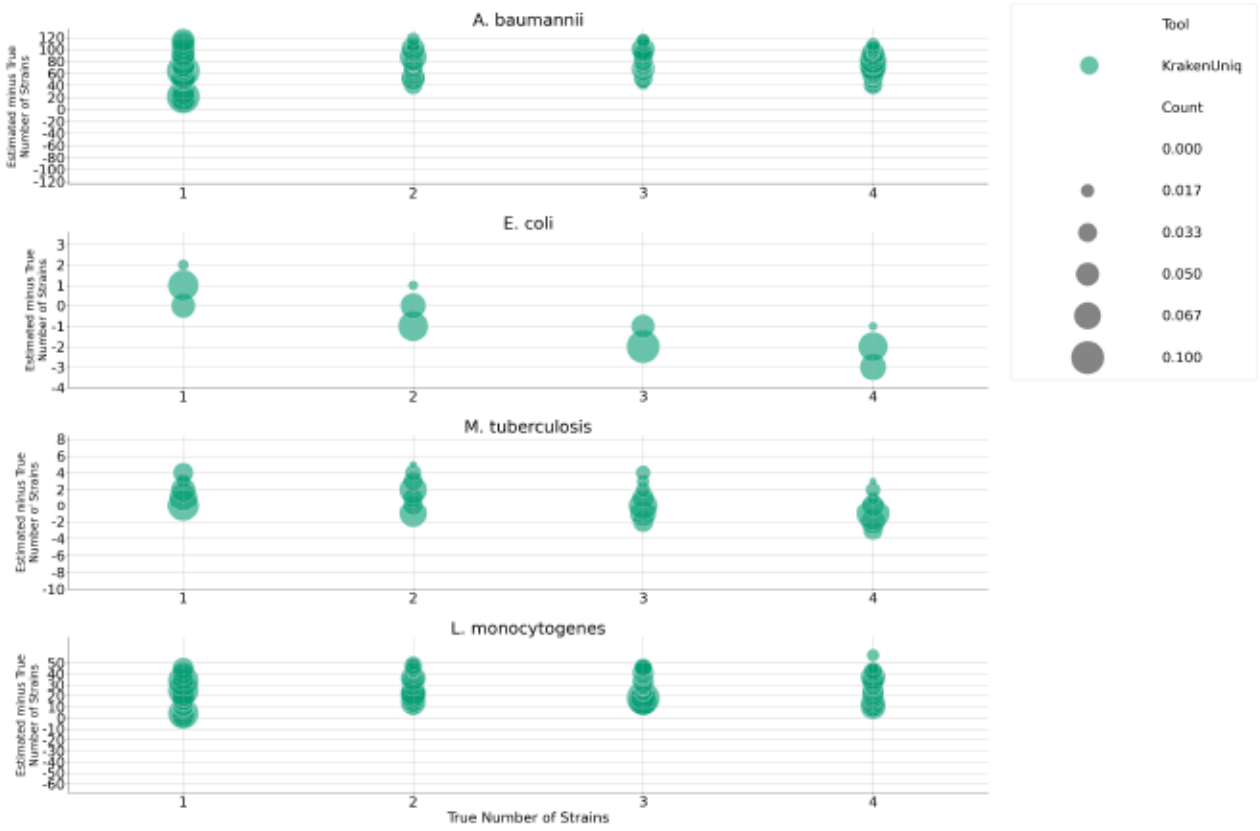


Figure A8: Inferred number of strains by KrakenUniq for samples made up of strains contained in the reference database. Size of circle represents fraction of samples.

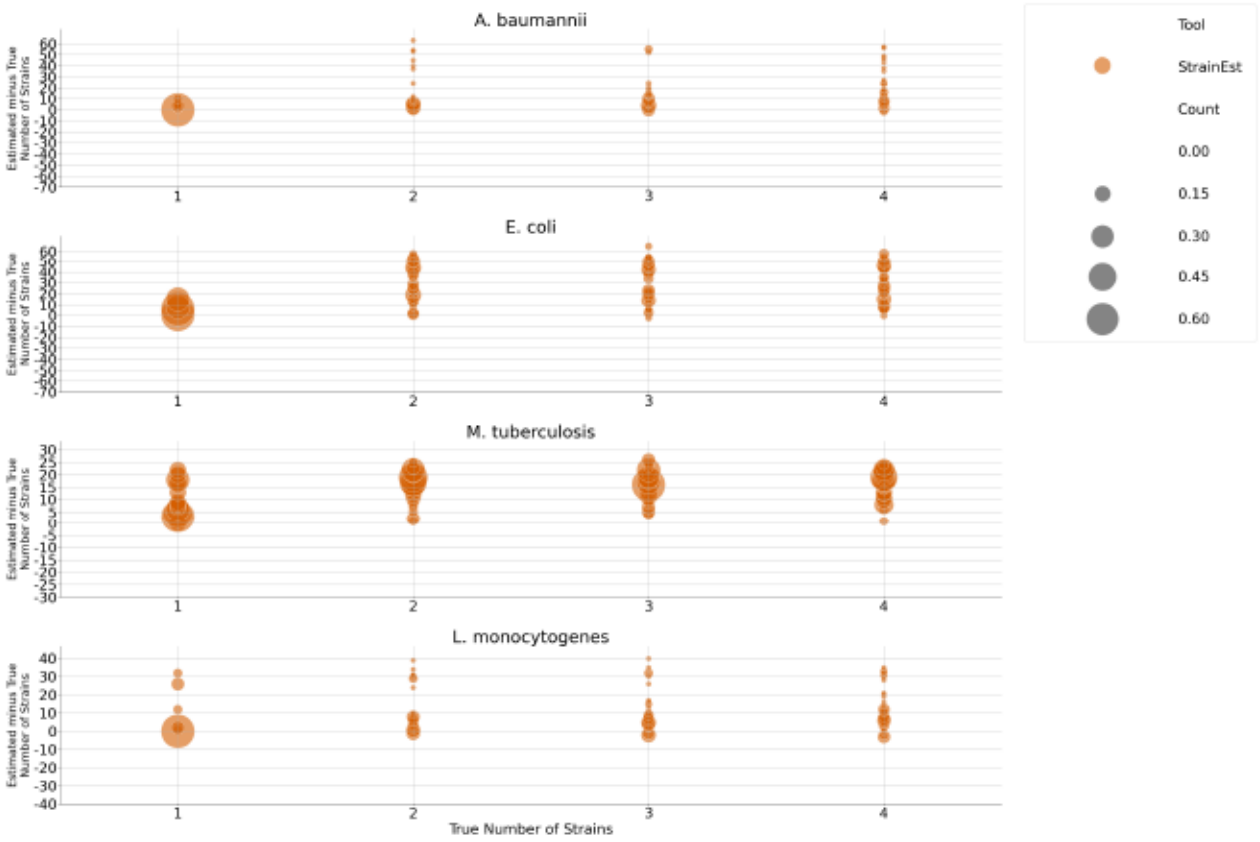


Figure A9: Inferred number of strains by StrainEst for samples made up of strains contained in the reference database. Size of circle represents fraction of samples.

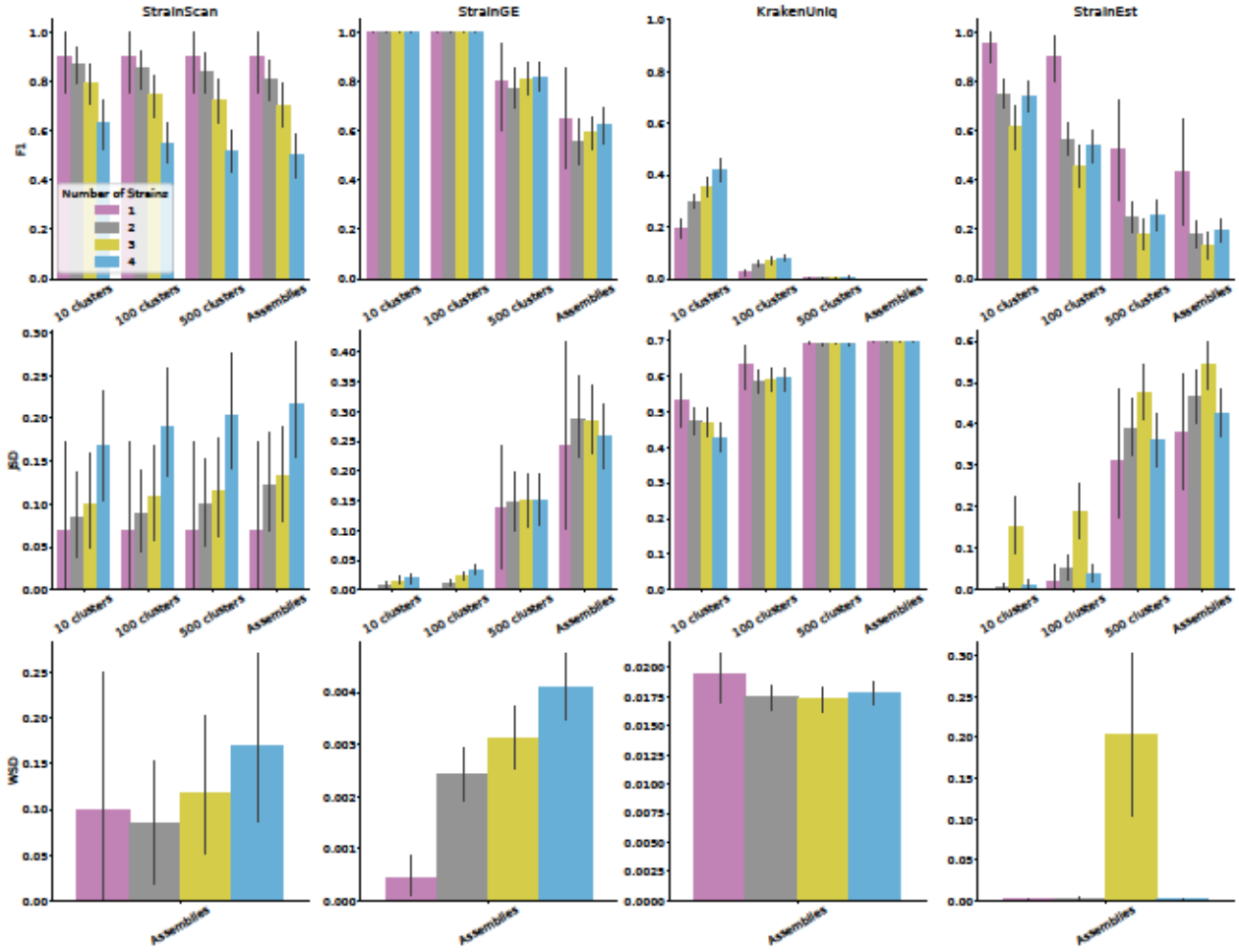


Figure A10: Results for samples made up of strains of *A. baumannii* contained in the reference database combined across all species for StrainScan, StrainGE, KrakenUniq, and StrainEst, stratified by number of strains in the samples. Samples analyzed include both samples with strains originating in the same and in different clusters at the 100 cluster level as well as samples with different coverages. Both samples that have equal relative abundance for all strains as well as the samples that have one dominant strain at 0.8 relative abundance with the remaining strains having equal shares of the remaining 0.2 relative abundance were analyzed in combination. Bar height shows mean performance and error bars the 95% confidence interval.

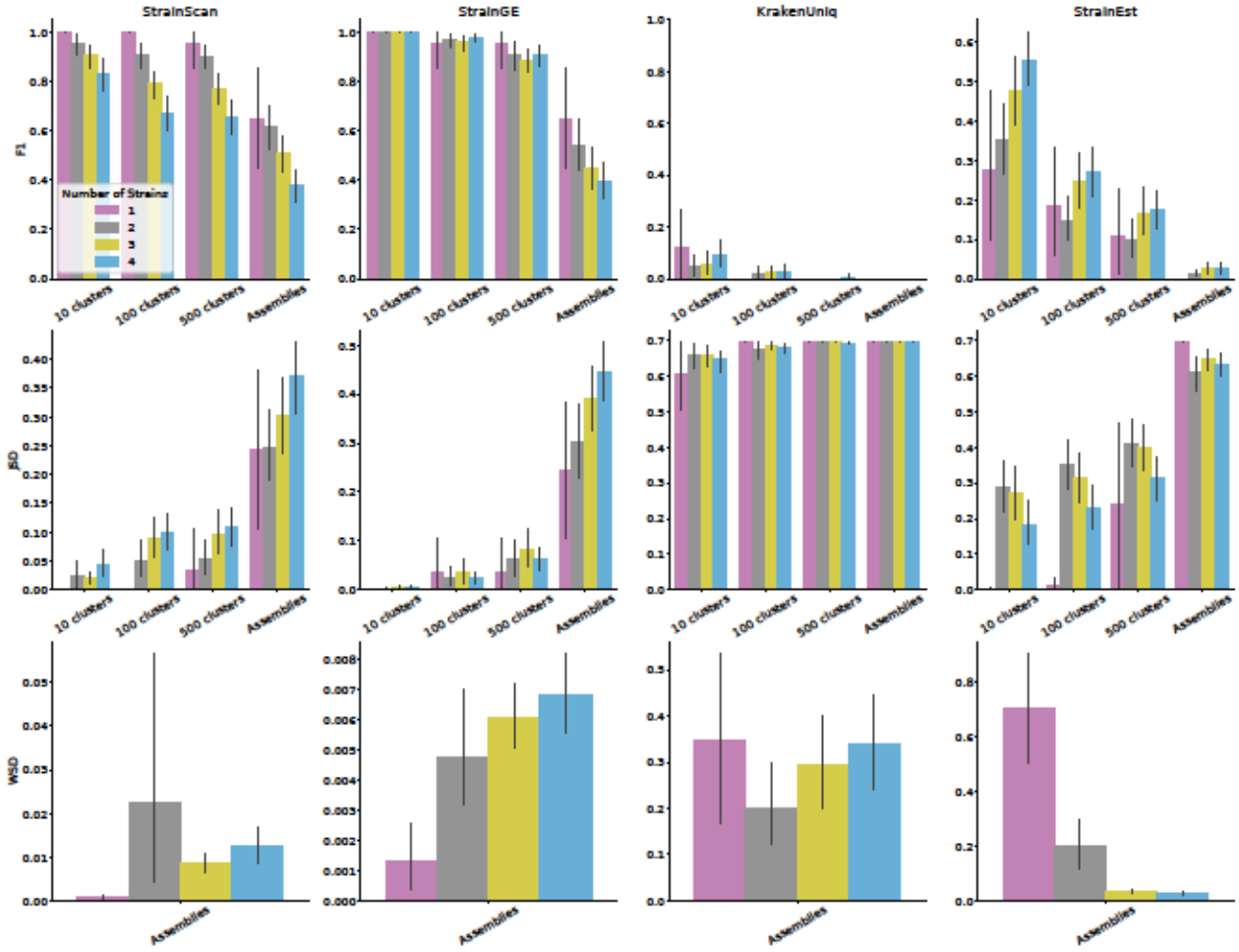


Figure A11: Results for samples made up of strains of *E. coli* contained in the reference database combined across all species for StrainScan, StrainGE, KrakenUniq, and StrainEst, stratified by number of strains in the samples. Samples analyzed include both samples with strains originating in the same and in different clusters at the 100 cluster level as well as samples with different coverages. Both samples that have equal relative abundance for all strains as well as the samples that have one dominant strain at 0.8 relative abundance with the remaining strains having equal shares of the remaining 0.2 relative abundance were analyzed in combination. Bar height shows mean performance and error bars the 95% confidence interval.

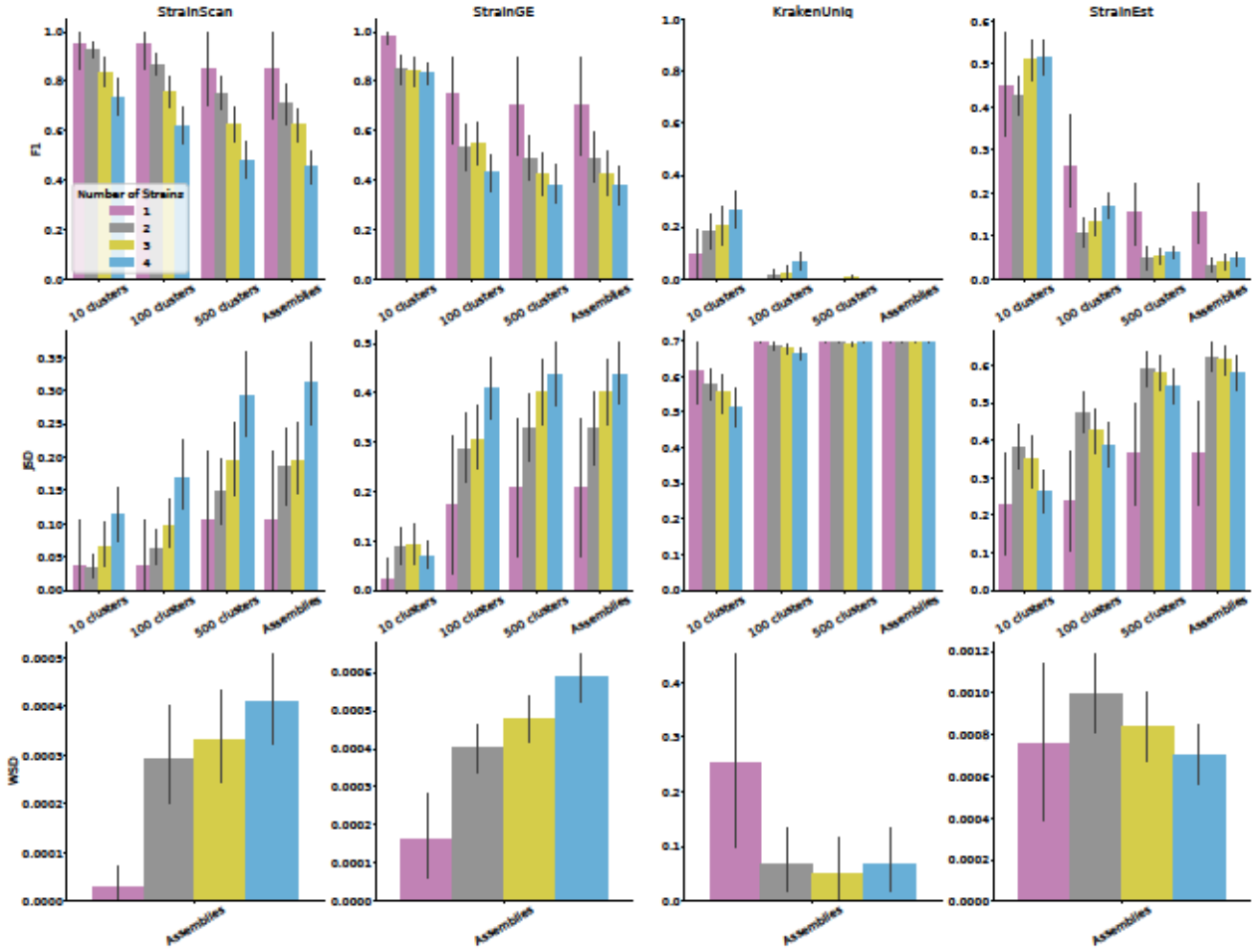


Figure A12: Results for samples made up of strains of *M. tuberculosis* contained in the reference database combined across all species for StrainScan, StrainGE, KrakenUniq, and StrainEst, stratified by number of strains in the samples. Samples analyzed include both samples with strains originating in the same and in different clusters at the 100 cluster level as well as samples with different coverages. Both samples that have equal relative abundance for all strains as well as the samples that have one dominant strain at 0.8 relative abundance with the remaining strains having equal shares of the remaining 0.2 relative abundance were analyzed in combination. Bar height shows mean performance and error bars the 95% confidence interval.

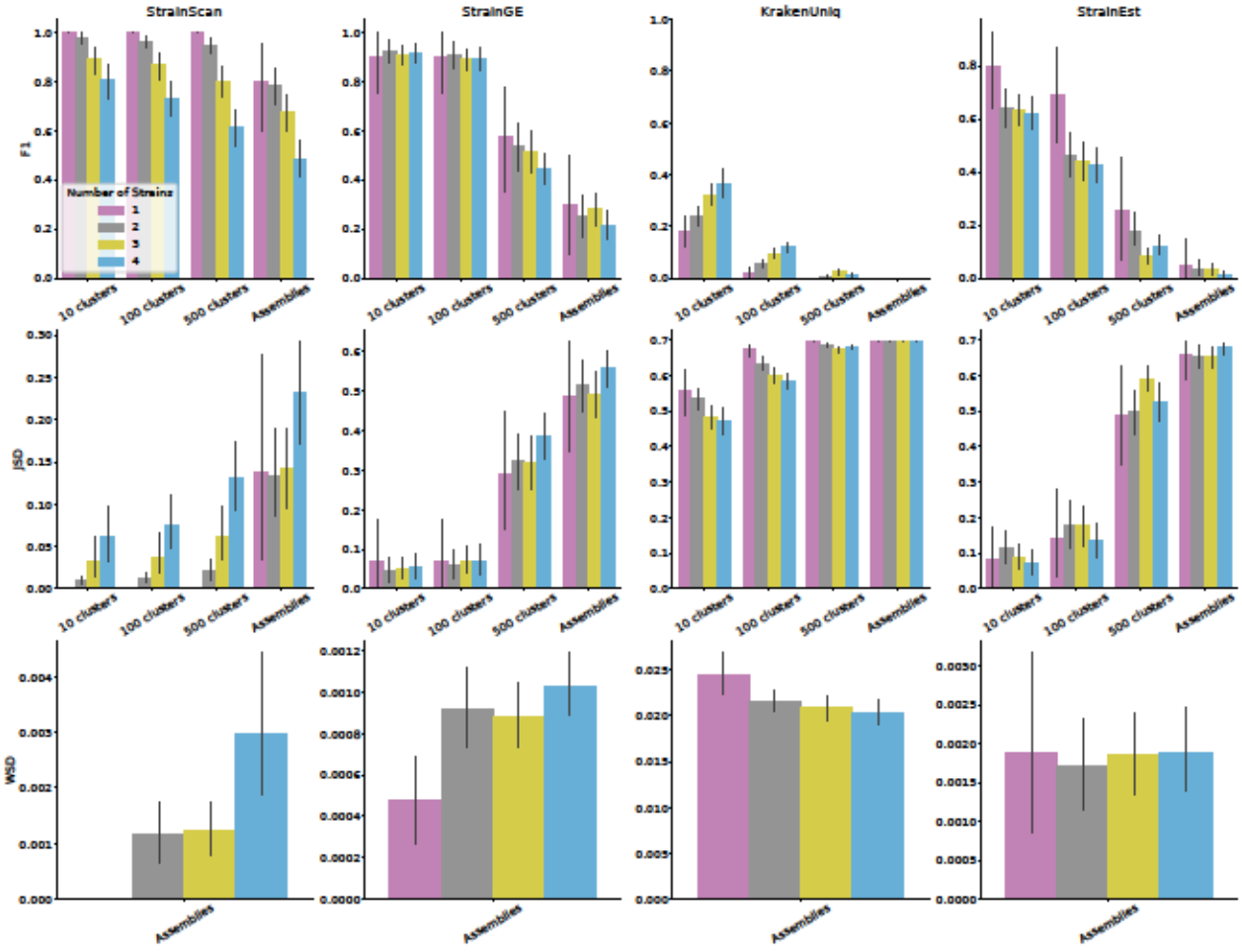


Figure A13: Results for samples made up of strains of *L. monocytogenes* contained in the reference database combined across all species for StrainScan, StrainGE, KrakenUniq, and StrainEst, stratified by number of strains in the samples. Samples analyzed include both samples with strains originating in the same and in different clusters at the 100 cluster level as well as samples with different coverages. Both samples that have equal relative abundance for all strains as well as the samples that have one dominant strain at 0.8 relative abundance with the remaining strains having equal shares of the remaining 0.2 relative abundance were analyzed in combination. Bar height shows mean performance and error bars the 95% confidence interval.

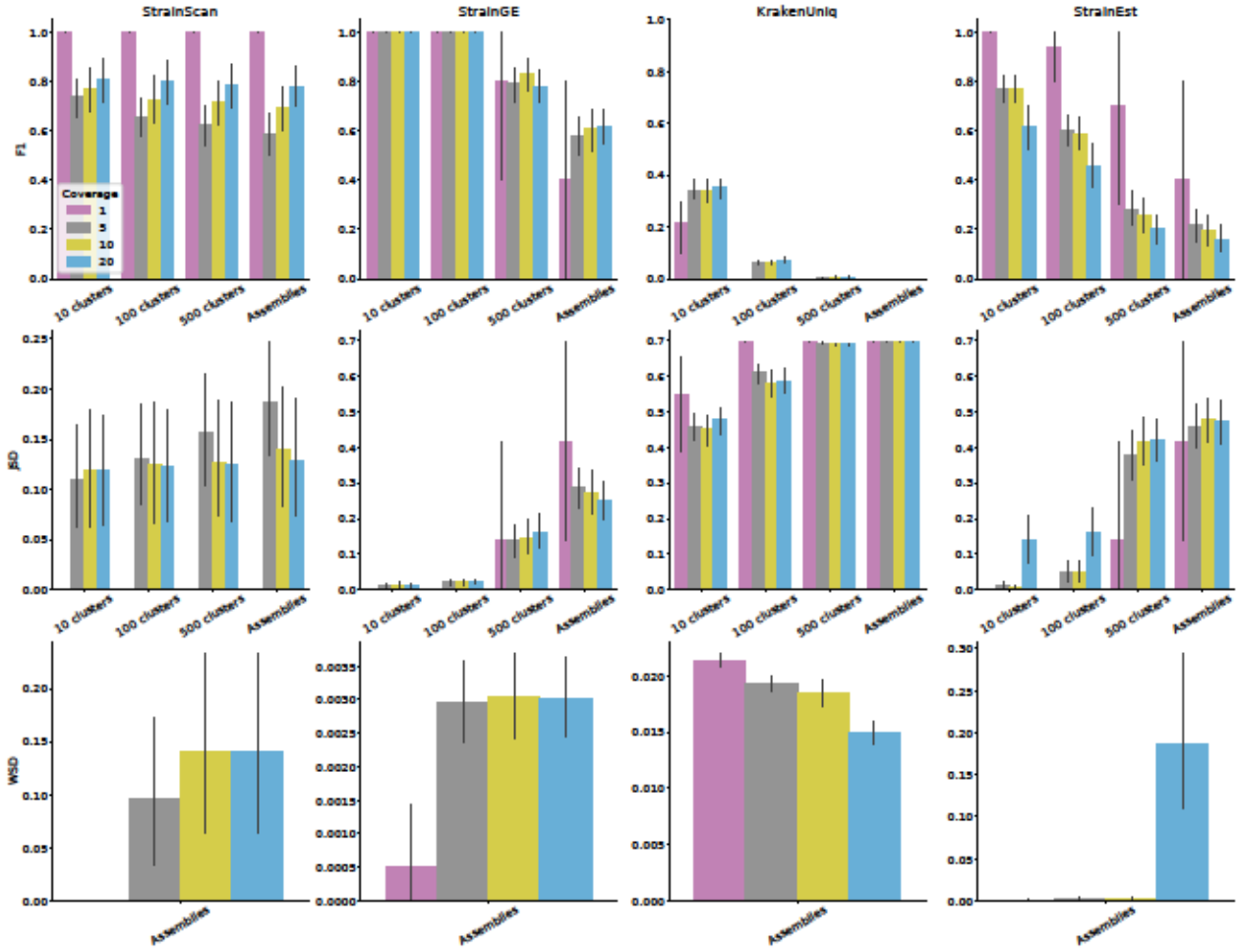


Figure A14: Results for samples made up of strains of *A. baumannii* contained in the reference database combined across all species for StrainScan, StrainGE, KrakenUniq, and StrainEst, stratified by the total coverage of the samples. Samples analyzed include both samples with strains originating in the same and in different clusters at the 100 cluster level as well as samples with different numbers of constituent strains. Both samples that have equal relative abundance for all strains as well as the samples that have one dominant strain at 0.8 relative abundance with the remaining strains having equal shares of the remaining 0.2 relative abundance were analyzed in combination. Bar height shows mean performance and error bars the 95% confidence interval.

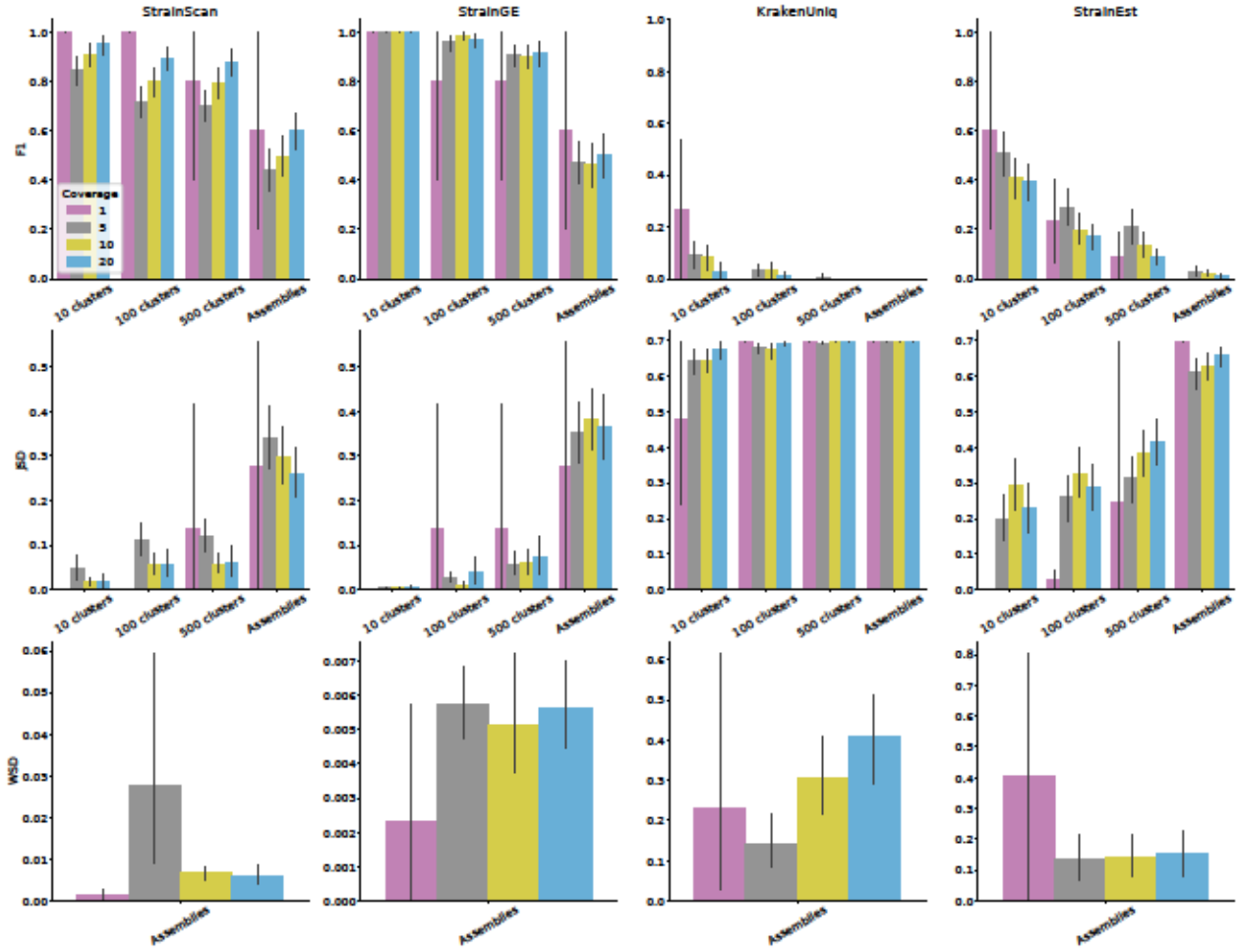


Figure A15: Results for samples made up of strains of *E. coli* contained in the reference database combined across all species for StrainScan, StrainGE, KrakenUniq, and StrainEst, stratified by the total coverage of the samples. Samples analyzed include both samples with strains originating in the same and in different clusters at the 100 cluster level as well as samples with different numbers of constituent strains. Both samples that have equal relative abundance for all strains as well as the samples that have one dominant strain at 0.8 relative abundance with the remaining strains having equal shares of the remaining 0.2 relative abundance were analyzed in combination. Bar height shows mean performance and error bars the 95% confidence interval.

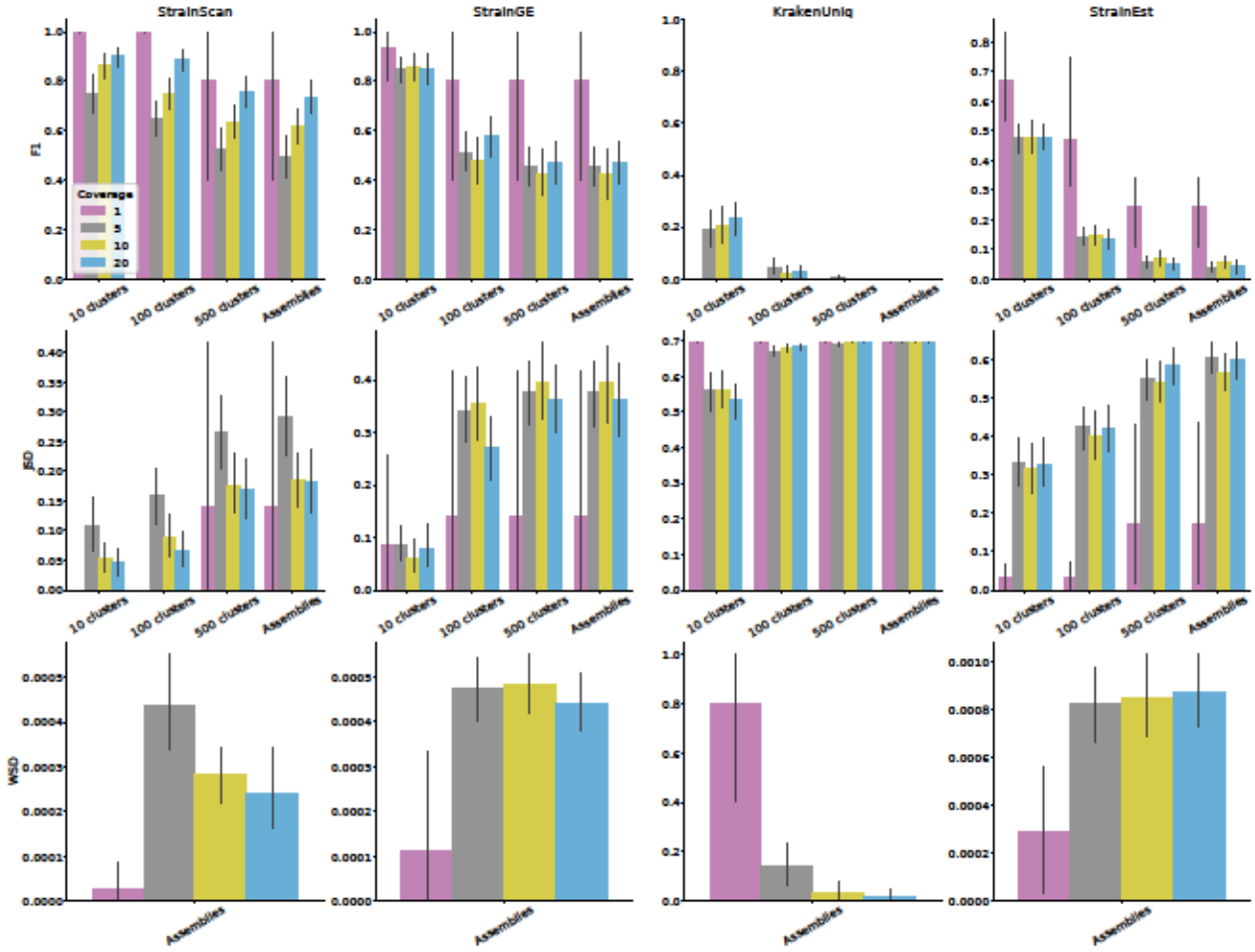


Figure A16: Results for samples made up of strains of *M. tuberculosis* contained in the reference database combined across all species for StrainScan, StrainGE, KrakenUniq, and StrainEst, stratified by the total coverage of the samples. Samples analyzed include both samples with strains originating in the same and in different clusters at the 100 cluster level as well as samples with different numbers of constituent strains. Both samples that have equal relative abundance for all strains as well as the samples that have one dominant strain at 0.8 relative abundance with the remaining strains having equal shares of the remaining 0.2 relative abundance were analyzed in combination. Bar height shows mean performance and error bars the 95% confidence interval.

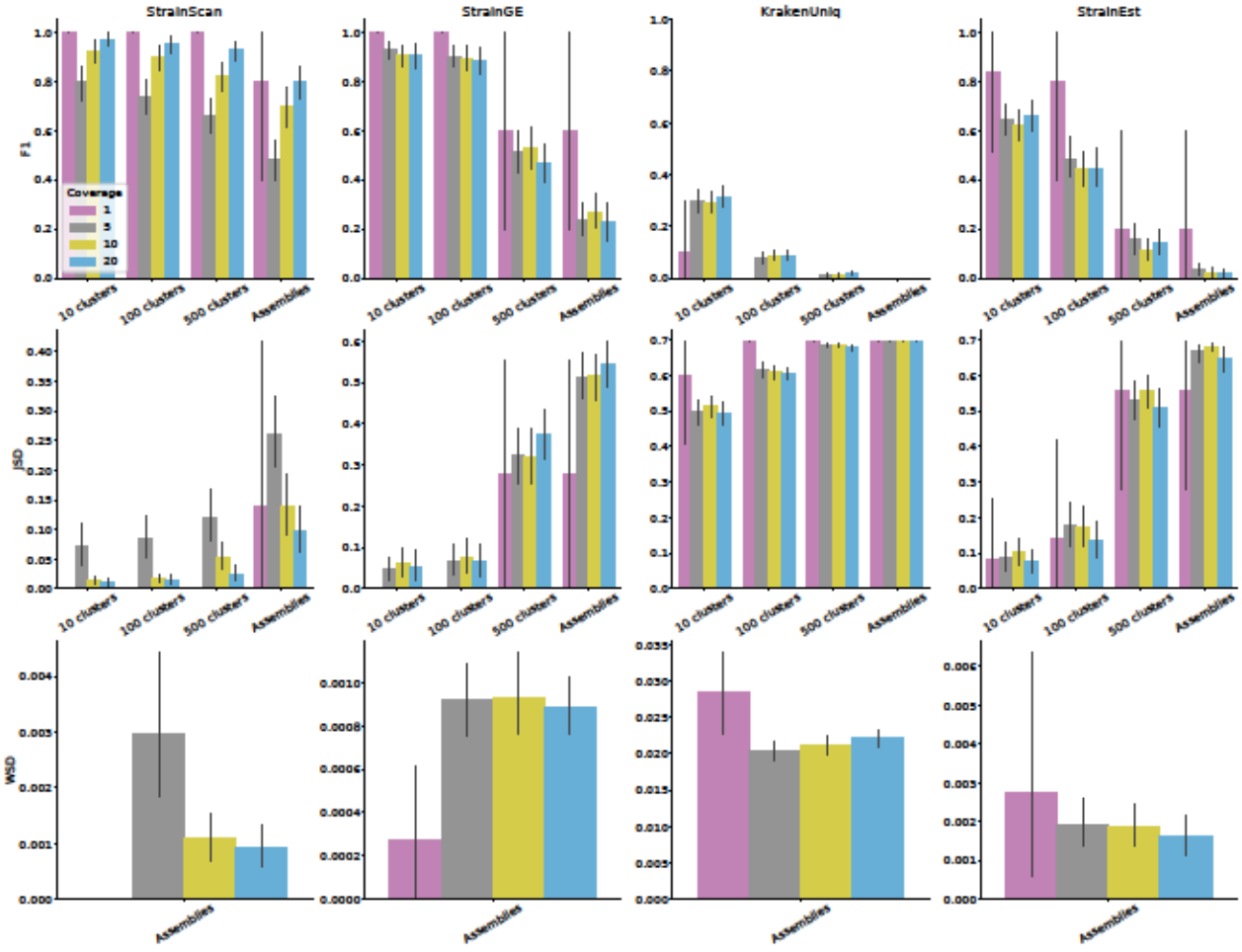


Figure A17: Results for samples made up of strains of *L. monocytogenes* contained in the reference database combined across all species for StrainScan, StrainGE, KrakenUniq, and StrainEst, stratified by the total coverage of the samples. Samples analyzed include both samples with strains originating in the same and in different clusters at the 100 cluster level as well as samples with different numbers of constituent strains. Both samples that have equal relative abundance for all strains as well as the samples that have one dominant strain at 0.8 relative abundance with the remaining strains having equal shares of the remaining 0.2 relative abundance were analyzed in combination. Bar height shows mean performance and error bars the 95% confidence interval.

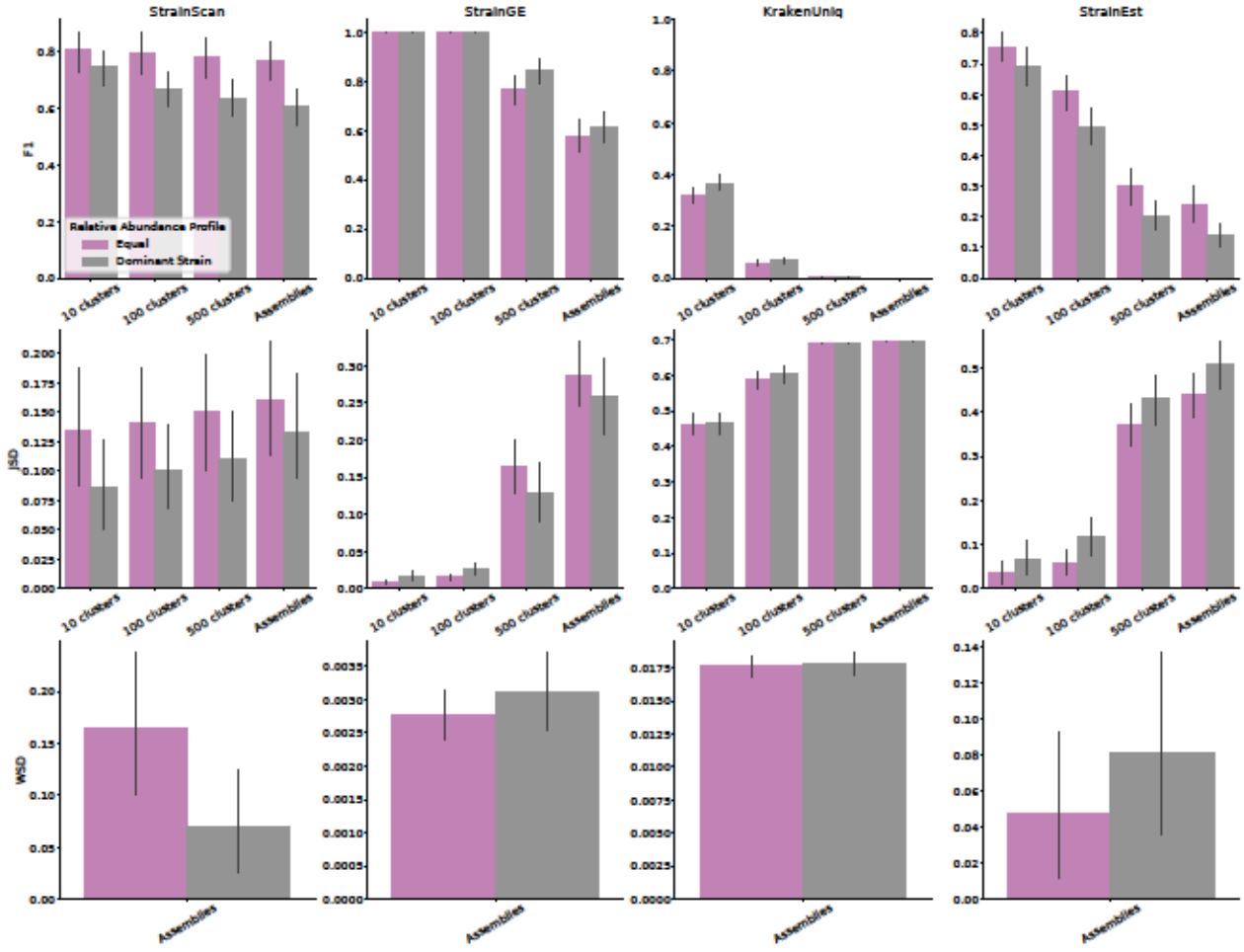


Figure A18: Results for samples made up of strains of *A. baumannii* contained in the reference database combined across all species for StrainScan, StrainGE, KrakenUniq, and StrainEst, stratified by the relative abundance profile. In the equal profile all strains have equal relative abundance. In the dominant strain samples one strain's relative abundance is 80%, with all other strains being equally distributed across the remaining 20%. Samples analyzed together include both samples with strains originating in the same and in different clusters at the 100 cluster level as well as samples with different numbers of constituent strains and coverages. Bar height shows mean performance and error bars the 95% confidence interval.

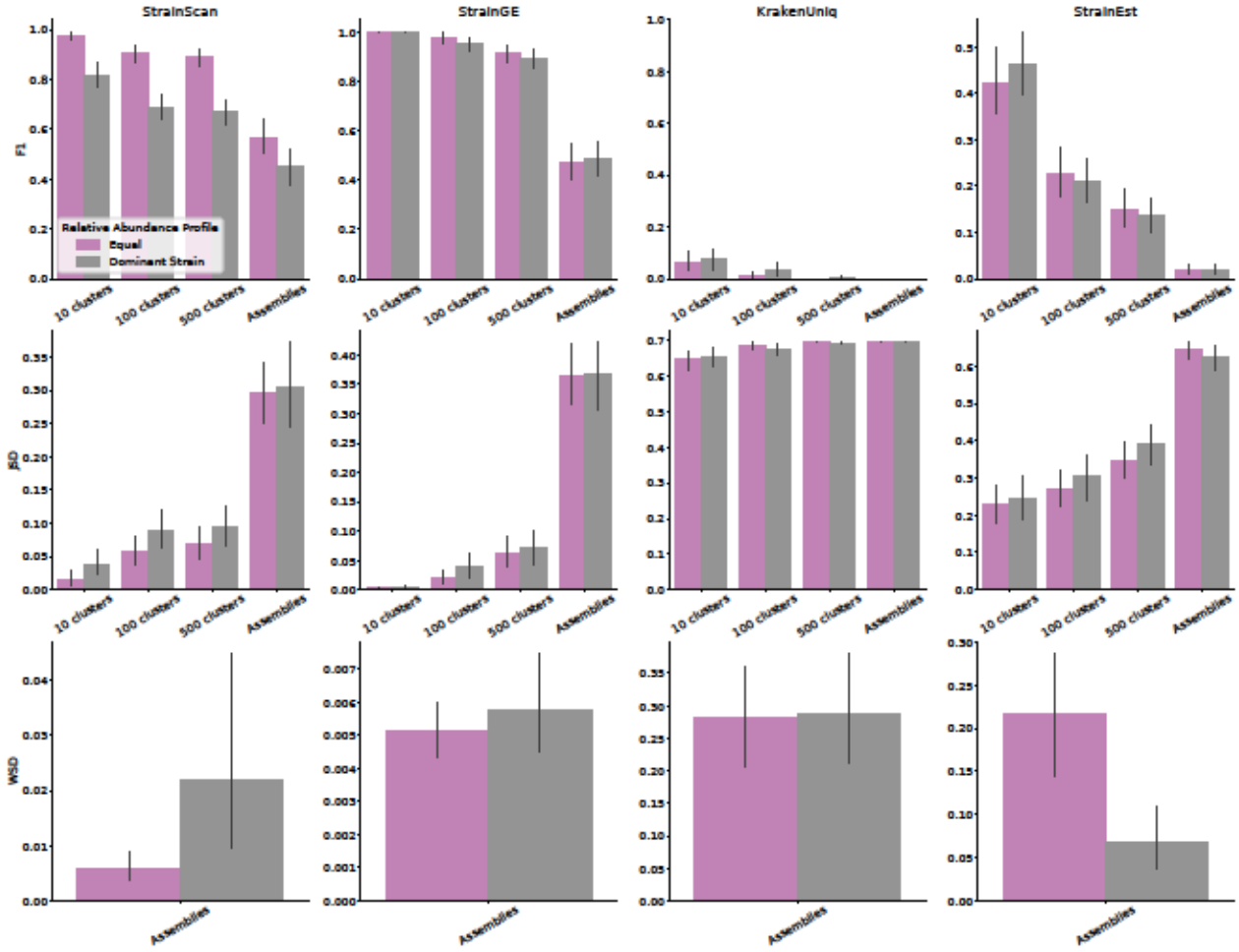


Figure A19: Results for samples made up of strains of *E. coli* contained in the reference database combined across all species for StrainScan, StrainGE, KrakenUniq, and StrainEst, stratified by the relative abundance profile. In the equal profile all strains have equal relative abundance. In the dominant strain samples one strain's relative abundance is 80%, with all other strains being equally distributed across the remaining 20%. Samples analyzed together include both samples with strains originating in the same and in different clusters at the 100 cluster level as well as samples with different numbers of constituent strains and coverages. Bar height shows mean performance and error bars the 95% confidence interval.

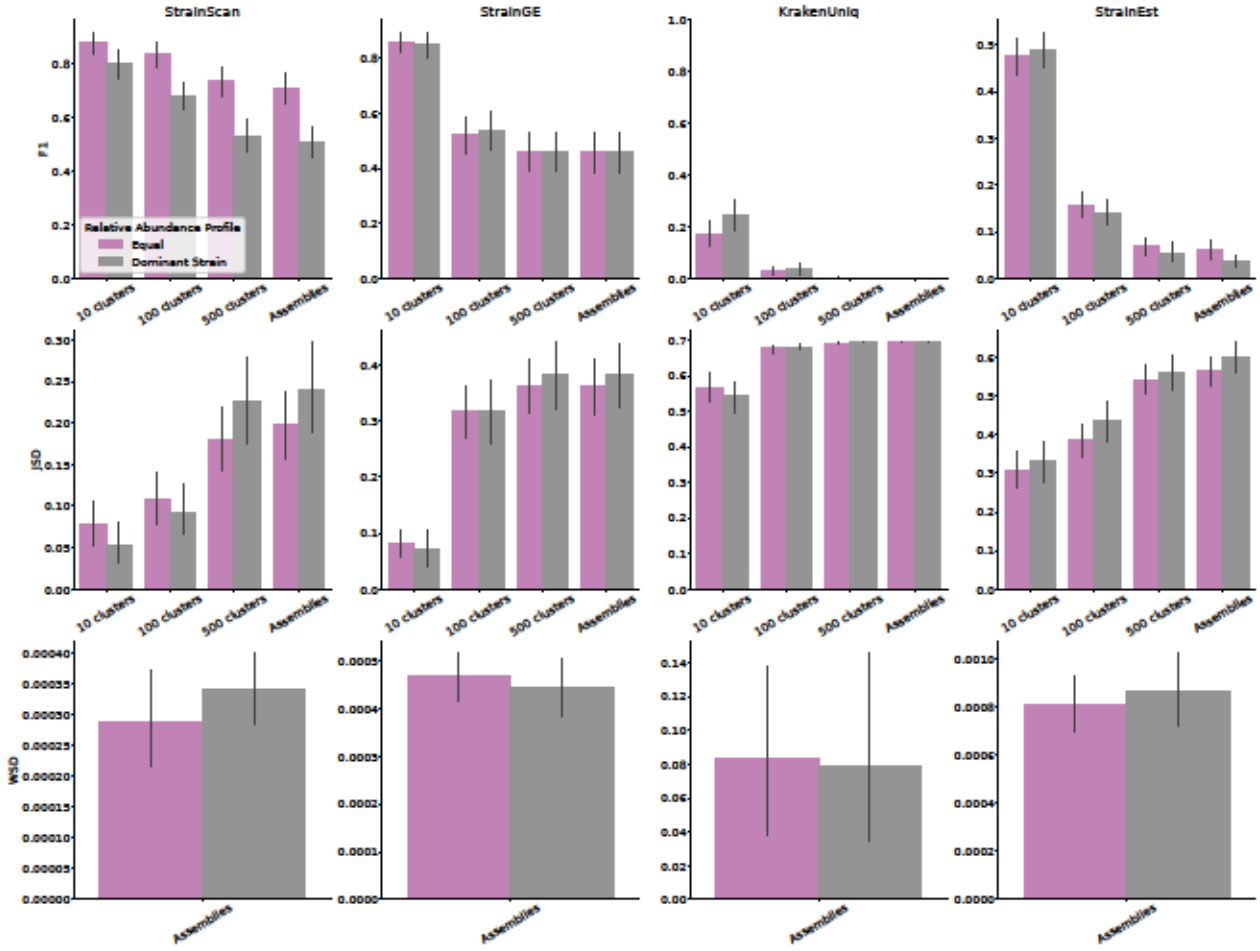


Figure A20: Results for samples made up of strains of *M. tuberculosis* contained in the reference database combined across all species for StrainScan, StrainGE, KrakenUniq, and StrainEst, stratified by the relative abundance profile. In the equal profile all strains have equal relative abundance. In the dominant strain samples one strain's relative abundance is 80%, with all other strains being equally distributed across the remaining 20%. Samples analyzed together include both samples with strains originating in the same and in different clusters at the 100 cluster level as well as samples with different numbers of constituent strains and coverages. Bar height shows mean performance and error bars the 95% confidence interval.

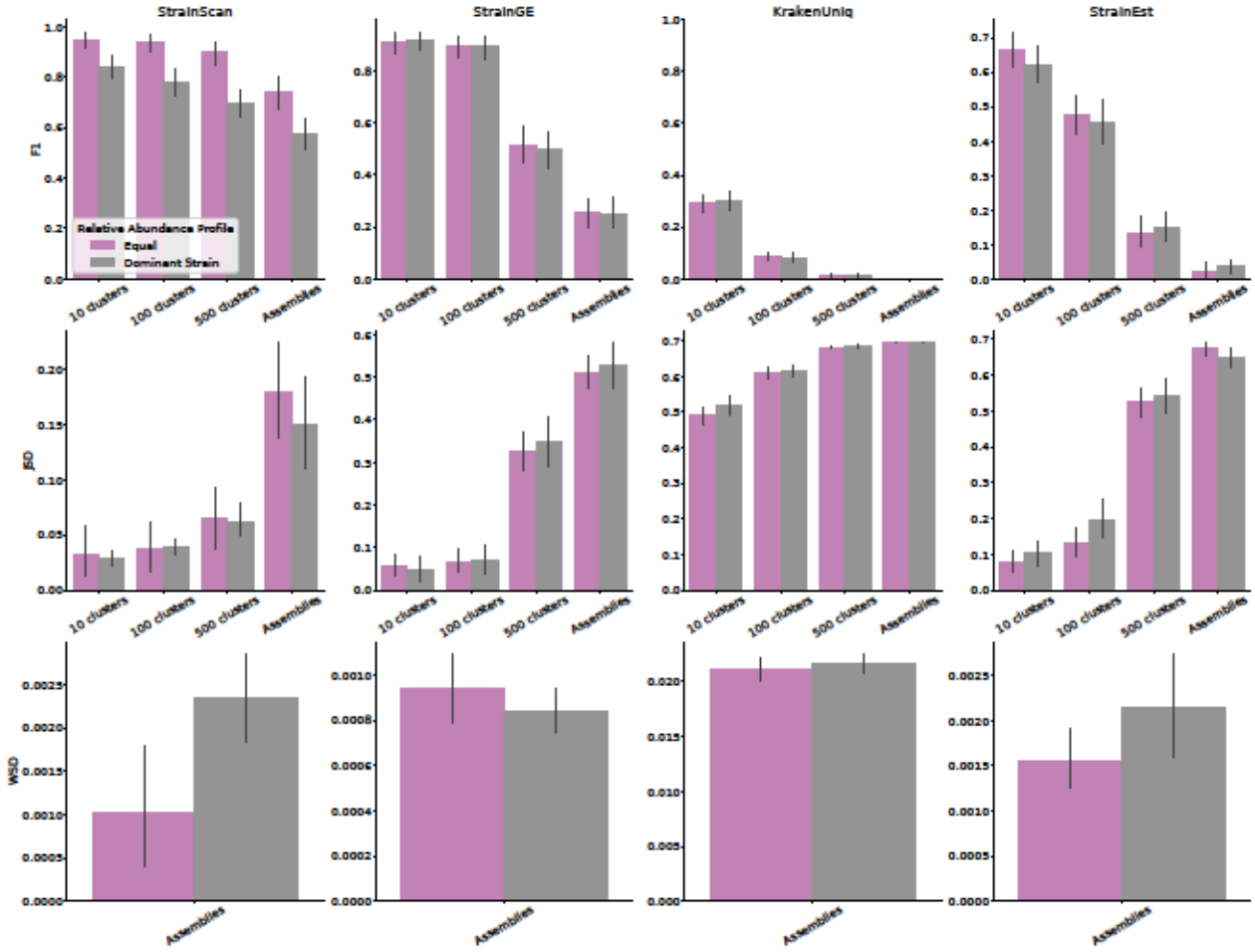


Figure A21: Results for samples made up of strains of *L. monocytogenes* contained in the reference database combined across all species for StrainScan, StrainGE, KrakenUniq, and StrainEst, stratified by the relative abundance profile. In the equal profile all strains have equal relative abundance. In the dominant strain samples one strain's relative abundance is 80%, with all other strains being equally distributed across the remaining 20%. Samples analyzed together include both samples with strains originating in the same and in different clusters at the 100 cluster level as well as samples with different numbers of constituent strains and coverages. Bar height shows mean performance and error bars the 95% confidence interval.

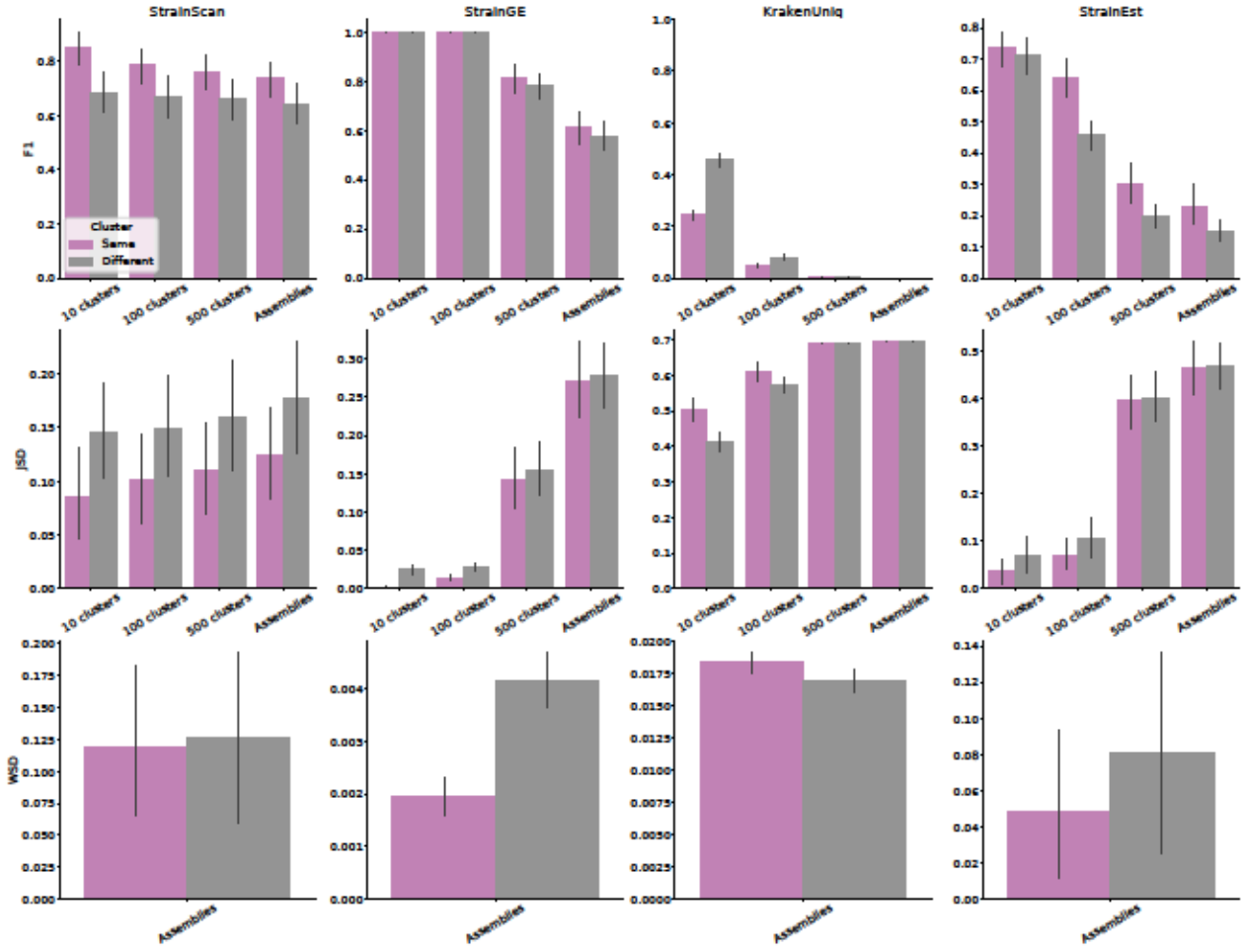


Figure A22: Results for samples made up of strains of *A. baumannii* contained in the reference database combined across all species for StrainScan, StrainGE, KrakenUniq, and StrainEst, stratified by the cluster of origin. Same cluster indicates that the strains in the sample are in the same cluster at the 100 cluster level, while different indicate they originate from different clusters. Samples analyzed together include both samples with different numbers of constituent strains and coverages. Both samples that have equal relative abundance for all strains as well as the samples that have one dominant strain at 0.8 relative abundance with the remaining strains having equal shares of the remaining 0.2 relative abundance were analyzed in combination. Bar height shows mean performance and error bars the 95% confidence interval.

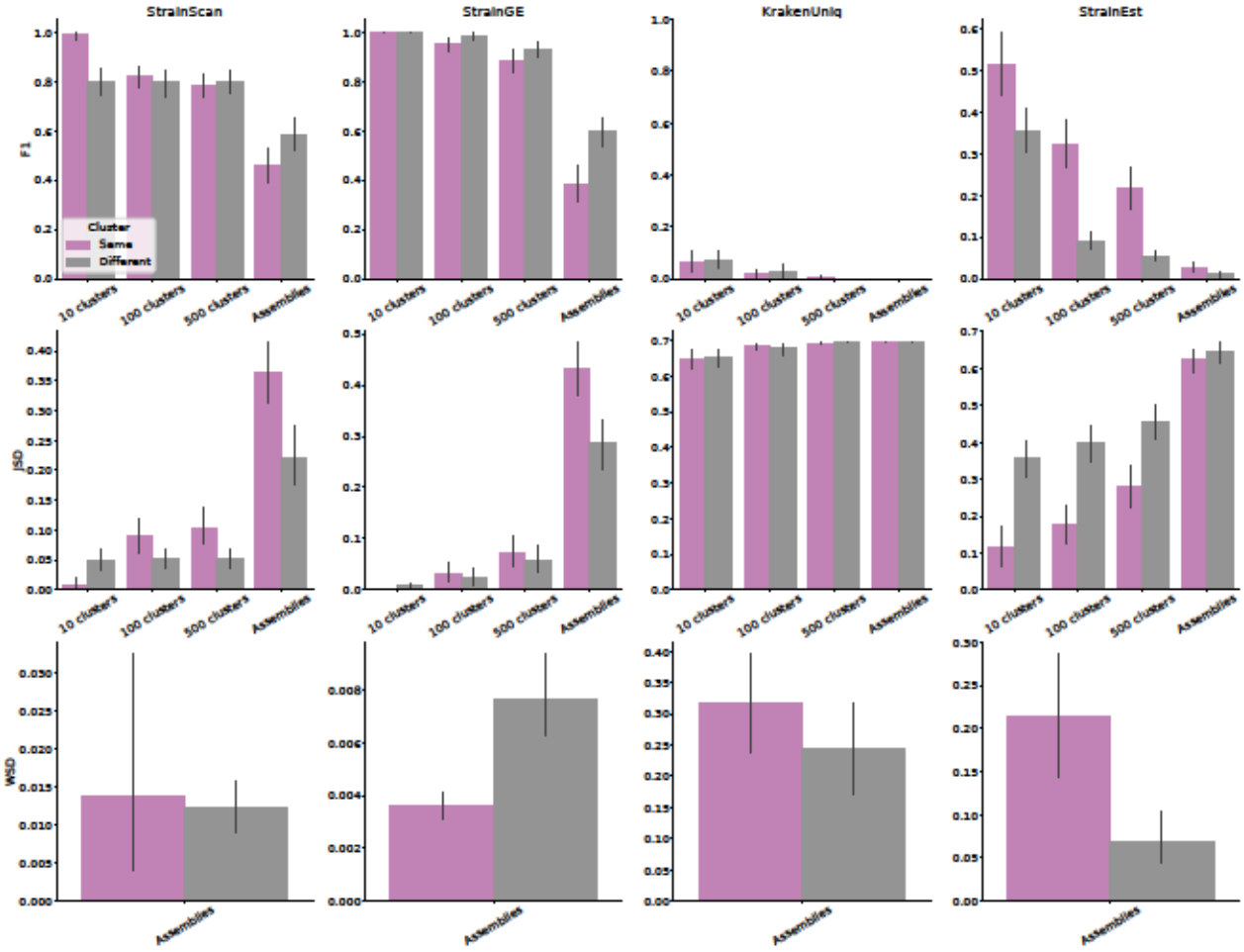


Figure A23: Results for samples made up of strains of *E. coli* contained in the reference database combined across all species for StrainScan, StrainGE, KrakenUniq, and StrainEst, stratified by the cluster of origin. Same cluster indicates that the strains in the sample are in the same cluster at the 100 cluster level, while different indicate they originate from different clusters. Samples analyzed together include both samples with different numbers of constituent strains and coverages. Both samples that have equal relative abundance for all strains as well as the samples that have one dominant strain at 0.8 relative abundance with the remaining strains having equal shares of the remaining 0.2 relative abundance were analyzed in combination. Bar height shows mean performance and error bars the 95% confidence interval.

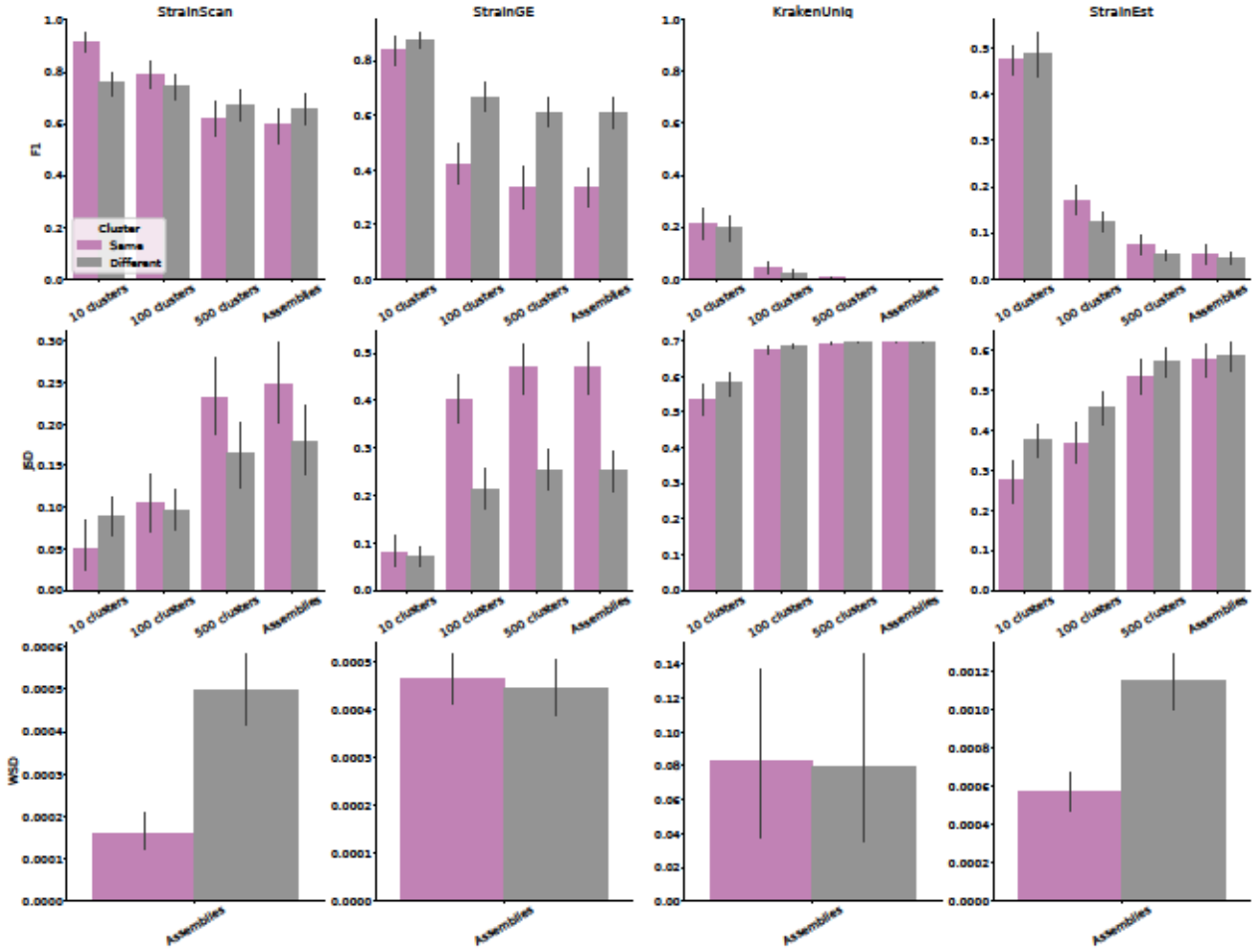


Figure A24: Results for samples made up of strains of *M. tuberculosis* contained in the reference database combined across all species for StrainScan, StrainGE, KrakenUniq, and StrainEst, stratified by the cluster of origin. Same cluster indicates that the strains in the sample are in the same cluster at the 100 cluster level, while different indicate they originate from different clusters. Samples analyzed together include both samples with different numbers of constituent strains and coverages. Both samples that have equal relative abundance for all strains as well as the samples that have one dominant strain at 0.8 relative abundance with the remaining strains having equal shares of the remaining 0.2 relative abundance were analyzed in combination. Bar height shows mean performance and error bars the 95% confidence interval.

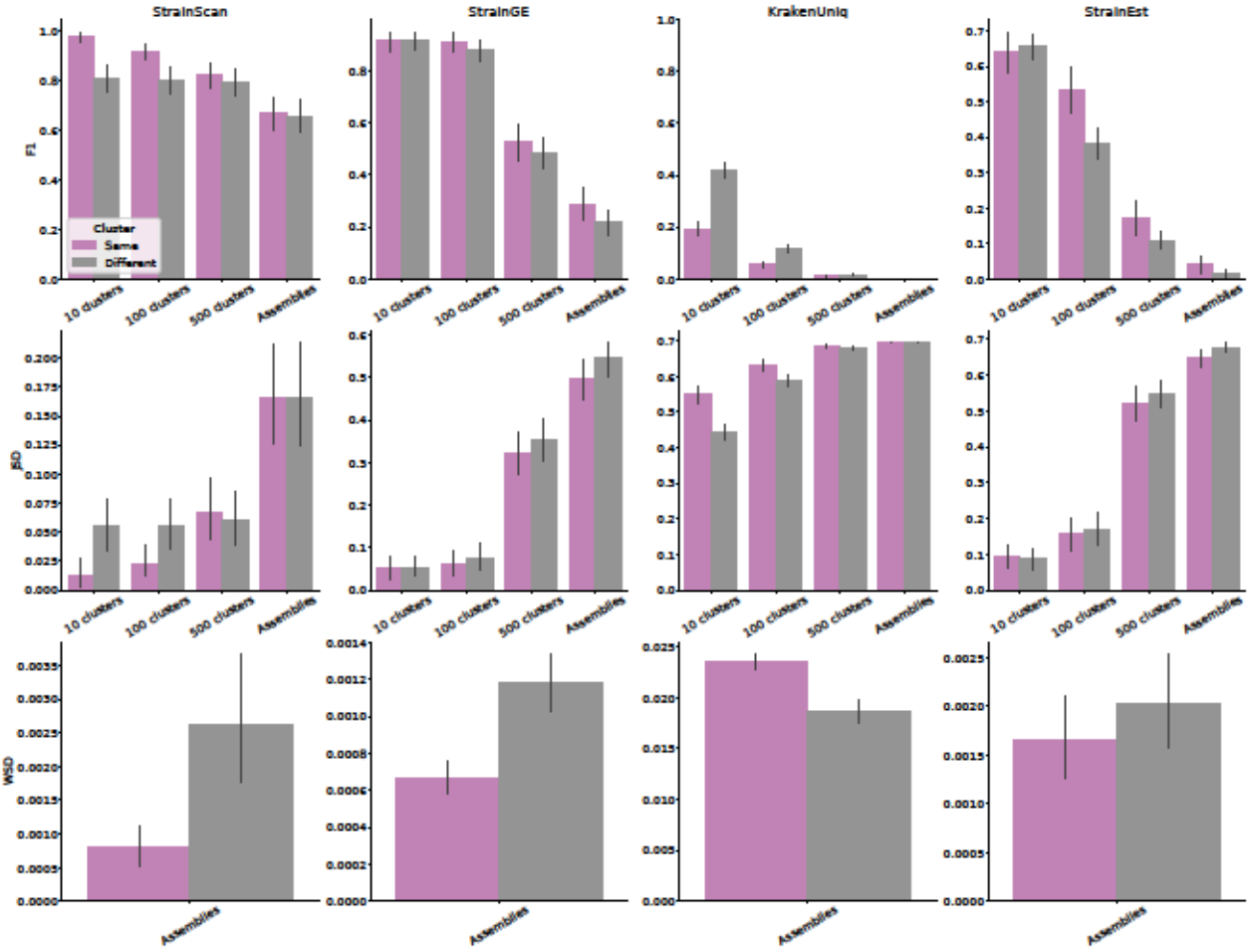


Figure A25: Results for samples made up of strains of *L. monocytogenes* contained in the reference database combined across all species for StrainScan, StrainGE, KrakenUniq, and StrainEst, stratified by the cluster of origin. Same cluster indicates that the strains in the sample are in the same cluster at the 100 cluster level, while different indicate they originate from different clusters. Samples analyzed together include both samples with different numbers of constituent strains and coverages. Both samples that have equal relative abundance for all strains as well as the samples that have one dominant strain at 0.8 relative abundance with the remaining strains having equal shares of the remaining 0.2 relative abundance were analyzed in combination. Bar height shows mean performance and error bars the 95% confidence interval.

Appendix B

Supplementary Text and Figures for Chapter IV

Modifications of the Accessory-Proportion Model

There exist many possibilities for expanding and changing the accessory-proportion model, two of which I will briefly address here.

An issue with the current formulation is the strict separation of core and accessory genes. It is clear that the number of inferred core genes will drop with the number of included reference strains, potentially too low for some species to generate high-quality estimates. Genes present in almost all strains are considered as soft core genes, genes that are distributed across most members of a species[247]. I implemented a version of the model in which some genes are considered both as accessory and as core genes with their importance in either part of the objective function being dependent on the percentage of reference strains they appear in. Specifically, $\omega_g = 1$ is set for genes g that are present in 100% of reference strains and $\omega_g = 0$ for genes that are present in 95% or fewer of the reference strains. Then we can linearly interpolate between these two points, setting

$$\omega_g = \max\{1 - 20(1 - r_g), 0\} \tag{2.1}$$

for each gene, where r_g is the percentage of reference strains carrying gene g. In an abuse of notation, let $\omega_l = \omega_g$ for positions l corresponding to gene g. Adding this weight to the objective function results in

$$\lambda_s \sum_{s \in \mathcal{S}} X_s + \frac{\lambda_p}{\mathbf{L}} \sum_{(l,n) \in \mathcal{L} \times \mathcal{N}} \omega_l \Omega_l D_{l,n} + \frac{\lambda_j}{\mathbf{G}} \sum_{g \in \mathcal{G}} (1 - \omega_g) I_g + \frac{\lambda_t}{\mathbf{G}} \sum_{s \in \mathcal{S}, g \in \mathcal{G}, \Lambda_{s,g} = 0} T_{s,g}. \tag{2.2}$$

Furthermore, keeping in mind the potential inaccuracies introduced, I did consider including duplicate genes in the model. To achieve this I proceeded as follows for each of the duplicate genes. First, the sequences of each gene in each reference strain were determined. Then one gene variant for each duplicate gene from each strain was selected by first choosing one from a starting strain and then iteratively including those gene variants for each strain with the lowest MASH distance to the genes already included. This approach led to a much larger number of genes (1587 genes in total of which 331 where core genes present in 100% of reference strains) included in the reference database. However, the performance of this approach was subpar. While variants of this approach could be developed, this avenue was not further pursued.

Supplementary Figures

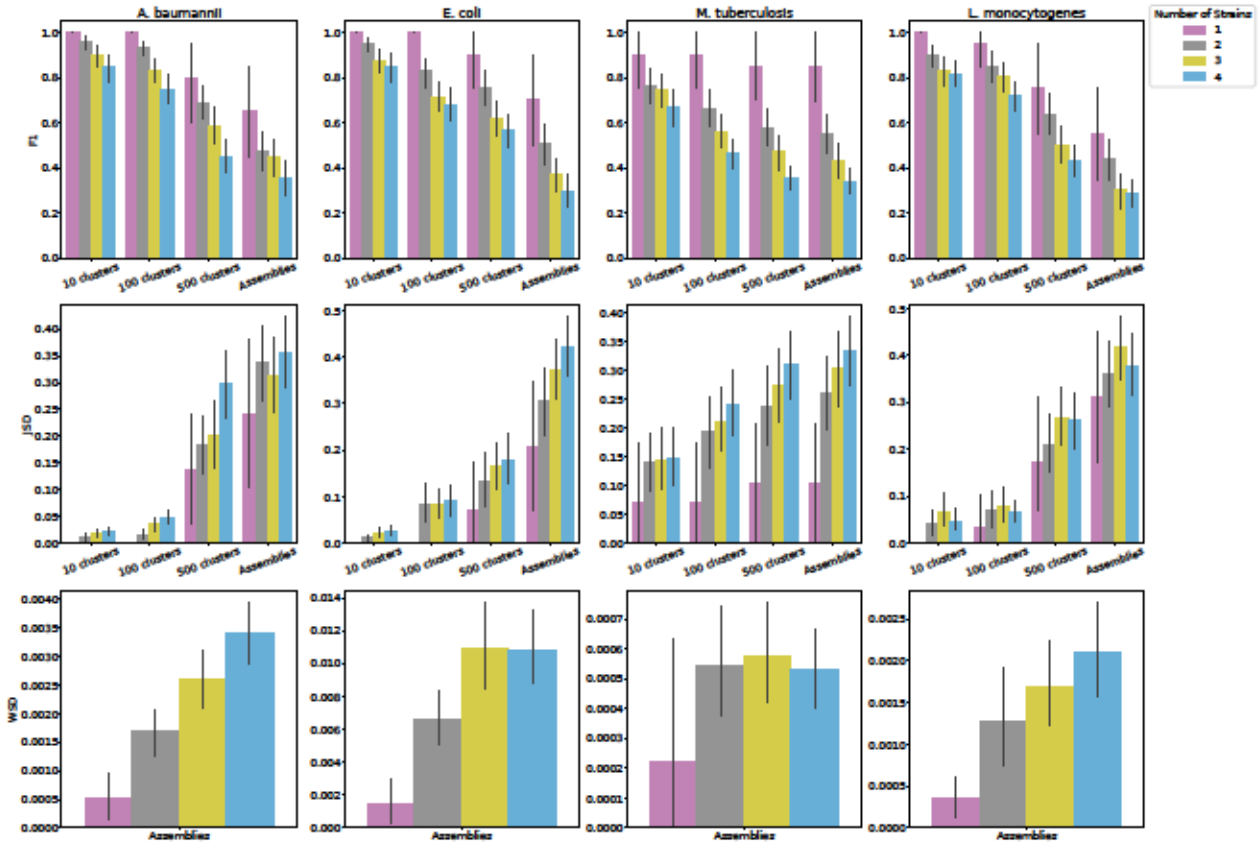


Figure B1: Performance of the APM for samples made up of strains contained in the reference database for all species (each column of figures corresponding to one species), stratified by the number of strains in the sample. Samples analyzed together include both samples with strains originating in the same and in different clusters at the 100 cluster level as well as samples with different coverages. Both samples that have equal relative abundance for all strains as well as the samples that have one dominant strain at 0.8 relative abundance with the remaining strains having equal shares of the remaining 0.2 relative abundance were analyzed in combination. Bar height shows mean performance and error bars the 95% confidence interval.

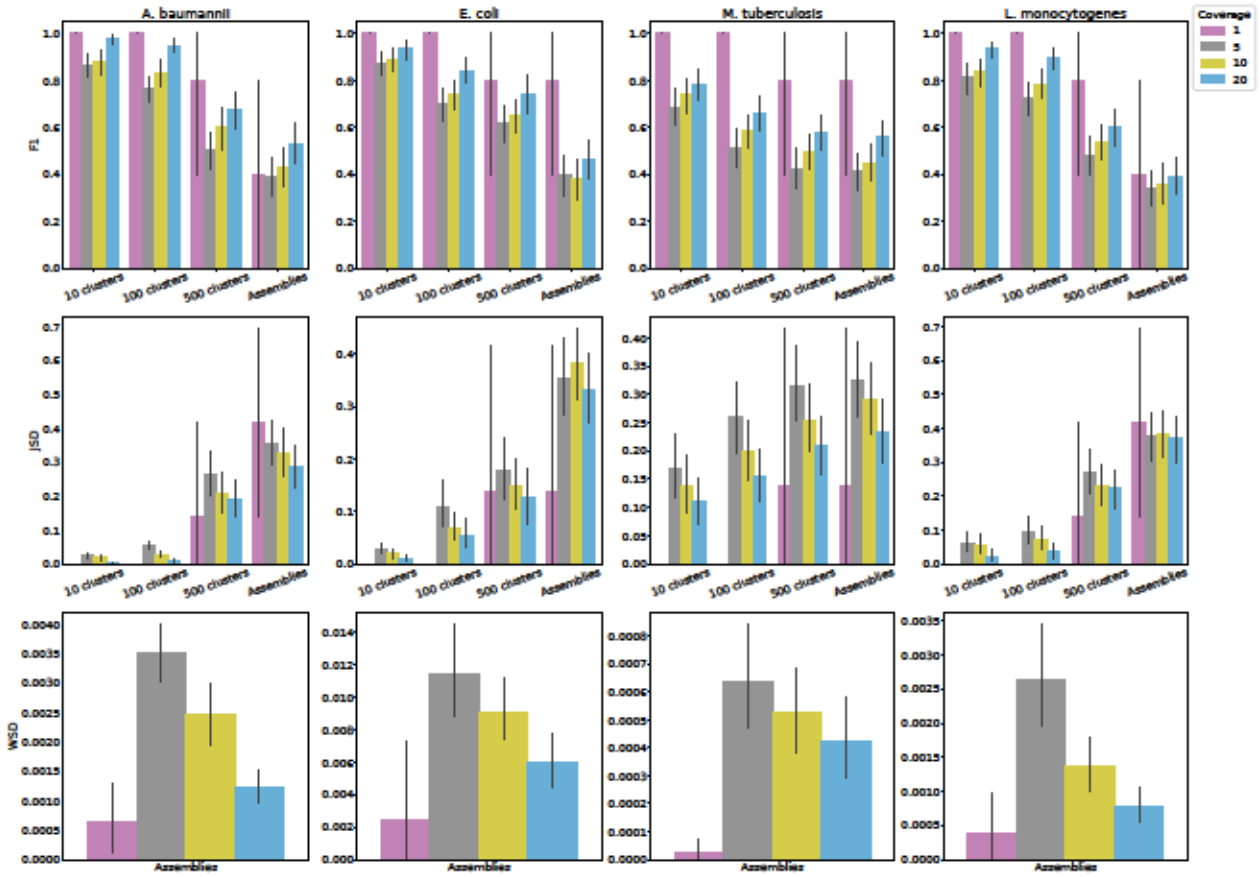


Figure B2: Performance of the APM for samples made up of strains contained in the reference database for all species (each column of figures corresponding to one species), stratified by the total coverage the sample. Samples analyzed together include both samples with strains originating in the same and in different clusters at the 100 cluster level as well as samples containing different number of strains. Both samples that have equal relative abundance for all strains as well as the samples that have one dominant strain at 0.8 relative abundance with the remaining strains having equal shares of the remaining 0.2 relative abundance were analyzed in combination. Bar height shows mean performance and error bars the 95% confidence interval. Note the high performance for 1-fold coverage. This is likely due to the fact that only samples with one constituent strain were created at this coverage.

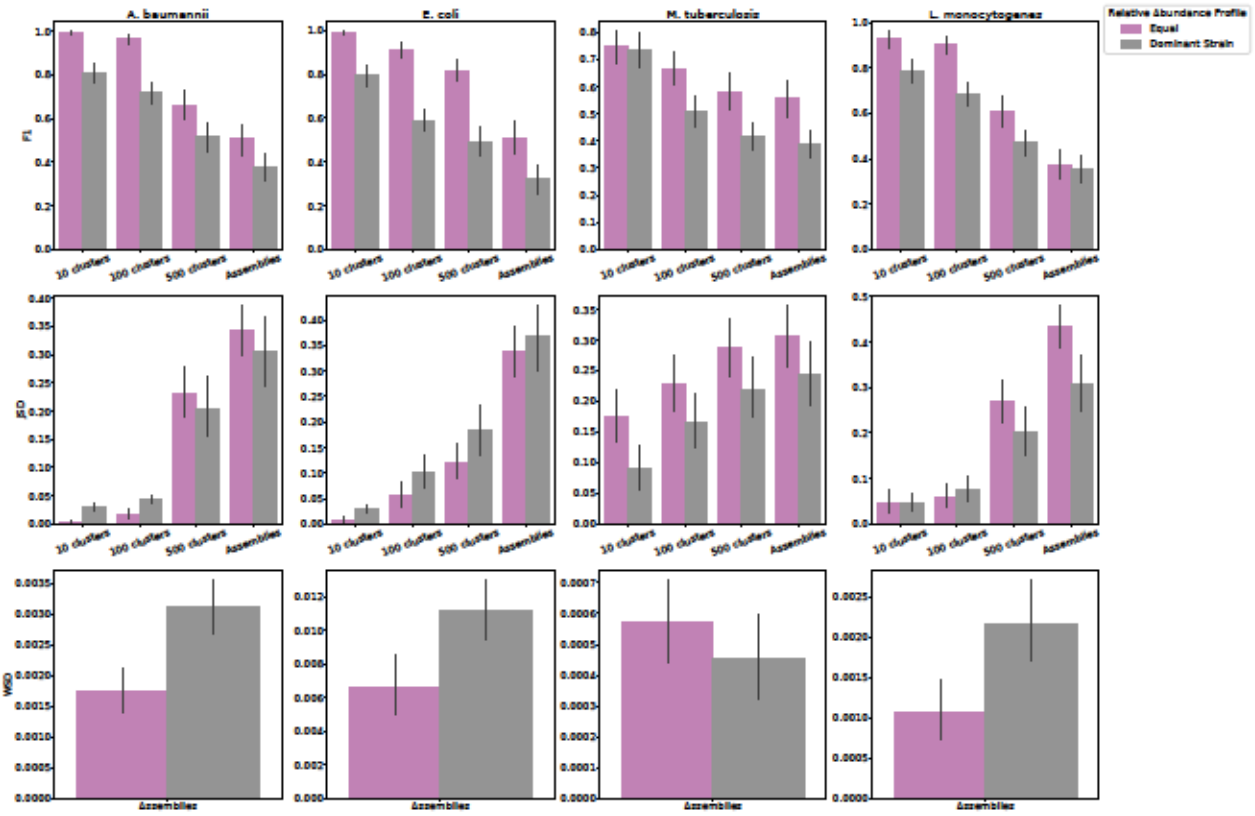


Figure B3: Results of the APM for samples made up of strains contained in the reference database for all species (each column of figures corresponding to one species), stratified by by the relative abundance profile. In the equal profile all strains have equal relative abundance. In the dominant strain samples one strain's relative abundance is 80%, with all other strains being equally distributed across the remaining 20%. Samples analyzed together include both samples with strains originating in the same and in different clusters at the 100 cluster level as well as samples with different numbers of constituent strains and coverages. Bar height shows mean performance and error bars the 95% confidence interval.

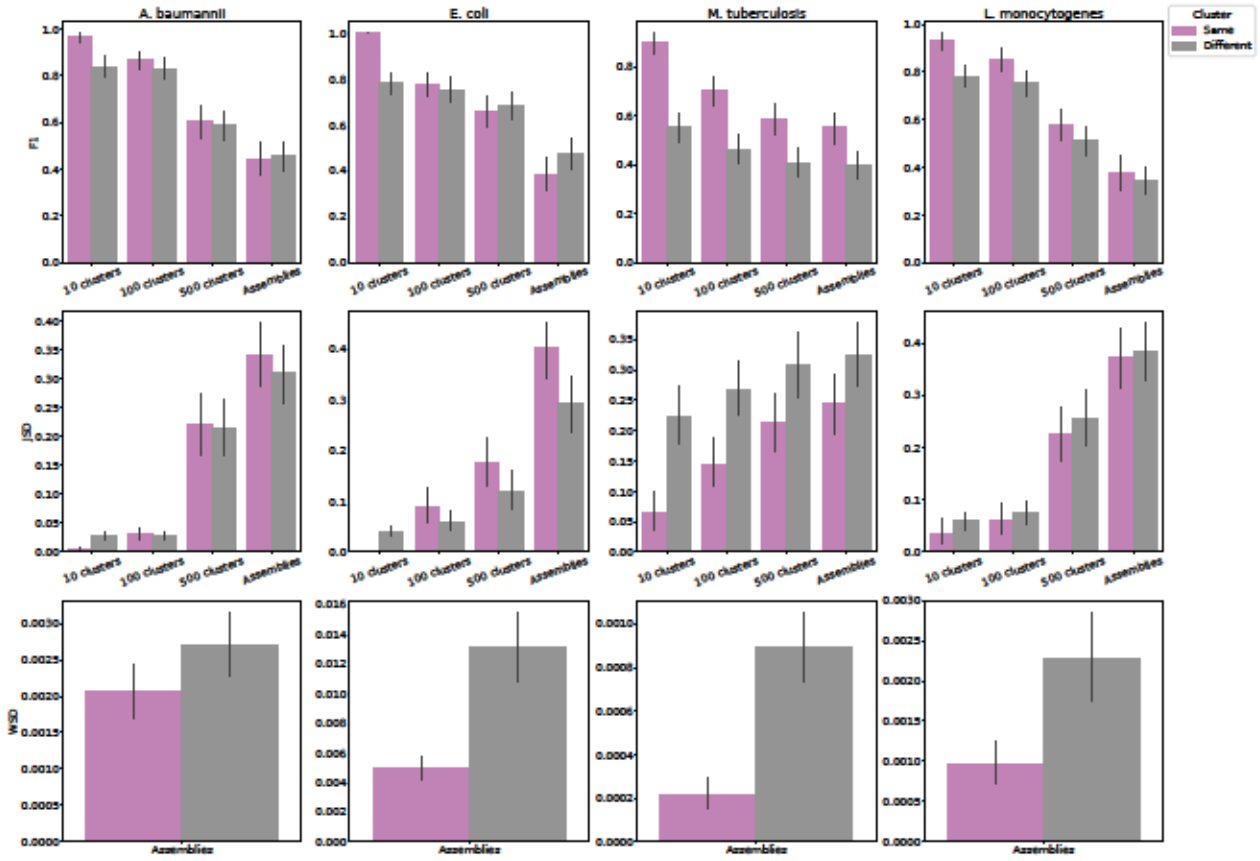


Figure B4: Results of the APM for samples made up of strains contained in the reference database for all species (each column of figures corresponding to one species), stratified by the cluster of origin. Same cluster indicates that the strains in the sample are in the same cluster at the 100 cluster level, while different indicate they originate from different clusters. Samples analyzed together include both samples with different numbers of constituent strains and coverages. Both samples that have equal relative abundance for all strains as well as the samples that have one dominant strain at 0.8 relative abundance with the remaining strains having equal shares of the remaining 0.2 relative abundance were analyzed in combination. Bar height shows mean performance and error bars the 95% confidence interval.

Appendix C

Derivation of NCOMP False Positive Rate

Here, I derive the expected number of false positives and false negatives in a k-disjunct multipool design (or the equivalent STD design) using the NCOMP decoder with $\Delta = 1$ and g the group size, defined as

Definition C.1. *For all columns i define $\mathcal{T}_i = \{j; M_{i,j} = 1\}$ and $\mathcal{S}_i = \{j; M_{i,j} = 1, Y_j = 1\}$. Then the ith item is inferred as positive if $|\mathcal{S}_i| \geq |\mathcal{T}_i|(g - \Delta)$.*

Furthermore, I make some necessary assumptions.

Assumption 1. Any two items X_i and X_k share at most one pool. This also means given the true status of X_i all the tests of pools containing X_i are conditionally independent.

Assumption 2. All items are i.i.d. positive with probability p.

Assumption 3. The sensitivity S_e and and specificity S_p are known and independent of group size and number of positives in a group.

Assumption 4. Given the true status of k slopes, their tests are all conditionally independent of each other.

With these assumptions, we can proceed to first derive the probability of sample i being false positive and then the probability of i being false negative, which will allow us to calculate our expected false positive and false negative rates given prevalence, sensitivity, specificity, and a test design.

We get the following for the false positive rate

$$\mathcal{P}(X_i \text{ negative} \mid \text{At least } k-1 \text{ pools containing } X_i \text{ positive}) = \frac{\mathcal{P}(\text{At least } k-1 \text{ pools containing } X_i \text{ positive} \mid X_i \text{ negative})\mathcal{P}(X_i \text{ negative})}{\mathcal{P}(\text{At least } k-1 \text{ pools containing } X_i \text{ positive})}.$$

with

$$\mathcal{P}(X_i \text{ negative}) = (1-p) = q. \tag{3.1}$$

We first have $\mathcal{P}(\text{At least } k-1 \text{ pools containing } X_i \text{ positive} \mid X_i \text{ negative}) = \mathcal{P}(\text{All k pools containing } X_i \text{ positive} \mid X_i$

negative) + $\mathcal{P}(k-1$ pools containing X_i positive $\mid X_i$ negative). The first part of this sum is

$$\mathcal{P}(\text{All k pools containing } X_i \text{ positive} \mid X_i \text{ negative}) =$$
$$\sum_{s_m=0}^{1} P(S_0 = 1, S_1 = 1, ..., S_{k-1} = 1 | S_0^T = s_0, S_1^T = s_1, ..., S_{k-1}^T = s_{k-1})$$
$$\times P(S_0^T = s_0, S_1^T = s_1, ..., S_{k-1}^T = s_{k-1}),$$

with $S_0, ..., S_{k-1}$ being the results of the tests of the k different slopes. S_m^T denotes the result that would have been obtained using perfect tests. By symmetry all summands where i S_m^Ts are 1 and j are 0 are the same. We can further take out the item, as it is negative. For true positive pools we hence get (by the assumption of conditional independence of different slopes under X_i)

$$S_e^a(1-q^{l-1})^a, \tag{3.2}$$

where l is the number of samples per pool and a is the number of true positives pools. For true negative pools we get

$$(1-S_p)^{k-a}q^{(k-a)l-(k-a)}. \tag{3.3}$$

Putting this all together we get

$$\begin{aligned}\mathcal{P}(\text{All k pools containing } X_i \text{ positive} \mid X_i \text{ negative}) = \\ S_e^k(1-q^{l-1})^k \\ +\binom{k}{k-1}S_e^{k-1}(1-q^{l-1})^{k-1}(1-S_p)q^{l-1} \\ +\binom{k}{k-2}S_e^{k-2}(1-q^{l-1})^{k-2}(1-S_p)^2q^{2l-2} \\ +...+\binom{k}{0}(1-S_p)^kq^{kl-k} \\ =\left(S_e(1-q^{l-1})+(1-S_p)q^{l-1}\right)^k\end{aligned}$$

The second part is

$$\mathcal{P}(k-1 \text{ pools containing } X_i \text{ positive} \mid X_i \text{ negative})$$
$$= k * \sum_{s_m=0}^{1} \mathcal{P}(S_0 = 0, ..., S_{k-1} = 1 | S_0^T = s_1, ..., S_{k-1}^T = s_{k-1}).$$

Similar to above for true positive pools we get

$$\begin{aligned}S_e^{a-1}(1-q^{l-1})^{a-1} \times (1-S_e)(1-q^{l-1}) \\ = S_e^{a-1}(1-S_e)(1-q^{l-1})^a,\end{aligned}$$

and for true negative pools we get

$$(1-S_p)^{k-a}q^{(k-a)l-(k-a)}. \tag{3.4}$$

Together this leaves us with

$$\begin{aligned}
&\mathcal{P}(k-1 \text{ pools containing } X_i \text{ positive} \mid X_i \text{ negative}) = k\times \\
&S_e^{k-1}(1-S_e)(1-q^{l-1})^k \\
&+\left(\binom{k}{k-1}-\binom{k-1}{k-1}\right)S_e^{k-2}(1-S_e)(1-q^{l-1})^{k-1}\times(1-S_p)q^{l-1} \\
&+Se^{k-1}(1-q^{l-1})^{k-1}\times S_pq^{l-1} \\
&+\left(\binom{k}{k-2}-\binom{k-1}{k-2}\right)S_e^{k-3}(1-S_e)(1-q^{l-1})^{k-2}(1-S_p)^2q^{2l-2} \\
&+\binom{k-1}{k-2}Se^{k-2}(1-q^{l-1})^{k-2}\times(1-S_p)S_pq^{2l-2} \\
&+\dots \\
&+\left(\binom{k}{k-(k-1)}-\binom{k-1}{k-(k-1)}\right)(1-S_e)(1-q^{l-1})\times(1-S_p)^{k-1}q^{(k-1)l-(k-1)} \\
&+\binom{k-1}{k-(k-1)}S_e(1-q^{l-1})\times(1-S_p)^{k-2}S_pq^{(k-1)l-(k-1)} \\
&+\binom{k-1}{0}(1-S_p)^{k-1}S_pq^{kl-k} \\
&= \\
&S_e^{k-1}(1-S_e)(1-q^{l-1})^k \\
&+\binom{k}{k-1}S_e^{k-2}(1-S_e)(1-q^{l-1})^{k-1}\times(1-S_p)q^{l-1} \\
&+\dots+\binom{k}{k-(k-1)}(1-S_e)(1-q^{l-1})\times(1-S_p)^{k-1}q^{(k-1)l-(k-1)} \\
&+Se^{k-1}(1-q^{l-1})^{k-1}\times S_pq^{l-1} \\
&+\binom{k-1}{k-2}Se^{k-2}(1-q^{l-1})^{k-2}\times(1-S_p)S_pq^{2l-2} \\
&+\dots+\binom{k-1}{k-(k-1)}S_e(1-q^{l-1})\times(1-S_p)^{k-2}S_pq^{(k-1)l-(k-1)} \\
&+\binom{k-1}{0}(1-S_p)^{k-1}S_pq^{kl-k} \\
&-\Bigg[\binom{k-1}{k-1}S_e^{k-2}(1-S_e)(1-q^{l-1})^{k-1}\times(1-S_p)q^{l-1} \\
&+\binom{k-1}{k-2}S_e^{k-3}(1-S_e)(1-q^{l-1})^{k-2}(1-S_p)^2q^{2l-2} \\
&+\dots+\binom{k-1}{k-(k-1)}(1-S_e)(1-q^{l-1})\times(1-S_p)^{k-1}q^{(k-1)l-(k-1)}\Bigg].
\end{aligned}$$

The first part of this is

$$\begin{aligned}
&S_e^{k-1}(1-S_e)(1-q^{l-1})^k \\
&+\binom{k}{k-1}S_e^{k-2}(1-S_e)(1-q^{l-1})^{k-1}\times(1-S_p)q^{l-1} \\
&+\dots+\binom{k}{k-(k-1)}(1-S_e)(1-q^{l-1})\times(1-S_p)^{k-1}q^{(k-1)l-(k-1)} \\
&=\frac{(1-S_e)}{S_e}\left[\left(S_e(1-q^{l-1})+(1-S_p)q^{l-1}\right)^k-(1-S_p)^kq^{kl-k}\right].
\end{aligned}$$

Further

$$\begin{aligned}
&Se^{k-1}(1-q^{l-1})^{k-1}\times S_pq^{l-1}\\
&+\binom{k-1}{k-2}Se^{k-2}(1-q^{l-1})^{k-2}\times(1-S_p)S_pq^{2l-2}\\
&+\ldots+\binom{k-1}{k-(k-1)}S_e(1-q^{l-1})\times(1-S_p)^{k-2}S_pq^{(k-1)l-(k-1)}\\
&+\binom{k-1}{0}(1-S_p)^{k-1}S_pq^{kl-k}\\
&=S_pq^{l-1}\Big(S_e(1-q^{l-1})+(1-S_p)q^{l-1}\Big)^{k-1},
\end{aligned}$$

and

$$\begin{aligned}
&\binom{k-1}{k-1}S_e^{k-2}(1-S_e)(1-q^{l-1})^{k-1}\times(1-S_p)q^{l-1}\\
&+\binom{k-1}{k-2}S_e^{k-3}(1-S_e)(1-q^{l-1})^{k-2}(1-S_p)^2q^{2l-2}\\
&+\ldots+\binom{k-1}{k-(k-1)}(1-S_e)(1-q^{l-1})\times(1-S_p)^{k-1}q^{(k-1)l-(k-1)}\\
&=(1-S_p)q^{l-1}\frac{(1-S_e)}{S_e}\left[\Big(S_e(1-q^{l-1})+(1-S_p)q^{l-1}\Big)^{k-1}-(1-S_p)^{k-1}q^{(k-1)l-(k-1)}\right].
\end{aligned}$$

Altogether we arrive at

$$\begin{aligned}
&\mathcal{P}(k-1\text{ pools containing }X_i\text{ positive}\mid X_i\text{ negative})\\
&=k\times\left\{\frac{(1-S_e)}{S_e}\left[\Big(S_e(1-q^{l-1})+(1-S_p)q^{l-1}\Big)^{k}-(1-S_p)^kq^{kl-k}\right]\right.\\
&\quad+S_pq^{l-1}\Big(S_e(1-q^{l-1})+(1-S_p)q^{l-1}\Big)^{k-1}\\
&\left.-(1-S_p)q^{l-1}\frac{(1-S_e)}{S_e}\left[\Big(S_e(1-q^{l-1})+(1-S_p)q^{l-1}\Big)^{k-1}-(1-S_p)^{k-1}q^{(k-1)l-(k-1)}\right]\right\}\\
&=k\times\left[\frac{(1-S_e)}{S_e}\Big(S_e(1-q^{l-1})+(1-S_p)q^{l-1}\Big)^{k}\right.\\
&\quad+S_pq^{l-1}\Big(S_e(1-q^{l-1})+(1-S_p)q^{l-1}\Big)^{k-1}\\
&\left.-(1-S_p)q^{l-1}\frac{(1-S_e)}{S_e}\Big(S_e(1-q^{l-1})+(1-S_p)q^{l-1}\Big)^{k-1}\right].
\end{aligned}$$

Then finally

$$\begin{aligned}
&\mathcal{P}(X_i\text{ is a false positive})\\
&=q\times\left\{\Big(S_e(1-q^{l-1})+(1-S_p)q^{l-1}\Big)^{k}\right.\\
&+k\times\left[\frac{(1-S_e)}{S_e}\Big(S_e(1-q^{l-1})+(1-S_p)q^{l-1}\Big)^{k}\right.\\
&\quad+S_pq^{l-1}\Big(S_e(1-q^{l-1})+(1-S_p)q^{l-1}\Big)^{k-1}\\
&\left.\left.-(1-S_p)q^{l-1}\frac{(1-S_e)}{S_e}\Big(S_e(1-q^{l-1})+(1-S_p)q^{l-1}\Big)^{k-1}\right]\right\}.
\end{aligned}$$

Having derived an expression for the false positive rate, we still need to derive one for the false negative rate. This is straight-

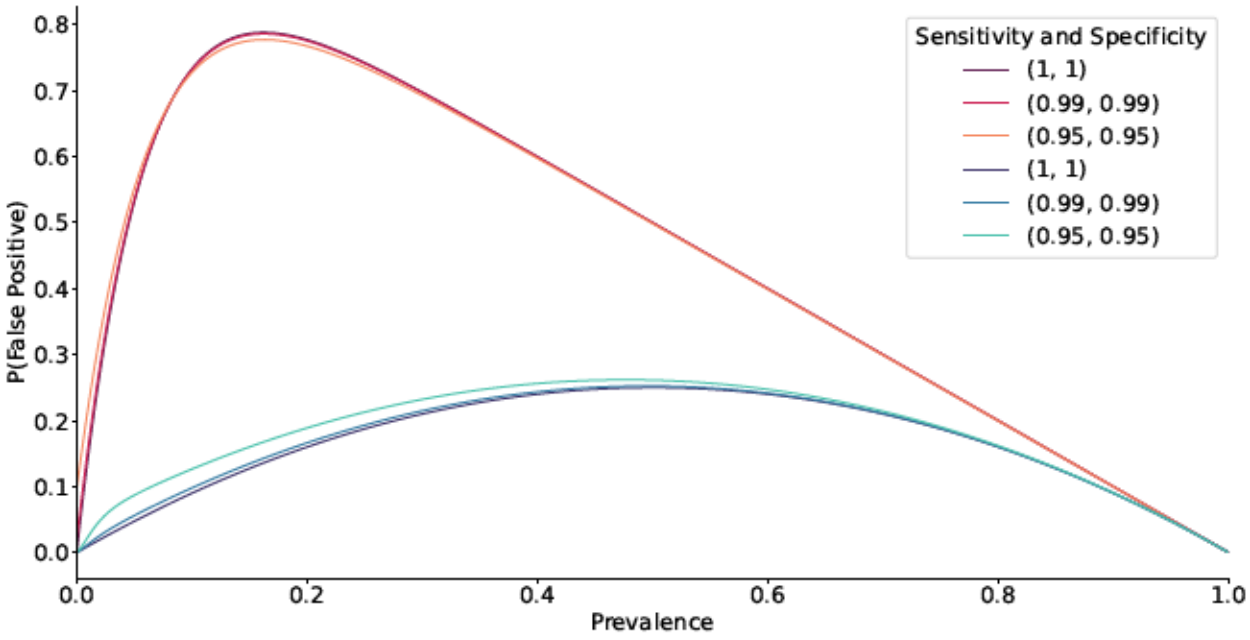


Figure C1: The probability of a sample being inferred as false positive at different prevalence rates and different sensitivity and specificity values for NCOMP (lines shaded red) and COMP (lines shaded blue).

forward and can be expressed as

$$\begin{aligned} &\mathcal{P}(\text{At least two pools containing } X_i \text{ negative} \mid X_i \text{ positive}) \\ &= 1 - \mathcal{P}(\text{At least } k-1 \text{ pools containing } X_i \text{ positive} \mid X_i \text{ positive}) \\ &= 1 - (S_e^k + k \times S_e^{k-1}(1 - S_e)). \end{aligned}$$

I compare the probability of a sample being false positive using NCOMP with COMP for array testing (as derived by Kim et al. [106]) in Figure C1.