UNIVERSITY OF CALIFORNIA SAN DIEGO

Distance-aware Algorithms for Scalable Evolutionary and Ecological Analyses

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy

in

Bioinformatics and System Biology

by

Metin Balaban

Committee in charge:

Professor Siavash Mir Arabbaygi, Chair
Vineet Bafna, Co-Chair
Professor Ronald Burton
Professor Melissa Gymrek
Professor Pavel Pevzner

2022

The Dissertation of Metin Balaban is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2022

DEDICATION

To advocates of open science and democratization of knowledge.

EPIGRAPH

Nothing in Biology Makes Sense Except in the Light of Evolution.

—Theodosius Dobzhansky

TABLE OF CONTENTS

Dissertation Approval Page iii

Dedication iv

Epigraph v

Table of Contents vi

List of Figures x

List of Tables xiv

Acknowledgements xvi

Vita xviii

Abstract of the Dissertation xx

Chapter 1 Introduction 1

Chapter 2 TreeCluster: Clustering Biological Sequences using Phylogenetic Trees . . . 5
2.1 Introduction 6
2.2 Materials and methods 8
2.2.1 Algorithms 8
2.2.2 TreeCluster software 14
2.2.3 Three applications of TreeCluster 15
2.3 Results 24
2.3.1 Results for Application 1: OTU clustering 24
2.3.2 Results for Application 2: HIV dynamics 26
2.3.3 Results for Application 3: improving PASTA 27
2.4 Discussion 28
2.5 Conclusion 32
2.6 Acknowledgement 32

Chapter 3 APPLES: Scalable Distance-based Phylogenetic Placement with or without Alignments 36
3.1 Introduction 37
3.2 Description 39
3.2.1 Background and notations 39
3.2.2 Problem Statement 41
3.2.3 Linear Time Algorithm 41
3.2.4 APPLES Software 44
3.3 Benchmark 44
3.3.1 Datasets 44

3.3.2 Methods Compared . . . 48
3.3.3 Evaluation Procedure . . . 49
3.4 Benchmark Results . . . 50
3.4.1 Assembly-free Placement of Genome-skims . . . 50
3.4.2 Alignment-based Placement . . . 51
3.4.3 Comparison to ML with Alignment Error . . . 58
3.5 Discussion . . . 59
3.5.1 Further observations on the results . . . 59
3.5.2 Observations on distance-based placement . . . 61
3.6 Availability . . . 62
3.7 Acknowledgment . . . 62

Chapter 4 Fast and Accurate Distance-based Phylogenetic Placement using Divide and Conquer . . . 70
4.1 Introduction . . . 71
4.2 Materials and Methods . . . 73
4.2.1 APPLES-2 Algorithm . . . 73
4.2.2 Experiments . . . 76
4.3 Results . . . 80
4.3.1 Single-gene Placement . . . 80
4.3.2 Multi-gene Web of Life (WoL) data set . . . 84
4.3.3 Placement of assemblies and scaffolds . . . 88
4.3.4 Placement of real MAGs and scaffolds onto WoL tree . . . 91
4.4 Discussion . . . 91
4.5 Acknowledgement . . . 96

Chapter 5 Phylogenetic double placement of mixed samples . . . 103
5.1 Introduction . . . 104
5.2 Approach . . . 106
5.2.1 Model . . . 106
5.2.2 Phylogenetic double-placement . . . 111
5.2.3 Solving the non-convex optimization problem . . . 113
5.3 Experimental Setup . . . 115
5.3.1 Datasets . . . 115
5.3.2 Distance calculation and backbone trees . . . 116
5.3.3 Evaluation . . . 117
5.4 Results . . . 118
5.4.1 Simulated mixture datasets . . . 118
5.4.2 Fungal Hybridization . . . 121
5.5 Discussion . . . 122
5.5.1 Relevant literature . . . 123
5.5.2 Shortcomings and future work . . . 124
5.6 Acknowledgement . . . 127

Chapter 6 Genome-wide alignment-free phylogenetic distance estimation under a no strand-bias model . . . 134
6.1 Introduction . . . 135
6.2 Approach . . . 137
6.2.1 Background information . . . 137
6.2.2 Containment Jaccard correction . . . 142
6.2.3 Calculation of TK4 terms via replacement . . . 144
6.2.4 Handling mixed-strand conditions . . . 146
6.2.5 NSB: TK4 distance estimation using *k*-mers . . . 148
6.3 Validation Results . . . 149
6.3.1 Simulation study . . . 149
6.3.2 Evaluation on biological bacterial data . . . 155
6.3.3 Evaluation on biological Yeast dataset . . . 157
6.4 Discussion . . . 158
6.5 Availability. . . . 161
6.6 Acknowledgement . . . 161

Chapter 7 Growing phylogenomic trees at the ultra-large scale using divide and conquer . . . 168
7.1 Introduction . . . 168
7.2 Results . . . 169
7.2.1 uDance overview . . . 169
7.2.2 Accuracy of uDance in Simulations . . . 172
7.2.3 Phylogenomic Reconstruction of 200,000 Microbial Genomes. . . . 176
7.3 Discussions . . . 181
7.4 Materials and Methods . . . 182
7.4.1 Workflow of uDance . . . 182
7.4.2 Simulations . . . 189
7.4.3 Biological Data . . . 191
7.4.4 Methods Compared . . . 194
7.4.5 Acknowledgements . . . 194

Chapter A Supplementary materials for "TreeCluster: Clustering Biological Sequences using Phylogenetic Trees" . . . 202
A.1 Proofs and supplementary algorithms . . . 202
A.1.1 Proofs for the Max-diameter min-cut partitioning problem . . . 202
A.1.2 Linear-time solution for the Sum-length min-cut partitioning problem . . . 204
A.1.3 Proofs for the Single-linkage min-cut partitioning problem . . . 206
A.1.4 Mean-diameter clustering with clade constraint . . . 209
A.2 Commands and parameters . . . 210

Appendix B Supplementary materials for "APPLES: Scalable Distance-based Phylogenetic Placement with or without Alignments" . . . 215
B.1 Proofs and derivations . . . 215

B.1.1 Proof of Lemma 1 216
B.1.2 Proof of Lemma 2. 217
B.1.3 Proof of Theorem 3 220
B.2 Supplementary Tables 220
B.3 Commands 223
B.3.1 Sampling Clades 223
B.3.2 Computing Distance Matrices 223
B.3.3 Backbone tree estimation 223
B.3.4 Backbone tree branch length re-estimation 224
B.3.5 Performing placement 225
B.3.6 Working with estimated backbone and query-to backbone alignments . . 226

Appendix C Supplementary materials for "Phylogenetic double placement of mixed samples" 228
C.1 Additional Methods 228
C.1.1 Proofs 228
C.1.2 Derivatives 229
C.2 Supplementary Tables 231

Appendix D Supplementary materials for "Genome-wide alignment-free phylogenetic distance estimation under a no strand-bias model" 233
D.1 Supplementary Methods 233
D.2 Supplementary Tables 233

Bibliography 236

LIST OF FIGURES

Figure 2.1. When the phylogenetic tree is ultrametric, clustering is trivial 7

Figure 2.2. Comparing Greengenes and TreeCluster 24

Figure 2.3. Effectiveness of transmission clustering 27

Figure 2.4. Alignment error for PASTA using the centroid and the mincut decompositions 28

Figure 2.5. An example showing that number of minimal clusterings under a diameter threshold can be exponential of number of leaves 30

Figure 2.6. Execution times of Cluster Picker, HIV-TRACE, and TreeCluster in log-scale 31

Figure 2.7. Comparison of various TreeCluster modes and Greengenes 33

Figure 2.8. Tree distance versus Hamming distance 34

Figure 2.9. An example showing that Mean-diameter min-cut partitioning is not conforming locality 35

Figure 3.1. Phylogenetic placement overview. 41

Figure 3.2. Accuracy on simulated data 52

Figure 3.3. Results on RNASim-VS 54

Figure 3.4. Scalability with respect to the number of queries 56

Figure 3.5. Impact of alignment error on placement accuracy 58

Figure 3.6. The reference biological trees obtained from Open Tree of Life (Drosophila and Anopheles) and from Boyd *et al.* (Columbicola) 63

Figure 3.7. APPLES versus pplacer on 5,000 backbone trees 64

Figure 3.8. Scalability with respect to the number of queries 64

Figure 3.9. Comparing various models of DNA evolution 65

Figure 3.10. The effect of imposing positivity constraint on error 66

Figure 3.11. Comparing APPLES versions 67

Figure 3.12. APPLES-HYBRID has higher accuracy on sparse RNAsim dataset 68

Figure 3.13. The effect of reestimating branch lengths of the backbone tree on accuracy 69

Figure 4.1. Results on RNASim-VS 81

Figure 4.2. Scalability with respect to the number of queries 83

Figure 4.3. Results on the WoL dataset 85

Figure 4.4. Results on WoL-metagenomic data set 89

Figure 4.5. Results on the real TD metagenomic dataset 90

Figure 4.6. Detecting erroneous placements 94

Figure 4.7. Clustering the backbone using TreeCluster 97

Figure 4.8. Impact of alignment trimming 98

Figure 5.1. Mixture model in evolutionary perspective 107

Figure 5.2. Demonstration of model properties 108

Figure 5.3. Results on real biological dataset. 118

Figure 5.4. Placement error on real biological dataset 120

Figure 5.5. Results on the Yeast (Saccharomyces) hybrid dataset 121

Figure 5.6. Expected proportion of $\frac{S_A \cap S_B \cap S_R \cap S_L}{S_A \cap S_B \cap S_R}$ when $\delta_{AB} = \delta_{AR} = \delta_{RB} = \delta$ 127

Figure 5.7. Model approximation accuracy 128

Figure 5.8. Phylogenetic placement overview 129

Figure 5.9. Trajectories of optimization function (i.e. OLS error) for yeast species on small backbone 129

Figure 5.10. Reference trees on biological datasets. 130

Figure 5.11. Contribution of each species in Columbicola dataset to the total LSE 131

Figure 5.12. Impact of kmer size to accuracy of double-placement on Drosophila (a) and Yeast (extended) (b) datasets 132

Figure 5.13. Placement error when constituents of simulated mixtures are both present, one present and one absent, or both absent in the reference set 133

Figure 5.14. Genomic contribution of the first constituent in the entire mixture 133

Figure 6.1. No strand-bias model of evolution overview 138

Figure 6.2. JC under-estimates TK4 140

Figure 6.3. Comparing the accuracy of distances estimated by different approaches on random and Yeast-based simulated genomes 152

Figure 6.4. Deviation from additivity measured for TK4 and JC models of evolution on the dataset of 40 8-taxa phylogenies simulated under GTR model 154

Figure 6.5. Results on 10 subsets of the bacterial dataset 156

Figure 6.6. Schematic diagram of our proposed pipeline 162

Figure 6.7. Estimating number of non-homologous (by random chance) matches between two genomes after replacements 162

Figure 6.8. The effect of parameter k on accuracy in simulated yeast genomes 163

Figure 6.9. Comparing the accuracy of distances estimated by different approaches on random and Yeast-based simulated genomes 164

Figure 6.10. Analysis of the yeast genome assemblies and skims using distances estimated under JC and TK4 models 165

Figure 6.11. Comparison of pairwise distances estimated using TK4 and JC models on 8 real Yeast genomes 166

Figure 6.12. Runtime comparion between NSB and Skmer in seconds 167

Figure 7.1. uDance overview. 170

Figure 7.2. Results on sum data set. 173

Figure 7.3. Main new trees 177

Figure 7.4. Various analyses comparing uDance trees. 180

Figure 7.5. Species tree estimation error measured using nRF on simulated data. 195

Figure 7.6. The 10K ASTRAL tree decorated with GTDB taxonomy. 196

Figure 7.7. ECDF of depth and the branch length of agreeing and disagreeing branches 197

Figure 7.8. All by all comparison between the 10k, 16k, 200k, and GTDB trees on NCBI defined phyla and super-phyla. 197

Figure 7.9. The distribution of number of marker genes per sequence in WoL2 dataset 198

Figure 7.10. Mainlines backbone selection strategy 198

Figure 7.11. Histogram of number of Archaea species in each marker gene 199

Figure 7.12. Determining APPLES-2 marker gene set 200

Figure 7.13. Outgroup taxa selection strategy. 200

Figure 7.14. Contamination detection using GUNC. 201

Figure 7.15. WoL heights distribution 201

Figure A.1. A sketch showing the setup for constructing the chain 207

LIST OF TABLES

Table 2.1. Number of singleton clusters (σ), total number of clusters (Σ), and maximum cluster size (*max*) for TreeCluster and GreenGenes for various thresholds 26

Table 3.1. Assembly-free placement of genome-skims 50

Table 3.2. Accuracy on RNASim-varied diameter dataset. 53

Table 3.3. Percentage of correct placements (shown as %) and the delta error (Δe) on the RNASim datasets with various backbone size (n) 53

Table 4.1. WoL based data sets 78

Table 4.2. Percentage of correct placements (shown as %) and the average placement error (Δe) on the RNASim-VS with various backbone size (n) 82

Table 5.1. Impact of k on the error of the backbone trees 117

Table 6.1. NSB: TK4 Distance estimation 150

Table 7.1. Properties of simulated dataset. 172

Table 7.2. test 176

Table B.1. GenBank accession numbers and URLs for the dataset of 22 Anopheles genomes 220

Table B.2. GenBank accession numbers and URLs for the dataset of 21 Drosophila genomes 221

Table B.3. GenBank accession numbers of microbial species used in contamination removal. 221

Table B.4. Assembly-free placement of genome-skims 222

Table C.1. List of SRRs and URLs for Drosophila species used in real data experiment. 232

Table C.2. GenBank accession numbers of yeast species/strains. 232

Table D.1. Different model conditions (MC), used for simulating genome sequences . . 234

Table D.2. Comparison of different methods to the ASTRAL tree on 10 sets of bacterial dataset 234

Table D.3. Distance error (deviation from additivity) for NSB-TK4 and Jellyfish-JC on bacterial dataset. 234

Table D.4. GenBank accession numbers of yeast species/strains. 235

Table D.5. Topological and distance error for NSB-TK4, Jellyfish-JC, and Skmer-JC on yeast dataset. 235

ACKNOWLEDGEMENTS

I owe most of my gratitude to my thesis advisor, Dr. Siavash Mirarab. Thank you for giving me the opportunity to be a part of your research group. I am deeply grateful for your support, care, flexibility, and mentorship in academic and non-academic sense. You lead me by your example of hard work and scientific rigor, and shaped me to be the scientist I am today. I cannot thank you enough for such a positive PhD experience.

I would like to thank my beloved wife, Melissa, whose charm, warmth and support gave me the motivation I needed to finish my dissertation in this final and crucial year. I would also like thank my parents and sisters for their support throughout my life and not minding me live abroad for years, far from home. Finally I would like to thank to my wife's family for their strong support during my studies in San Diego.

I would like to acknowledge and thank all of the members of the Mirarab Lab for their support and friendship over the past five years. I am grateful to have had the opportunity to work in such a supportive environment surrounded by several incredible bioinformaticians. I also specifically thank Yueyu Jiang and Shahab Sarmashghi their support on the various projects detailed in this dissertation. In addition, I was lucky to work alongside several graduate students during my five years in the lab: Chao Zhang, Erfan Sayyari, Uyen Mai, Eleonora Rachtman, Maryam Rabiee, and Niema Moshiri. The countless conversations greatly enriched my graduate school experience and I feel fortunate to have had the pleasure to work with all of them.

I want to thank Bernard Moret for introducing me to Computational Biology field. His influence on my research interests, research style, and personality is immeasurable. I really appreciate that he continued to give me helpful advice and support my career even after I started the PhD program.

I would like to further thank the rest of my doctoral committee members, Professor Vineet Bafna, Professor Ronald Burton, Professor Melissa Gymrek, and Professor Pavel Pevzner for their valuable suggestions and advice during my doctoral study.

Finally, I would like to make a formal acknowledgment to the collaborators and co-

authors of the papers that I published and used to write this dissertation:

Chapter 2, in full, is a reprint of the material as it appears in "Balaban, M., Moshiri, N., Mai, U., Jia, X., & Mirarab, S. (08 2019). TreeCluster: Clustering biological sequences using phylogenetic trees. PLOS ONE, 14(8), 1–20". The dissertation author was the primary investigator and first author of this paper.

Chapter 3, in full, is a reprint of the material as it appears in "Balaban, M., Sarmashghi, S., & Mirarab, S. (09 2019). APPLES: Scalable Distance-Based Phylogenetic Placement with or without Alignments. Systematic Biology, 69(3), 566–578". The dissertation author was the primary investigator and first author of this paper.

Chapter 4, in full, is a reprint of the material as it appears in "Balaban, M., Jiang, Y., Roush, D., Zhu, Q., & Mirarab, S. (2021). Fast and Accurate Distance-based Phylogenetic Placement using Divide and Conquer. Molecular Ecology Resources, (March), 1–15". The dissertation author was the primary investigator and first author of this paper.

Chapter 5, in full, is a reprint of the material as it appears in "Balaban, M., & Mirarab, S. (07 2020). Phylogenetic double placement of mixed samples. Bioinformatics, 36(Supplement_1), i335–i343". The dissertation author was the primary investigator and first author of this paper.

Chapter 6, in full, has been submitted for publication of the material as it may appear in "Balaban, M., Bristy, N. A., Faisal, A., Bayzid, M. S., & Mirarab, S. (2022). Genome-wide alignment-free phylogenetic distance estimation under a no strand-bias model. Bioinformatics Advances". The dissertation author was the co-primary investigator and co-first author of this paper.

Chapter 7, in full, is currently being prepared for submission for publication of the material. "Balaban, M., Jiang Y., Zhu Q., McDonald D., Knight R., Mirarab S. Growing phylogenomic trees at the ultra-large scale using divide and conquer.". The dissertation author was the primary investigator and first author of this paper.

VITA

2015	B. S. in Computer Engineering, Middle East Technical University, Ankara
2017	M. Sc. in Computer Science, EPFL, Lausanne
2022	Ph. D. in Bioinformatics and System Biology, University of California San Diego Diego

PUBLICATIONS

Hasan, N. B., Biswas, A., Balaban, M., Mirarab, S., & Bayzid, M. S. (2022). Fast and Accurate Branch Support Calculation for Distance-Based Phylogenetic Placements. L. Jin & D. Durand (edit.), Comparative Genomics (pp. 33–51). Cham: Springer International Publishing.

Jiang, Y., Balaban, M., Zhu, Q., & Mirarab, S. (04 2022). DEPP: Deep Learning Enables Extending Species Trees using Single Genes. Systematic Biology. doi:10.1093/sysbio/syac031

Sarmashghi, S., Balaban, M., Rachtman, E., Touri, B., Mirarab, S., & Bafna, V. (2021). Estimating repeat spectra and genome length from low-coverage genome skims with RESPECT. PLoS Computational Biology. doi:10.1101/2021.01.28.428636

Balaban, M., Jiang, Y., Roush, D., Zhu, Q., & Mirarab, S. (2021). Fast and Accurate Distance-based Phylogenetic Placement using Divide and Conquer. Molecular Ecology Resources, (March), 1–15. doi:10.1111/1755-0998.13527

Balaban, M., Bristy, N. A., Faisal, A., Bayzid, M. S., & Mirarab, S. (2021). Genome-wide alignment-free phylogenetic distance estimation under a no strand-bias model. bioRxiv. doi:10.1101/2021.11.10.468111. In review for Bioinformatics Advances.

Rachtman, E., Balaban, M., Bafna, V., & Mirarab, S. (2020). The impact of contaminants on the accuracy of genome skimming and the effectiveness of exclusion read filters. Molecular Ecology Resources, 20(3), 649–661. doi:10.1111/1755-0998.13135

Balaban, M., & Mirarab, S. (07 2020). Phylogenetic double placement of mixed samples. Bioinformatics, 36(Supplement_1), i335–i343. doi:10.1093/bioinformatics/btaa489

Balaban, M., Sarmashghi, S., & Mirarab, S. (09 2019). APPLES: Scalable Distance-Based Phylogenetic Placement with or without Alignments. Systematic Biology, 69(3), 566–578. doi:10.1093/sysbio/syz063

Balaban, M., Moshiri, N., Mai, U., Jia, X., & Mirarab, S. (08 2019). TreeCluster: Clustering biological sequences using phylogenetic trees. PLOS ONE, 14(8), 1–20. doi:10.1371/journal.pone.0221068

Doerr, D., Balaban, M., Feijão, P., & Chauve, C. (2017). The gene family-free median of three. Algorithms for Molecular Biology, 12(1), 14.

ABSTRACT OF THE DISSERTATION

Distance-aware Algorithms for Scalable Evolutionary and Ecological Analyses

by

Metin Balaban

Doctor of Philosophy in Bioinformatics and System Biology

University of California San Diego, 2022

Professor Siavash Mir Arabbaygi, Chair
Vineet Bafna, Co-Chair

Thanks to the advances in sequencing technologies in the last two decades, the set of available whole-genome sequences has been expanding rapidly. One of the challenges in phylogenetics is accurate large-scale phylogenetic inference based on whole-genome sequences. A related challenge is using incomplete genome-wide data in an assembly-free manner for accurate sample identification with reference to phylogeny. This dissertation proposes new scalable and accurate algorithms to address these two challenges. First, I present a family of scalable methods called TreeCluster for breaking a large set of sequences into evolutionary homogeneous clusters. Second, I present two algorithms for accurate phylogenetic placement of

genomic sequences on ultra-large single-gene and whole-genome based trees. The first version, APPLES, scales linearly with the reference size while APPLES-2 scales sub-linearly thanks to a divide-and-conquer strategy based on the TreeCluster method. Third, I develop a solution for assembly-free sample phylogenetic placement for a particularly challenging case when the specimen is a mixture of two cohabiting species or a hybrid of two species. Fourth, I address one limitation of assembly-free methods—their reliance on simple models of sequence evolution—by developing a technique to compute evolutionary distances under a complex 4-parameter model called TK4. Finally, I introduce a divide-and-conquer workflow for incrementally growing and updating ultra-large phylogenies using many of the ingredients developed in other chapters. This workflow (uDance) is accurate in simulations and can build a 200,000-genome microbial tree-of-life based on 388 marker genes.

Chapter 1

Introduction

Since Charles Darwin drew his first evolutionary tree in Transmutation of Species in 1837, representing the genealogy of related species, organisms, cells, or biological entities in the form of trees has been integral in evolutionary biology, ecology, and other fields in life sciences. Despite the early efforts to build evolutionary trees (phylogenies) based on qualitative traits such as phenotype and fossil records, today, phylogenetic trees are predominantly inferred from molecular data [88].

Inferring a phylogeny is not solely a hypothesis about the genealogy and how samples have evolved through time but is also a hierarchical and compact way of organizing and representing the relationship between the species or samples in a library of molecular data (e.g., DNA). For example, the library and phylogeny in question can be all SARS-CoV-2 strains sampled in a country [1]. A phylogeny and clusters defined by phylogenetic relationships of viral strains can help determine variants of importance or concern. Thanks to the advances in sequencing technologies in the last two decades, the databases of whole genome sequences such as viral and microbial genomes have been expanding rapidly. Following the expansion of these genomic databases and resources, there is a growing need for scalable algorithms that can process the amassing data. An ingredient often used in scalable algorithms is divide-and-conquer, breaking a large set of sequences into evolutionarily homogeneous clusters. In addition, clustering a large set of sequences with shared evolutionary history using a phylogenetic tree is an efficient and

natural approach to the problem. In Chapter 2, I define a family of optimization problems that, given a phylogenetic tree, I can compute the minimum number of clusters such that the inferred clusters adhere to constraints on their heterogeneity. I test these clustering algorithms on several bioinformatics applications, including viral transmission and clustering Operational Taxonomic Units (OTUs) in the microbial samples.

Today, some of the largest phylogenies ever built are microbial phylogenies. Since the last decade, extensive 16S marker sequence databases with hundreds of thousands and even millions of leaves have been published [48, 190]. Lately, genome-wide microbial reference trees with ten thousand species and more have become available. These reference resources are essential for sample identification and microbiome environmental sampling. Phylogenetic placement is one of the most common techniques for mapping unknown or novel sequences onto the reference database. Phylogenetic placement is the problem of finding the optimal position for a new query sequence on an existing reference(backbone) tree. Placement, as opposed to a de-novo reconstruction of the phylogeny, has two advantages: it is more scalable and more tolerant to noise or missing data in the query sequence. As the size of available reference trees used in these analyses continues to grow, there is a growing need for methods that place genomic sequences on ultra-large trees with high accuracy. In Chapter 3 and Chapter 4, I introduce the distance-based phylogenetic placement algorithm APPLES and its successor APPLES-2. In Chapter 3, I emphasized the flexibility of distance-based placement to analyze both assembled and unassembled environmental samples. On the other hand, in Chapter 4, I introduce improvements in time complexity and accuracy of distance-based placement and studied the feasibility and accuracy of whole-genome placement of microbial genomes and metagenome-assembled genomes (MAG). One of the strong results I present is that I can place a query genome onto the 10,575-genome microbial tree Zhu et al. (2019) published in under 4 seconds using 381 marker genes sampled globally from the whole microbial genome. It is feasible to phylogenetically place all assembled microbial genomes uploaded to RefSeq onto the reference tree of microbes. There are two shortcomings of phylogenetic placement in comparison

to de-novo phylogenetic reconstruction. First, the phylogenetic placement does not provide the resolution of relationships between query sequences. Secondly, the query sequences cannot refine and update the backbone. In Chapter7, I introduced a divide-and-conquer workflow for incrementally growing and updating ultra-large phylogenies. This workflow, uDance, can build a 200,000-genome microbial tree-of-life based on 388 marker genes, yielding an order of magnitude improvement in size over the tree from Zhu et al. (2019).

One emerging application of APPLES is sample identification using genome-skims [26]. A genome-skim is a shotgun sequence of the nuclear DNA with $0.5-2\times$ coverage per specimen. This coverage is shallow and insufficient for assembly; however, using Skmer [209], it is possible to estimate the genomic distance between two genome-skims or a genome-skim and a genome. In Chapter 3, I demonstrate that sample identification of genome skims through distance-based phylogenetic placement is possible, assuming that the specimen contains genomic material from a single species. An extension of this problem is to identify the mixed sample from two species that possibly share the same environment. The same problem arises when a genome-skim originates from a single allopolyploid hybrid species (e.g., yeast), where the hybrid inherits a complete set of chromosomes from two ancestor species. Detection of the ancestral species without requiring assembly poses a challenge. In Chapter 5, I introduce a model that relates the distances between a mixed sample and reference species to those between constituents and reference species. In addition, I present an algorithm to find the optimal phylogenetic placement of the mixture constituents onto a given reference tree using this model.

One of the main aspects of distance-based phylogenetics is the model used in the computation of evolutionary distance. Under the infinite sites assumption, each mutation falls on a different site in a genome. In other words, there are no reversals where a character mutates to another character, and during the course of evolution, it reverts to the previous one. When the infinite sites assumption holds, the evolutionary distance is simply the average nucleotide identity (ANI) between two genomes [100]. However, often this assumption is false and therefore relaxed so that the number of observed substitutions is less than the actual number of events. Jukes-Cantor

model [105] is a single parameter evolutionary model that allows obtaining evolutionary distance from the ANI. One limitation of assembly-free methods is their reliance on simplified models of sequence evolution such as Jukes-Cantor. Complex and multi-parametric models of evolution, such as the general time reversible (GTR) model [232] are the workhorse of alignment-based phylogenetics. In Chapter 6, I investigate the (im)possibility of computing genomic distances under complex model models of evolution in the assembly and alignment-free scenarios. In addition, I introduce a technique to compute evolutionary distances under a 4-parameter model called TK4, which shows promise in the simulated data sets as an improvement over the classic JC model.

Chapter 2

TreeCluster: Clustering Biological Sequences using Phylogenetic Trees

Clustering homologous sequences based on their similarity is a problem that appears in many bioinformatics applications. The fact that sequences cluster is ultimately the result of their phylogenetic relationships. Despite this observation and the natural ways in which a tree can define clusters, most applications of sequence clustering do not use a phylogenetic tree and instead operate on pairwise sequence distances. Due to advances in large-scale phylogenetic inference, we argue that tree-based clustering is under-utilized. We define a family of optimization problems that, given an arbitrary tree, return the minimum number of clusters such that all clusters adhere to constraints on their heterogeneity. We study three specific constraints, limiting (1) the diameter of each cluster, (2) the sum of its branch lengths, or (3) chains of pairwise distances. These three problems can be solved in time that increases linearly with the size of the tree, and for two of the three criteria, the algorithms have been known in the theoretical computer scientist literature. We implement these algorithms in a tool called TreeCluster, which we test on three applications: OTU clustering for microbiome data, HIV transmission clustering, and divide-and-conquer multiple sequence alignment. We show that, by using tree-based distances, TreeCluster generates more internally consistent clusters than alternatives and improves the effectiveness of downstream applications. TreeCluster is available at https://github.com/niemasd/TreeCluster.

2.1 Introduction

Homologous molecular sequences across different species or even within the same genome can show remarkable similarity due to their shared evolutionary history. These similarities have motivated many applications to first group the elements of a diverse set of sequences into *clusters* of set of sequences with high similarity for use in subsequent steps. The precise meaning of clusters depends on the application. For example, when analyzing 16S microbiome data, the standard pipeline is to use Operational Taxonomic Units (OTUs), which are essentially clusters of closely related sequences that do not diverge more than a certain threshold [58, 78, 211]. Another example is HIV transmission inference, a field in which a dominant approach is to cluster HIV sequences from different individuals based on their similarity (again using a threshold) and to use these clusters as proxies to define clusters of disease transmission [110, 196].

Shared evolutionary histories, which is the origin of similarity among homologous sequences, can be shown using phylogenetic trees. The phylogenetic tree can be inferred from sequence data, [87, 239] and recently developed methods can infer approximate maximum-likelihood (ML) phylogenetic trees in sub-quadratic time, enabling them to scale to datasets of even millions of sequences [188]. Moreover, accurate alignment of datasets with hundreds of thousands of species (a prerequisite to most phylogenetic reconstruction methods) is now possible using divide-and-conquer methods [167, 176].

Most existing sequence clustering methods use the pairwise distances among sequences as input but do not take advantage of phylogenetic trees. For example, the widely-used UCLUST [58] searches for a clustering that minimizes the Hamming distance of sequences to the cluster centroid while maximizing the Hamming distance between centroids. Several other clustering methods have been developed for various contexts, such as gene family circumscription [60, 130] and large protein sequence databases [223].

Using phylogenies for clustering has two potential advantages. i) Since phylogenies explicitly seek to infer the evolutionary history, phylogeny-based clustering has the potential

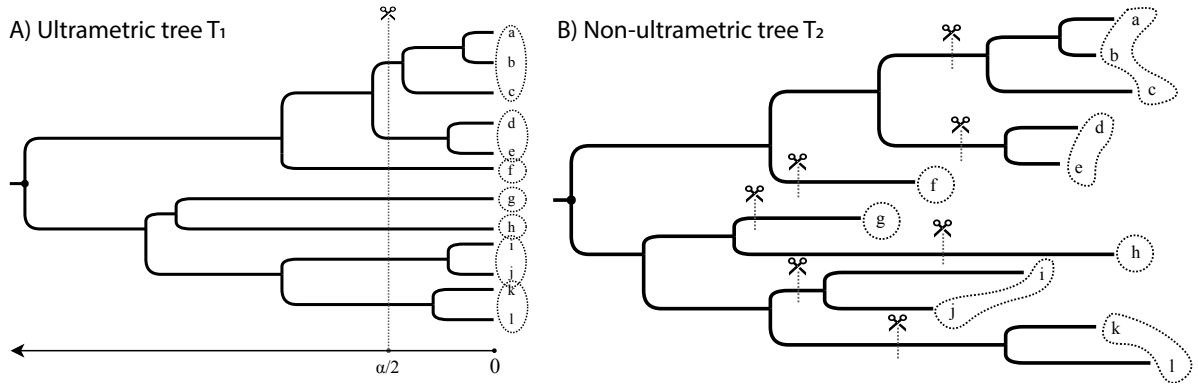


Figure 2.1. When the phylogenetic tree is ultrametric, clustering is trivial. For a threshold α, cut the tree at $\frac{\alpha}{2}$ height (A). When the tree is not ultrametric, it is not obvious how to cluster leaves (B). In both cases, a set of cut edges defines a clustering.

to not only reflect evolutionary distances (i.e., branch lengths) but also relationships (i.e., the tree topology). Recall also that branch lengths in a phylogeny are model-based "corrections" of sequence distances in a statistically-rigorous way [175, 239], and therefore, may better reflect divergence between organisms. *ii*) When inferred using subquadratic algorithms, the tree can eliminate the need to compute all pairwise distances, which can improve speed and scalability. Moreover, a phylogeny often has to be inferred for purposes other than clustering and thus typically is readily available. However, despite these potentials, to our knowledge, no systematic method for phylogeny-guided clustering exists. Built for analyzing HIV transmissions, ClusterPicker [195] clusters sequences based on their distances while using the phylogenetic tree as a constraint; however, it still uses sequence (not tree) distances and scales cubically with respect to number of sequences in the worst case.

Given a rooted phylogenetic tree, if the tree is ultrametric (that is, distances of all the leaves to the root are identical), clustering sequences based on the tree can proceed in an obvious fashion: the tree can be cut at some distance from the root, thereby partitioning the tree into clusters (Fig 2.1A). This approach extends in natural ways to unrooted ultrametric trees by first rooting the tree at the unique midpoint and proceeding as before. However, inferred phylogenetic trees are rarely ultrametric. Different organisms can evolve with different rates of evolution, and

even when the rates are identical (leading to an ultrametric true tree), there is no guarantee that the inferred trees will be ultrametric. Given a non-ultrametric (and perhaps unrooted) tree, the best way to cluster sequences is not obvious (Fig 2.1B).

One way to approach tree-based clustering is to treat it as an optimization problem. We can define problems of the following form: "find the minimum number of clusters such that some criteria constrain each cluster." Interestingly, at least two forms of such optimization problems have been addressed as early as the 1970s by the theoretical computer science community, in the context of proving more challenging theorems. The *tree partitioning* problem is to cut a tree into the minimum number of subtrees such that the maximum path length between two nodes in the same subtree [184] or the sum of all edge weights in each subtree [118] is constrained by a given threshold. Both problems can be solved exactly using straightforward linear-time algorithms; however, to our knowledge, these algorithms are mostly ignored by bioinformaticians.

Here, we argue that a fast and efficient tree-based clustering approach can be beneficial to several questions in bioinformatics. In this paper, we introduce a family of tree partitioning problems and describe linear-time solutions for three instances of the problem (two of which correspond to the aforementioned max and sum problems with known algorithms). We then show that tree-based clustering can result in improved downstream biological analyses in three different contexts: defining microbial OTUs, HIV transmission clustering, and divide-and-conquer multiple sequence alignment.

2.2 Materials and methods

2.2.1 Algorithms

Problem definition

Let $T = (V, E)$ be an unrooted binary tree represented by an undirected acyclic graph with vertices V (each with degree one or three), weighted edges E, and leafset $\mathscr{L} \subset V$. We denote the path length between leaves u and v on T with $d_T(u, v)$ or simply $d(u, v)$ when clear

by context. The weight of an edge (u,v) (i.e., its branch length) is denoted by $w(u,v)$.

A clustering of the leaves of the tree T can be defined by cutting a subset of edges $C \subseteq E$. We define a partition $\{L_1, L_2 \cdots, L_N\}$ of $\mathscr{L}$ to be an *admissible* clustering if it can be obtained by removing some edge set C from E and assigning leaves of each of the resulting connected components to a set L_i (note: $N \leq |C|+1$).

For a given tree T, let $f_T : 2^{\mathscr{L}} \rightarrow \mathbb{R}$ be a function that maps a subset of the leafset $\mathscr{L}$ to a real number. The purpose of $f_T(.)$ is to characterize the diversity of elements at the leaves within each cluster, and it is often defined as a function of the edge weights in the cluster. For example, it can be the diameter of a subset: $f_T(L) = \max_{u,v \in L} d_T(u,v)$. We define a family of problems that seek to minimize the number of clusters while each cluster has to adhere to constraints defined using $f_T(.)$. More formally:

Definition 1 (Min-cut partitioning problem family)**.** *Given a tree T with leafset $\mathscr{L}$ and a real number α, find an admissible partition $\{L_1 \ldots L_N\}$ of $\mathscr{L}$ that satisfies $\forall i, f_T(L_i) \leq \alpha$ and has the minimum cardinality (N) among all such clusterings.*

A natural way to limit the diversity within a cluster is to constrain all pairwise distances among members of the cluster to be less than a given threshold:

Definition 2 (Max-diameter min-cut partitioning problem)**.** *The Min-cut partitioning problem (Definition 1) is called Max-diameter min-cut partitioning problem when $f_T(L) = \max\limits_{u,v \in L} d(u,v)$.*

One potential disadvantage of max diameter min-cut partitioning is its susceptibility to outliers: the largest distance within a cluster may not be always an accurate representation of the degree of diversity in the cluster. A natural choice that may confine the effect of outliers is the following:

Definition 3 (Sum-length min-cut partitioning problem)**.** *The Min-cut partitioning problem is called Sum-length min-cut partitioning problem when $f_T(L) = \sum\limits_{(u,v) \in edges(T|L)} w(u,v)$ where $T|L$ is the tree T restricted to a subset of leaves L.*

We also study a third problem, which we will motivate later:

Definition 4 (Single-linkage min-cut partitioning problem)**.** *The Min-cut partitioning problem is called Single-linkage min-cut partitioning problem when* $f_T(L) = \max_{S \subset L} \{ \min_{u \in S, v \in L-S} d(u,v) \}$.

Next, we will show linear-time algorithms for the Max-diameter, Sum-length, and Single-linkage min-cut partitioning problems. All three algorithms use variations of the same greedy algorithm and two of them (max and sum) have already been described in the theoretical computer science literature. Nevertheless, we reiterate the solutions using consistent terminology and provide alternative proofs of their correctness.

Linear-time solution for Max-diameter min-cut partitioning

A linear-time solution for the Max-diameter min-cut partitioning problem was first published by Parley *et al.* [184] (with all edge weights equal to 1). We present Algorithm 1, which is similar to the Parley *et al.* algorithm (but adds branch lengths), and we give an alternative proof. The algorithm operates on T^o, which is an arbitrary rooting of T at node o. We denote the subtree rooted at an internal node u as U. Let the two children of u be called u_l and u_r, and let the tree rooted by them be U_l and U_r. We use w_l and w_r to denote $w(u,u_l)$ and $w(u,u_r)$, respectively, when clear by context.

For a cut set C of the tree, we define $B(C,u)$ to be the length of the path from u to the most distant *connected* leaf in U in the clustering defined by C. The algorithm uses a bottom-up traversal of the tree and for each node u that we visit, we may decide to cut one of its child edges. Thus, at each stage, a current clustering C_u is defined; we use $B(u)$ a shorthand for $B(C_u,u)$. When we arrive at node u, one or more new paths form between the two trees U_r and U_l. Among those paths, the longest one has the length $B(u_l)+w_l+B(u_r)+w_r$. If this value exceeds the threshold, we break either (u,u_r) or (u,u_l), depending on which minimizes $B(u)$. Note that the algorithm always cuts at most one child edge of every node, and thus, $B(u)$ is always well-defined.

Algorithm 1: Linear-time solution for Max-diameter min-cut partitioning

Input: A tree $T^o = (V, E)$ and a threshold α

```
B(v) ← 0 for v ∈ V
for u ∈ post order traversal of internal nodes of T^o do
    if B(u_l) + w_l + B(u_r) + w_r > α then
        if B(u_l) + w_l ≤ B(u_r) + w_r then
            E ← E − {(u, u_r)}
            B(u) ← B(u_l) + w_l
        else
            E ← E − {(u, u_l)}
            B(u) ← B(u_r) + w_r
    else
        B(u) ← max(B(u_l) + w_l, B(u_r) + w_r)
return Leafsets of every connected component in T^o
```

Theorem 1. *Let $A(u)$ be the minimum number of clusters under U, each with a diameter less than α (i.e., $A(o)$ is the objective function). Algorithm 1 computes a clustering with the minimum $A(o)$ for the rooted tree T^o. In addition, among all possible such clusterings, the algorithm picks* $\arg\min_C B(C, o)$.

Corollary 1. *Let C' be the cut set obtained by running Algorithm 1 on an arbitrary rooting T^o of tree T. C' optimally solves the Max-diameter min-cut partitioning problem.*

The proof of the theorem and the corollary are both given in Appendix A.

Linear solution for the Sum-length min-cut partitioning problem

A linear-time algorithm that partitions trees into the fewest clusters, each with total node weights less than or equal to α, has been previously published by Kundu *et al.* [118]. In order to solve the Sum-length min-cut partitioning problem, we present an altered version of the original algorithm that works on edge (instead of node) weights and that focuses on binary trees. Algorithm 1 with two simple modifications solves the Sum-length min-cut partitioning problem optimally (see Algorithm A in Appendix A). The first modification is that we define the auxiliary variable $B(C, u)$ to denote the sum of weights of all descendent edges connected to u at the stage

it is processed by the algorithm. Secondly, in the bottom-up traversal of internal nodes of T^o, for node u, w.l.o.g, let $B(u_l)+w_l \geq B(u_r)+w_r$. If the sum of branch lengths in the combined subtree exceeds α, we break the edge (u,u_l). Unlike Algorithm 1, where $B(u_l)+w_l+B(u_r)+w_r \leq \alpha$, here, $B(u)$ is set to $B(u_l)+w_l+B(u_r)+w_r$. The proof for the correctness of the algorithm is analogous to that of Algorithm 1 and is given in Appendix A.

Single-linkage min-cut partitioning

We now address the Single-linkage min-cut partitioning problem (Definition 4), which can be considered a relaxation of the Max-diameter min-cut partitioning. To motivate this problem, first consider the following definition.

Definition 5 (Single-linkage clustering)**.** *We call a partition of $\mathscr{L}$ to be a Single-linkage clustering when for every $a,b \in \mathscr{L}$, a and b are in the same cluster if and only if there exists a chain $\mathscr{H} = c_0, c_1, \ldots, c_m, c_{m+1}$, where $a = c_0$ and $b = c_{m+1}$, and for every $0 \leq i \leq m$, we have $d(c_i, c_{i+1}) \leq \alpha$.*

Thus, every pair of nodes is put in the same cluster if (but not only if) their distance is below the threshold (the rest follows from transitivity). The next result (proved in Appendix A.) motivates the choice of $f_T(.)$ in Definition 4.

Proposition 1. *The optimal solution to the Single-linkage min-cut partitioning problem (Definition 4) is identical to the Single-linkage clustering of Definition 5.*

Algorithm 2 shows a linear-time solution to the Single-linkage min-cut partitioning problem. For each node u, the algorithm first finds the closest leaf in the left and right sub-trees of u via post-order traversal, and it then finds the closest leaf outside the sub-tree rooted at u via pre-order traversal. Then, on a post-order traversal, it cuts each child edge iff the minimum distance of leaves under it to leaves under its sibling *and* to any leaf outside the node both exceed the threshold. The following theorem states the correctness of the algorithm (proof is given in Appendix A).

Algorithm 2: SINGLE-LINKAGE Single-linkage min-cut partitioning

$minBelow[u] \leftarrow minAbove[u] \leftarrow \infty$ **for** $v \leftarrow V$
for $u \in$ *post order traversal of* T^o **do**
  if u ***in*** $\mathscr{L}$ **then**
    $minBelow[u] \leftarrow 0$;
  else
    $minBelow[u] \leftarrow min\big(minBelow[u_l] + w_l, minBelow[u_r] + w_r\big)$;
for $u \in$ *pre order traversal of* T^o **do**
  if $u \neq o$ **then**
    $minAbove[u] \leftarrow min\big(minBelow[s] + w(v,s), minAbove[v] + w(v,v)\big)$;
for $u \in$ *post order traversal of internal nodes of* T^o **do**
  if $minBelow[u_l] + w_l + minBelow[u_r] + w_r > \alpha$ ***and*** $minBelow[u_l] + w_l + minAbove[u] > \alpha$ **then**
    $E \leftarrow E \setminus (u, u_l)$
  if $minBelow[u_l] + w_l + minBelow[u_r] + w_r > \alpha$ ***and*** $minBelow[u_r] + w_r + minAbove[u] > \alpha$ **then**
    $E \leftarrow E \setminus (u, u_r)$
  if $minBelow[u_l] + w_l + minAbove[u] > \alpha$ ***and*** $minBelow[u_r] + w_r + minAbove[u] > \alpha$ **then**
    $E \leftarrow E \setminus (v, u)$
return *Leafsets of every connected component in* T^o

Theorem 2. *The partitioning computed by Algorithm 2 optimally the solves Single-linkage min-cut partitioning problem (Definition 5).*

Clade constraint for rooted trees

So far, we have focused on unrooted trees. This choice is partially driven by the fact that phylogenetic reconstruction tools predominantly use time-reversible models of sequence evolution (e.g. GTR [232]) and therefore output an unrooted tree. Nevertheless, researchers have developed various methods for rooting trees [64, 109], including accurate and linear-time methods such as MV rooting [146]. When a rooted tree is available, each "monophyletic clade," i.e., group of entities that includes all descendants of their common ancestor, is a biologically meaningful unit. Thus, we may want to constrain each cluster to be a clade. These "clade" constraints make clustering easier: our algorithms can be easily altered to ascertain that each

cluster is also a clade. Specifically, in Algorithm 1, when we have $B(u_l) + w_l + B(u_r) + w_r > \alpha$, we simply need to cut both (u, u_l) and (u, u_r) (instead of cutting only the longer one). This small modification allows the Max-diameter, Sum-length, and Single-linkage min-cut partitioning problems to be solved in linear time while imposing the clade constraint.

Centroid (representative) sequence

Many sequencing clustering methods produce a representative sequence per cluster, often one that is used internally by the algorithm. Our clustering approach is representative-free. However, if a representative is needed for downstream applications, several choices are available. For example, one can in linear-time find the midpoint or balance point of a cluster [146] (i.e., the node that minimizes variance of root to tip distances); then, the leaf closest to the midpoint or balance point can be used as the representative. Another alternative is to use the consensus sequence among all sequences belonging to a cluster (i.e., choosing the most frequent letter for each site). Constructing and using a consensus sequence may be preferable to using one of the given sequences as the centroid[261]. A third alternative that we explore in our results is to use ancestral sequence reconstruction. For each subtree defined by a cluster, we first root it at its balance point. Then, we perform maximum likelihood ancestral state reconstruction (ASR) and use the reconstructed root sequence as the centroid.

2.2.2 TreeCluster software

We implemented linear-time algorithms for min-cut partitioning problem subject to Max-diameter, Sum-branch, Single-linkage, and other clustering criteria, with and without clade constraints in a freely-available open source tool called TreeCluster. TreeCluster takes a newick tree and a threshold value as input and returns clusters in a formatted text file. TreeCluster uses treeswift[169] package for fast tree operations.

2.2.3 Three applications of TreeCluster

While sequence clustering has many applications, in this paper, we highlight three specific areas as examples.

Application 1: OTU clustering

Biological Problem.

For microbiome analyses using 16S sequences generated from whole communities, the standard pipeline uses operational taxonomic units (OTUs). Sequences with similarity at or above a certain threshold (e.g. 97%) are grouped into OTUs, which are the most fine-grained level at which organisms are distinguished. All sequences assigned to the same OTU are treated as one organism in downstream analyses, such as taxonomic profiling, taxonomic identification, sample differentiation, or machine learning. The use of a similarity threshold instead of a biological concept of species is to avoid the notoriously difficult problem of defining species for microbial organisms [37, 212]. Futher, the use of clusters of similar sequences as OTUs can provide a level of robustness with respect to sequencing errors.

Most applications of OTUs are closed-reference: a reference database of known organisms is selected, and OTUs are defined for reference sequences using methods such as UCLUST[58] and Dotur[211]. These methods cluster sequences based on a chosen threshold of similarity, often picking a centroid sequence to represent an OTU. Reads from a 16S sample are then compared to the OTUs, and the closest OTU is found for each read (judging distance by sequence similarity). Once all reads are processed for all samples, an OTU table can be built such that rows represent samples, columns represent OTUs, and each cell gives the frequency of an OTU in a sample. This table is then used in downstream analyses. Several large reference databases exist for these OTU-based analyses [48, 147, 190]. One of these databases, popularized through pipelines such as Qiita [77], is Greengenes [48].

Regardless of the downstream application of an OTU table, one would prefer the OTUs to be maximally coherent (i.e., internally consistent) so they represent organisms as faithfully

as possible. We will focus our experiments on the closed-reference OTU picking methods and the Greengenes as the reference library. However, note that open-reference OTU picking and sub-operational-taxonomic-unit (sOTU) methods [5, 33, 59] also exist and involve a similar need for sequence clustering.

Existing methods.

Despite the availability of hierarchical clustering tools for OTU clustering [32, 211], non-hierarchical clustering methods [58, 131] are more widely used, perhaps due to their lower computational demand. Two prominent methods are UCLUST [58] and CD-HIT [131], which share the same algorithmic strategy: for a given threshold α, UCLUST determines a set of representative sequences dynamically by assigning query sequences into representative sequences (centroids) such that, ideally, the distance between each query and its assigned centroid is less than α while distances between centroids is more than α. UCLUST is a heuristic algorithm, and the processing order of the queries may affect the resulting clustering. CD-HIT differs from UCLUST primarily in its strategy for computing distances.

Formulation as min-cut partitioning.

We define OTUs by solving the Min-diameter, Sum-Length, or Single-linkage min-cut partitioning problems using a chosen threshold α and an inferred ML phylogeny. Each cluster in the resulting partition is designated as an OTU.

Experiments.

We evaluate the quality of tree-based OTU clustering by comparing it to UCLUST as used by Greengenes [48]. We run TreeCluster on the phylogenetic tree of 203,452 sequences in the Greengenes v13.5 database in three modes: max, sum, and single-linkage. We use the following 20 thresholds: $[0.005, 0.05]$ with a step size of 0.005, and $(0.05, 0.15]$ with a step size of 0.01. For single-linkage, we only go up to 0.1 because, above this threshold, the number of clusters becomes much smaller than other methods.

From the same Greengenes database, we extract OTU clusters for all available sequence identity thresholds up to 0.15 (i.e., 0.03, 0.06, 0.09, 0.12, and 0.15). We measure the quality of a clustering $\{L_1, \ldots, L_N\}$ by its weighted average of average pairwise distance per cluster (which we call *cluster diversity* for shorthand), given by the following formula:

$$\mu(\{L_1, \ldots, L_N\}) = \frac{\sum_{k=1}^{N} |L_k| \frac{\sum_{i,j \in L_k} d(i,j)}{|L_k|^2}}{\sum_{k=1}^{N} |L_k|} = \frac{1}{n} \sum_{k=1}^{N} \sum_{i,j \in L_k} \frac{d(i,j)}{|L_k|} \tag{2.1}$$

where n denotes the number of sequences clustered. We compute distance $d(i, j)$ between two elements using two methods: *tree distance*, which is the path length on the inferred phylogenetic tree, and sequence-based Hamming distance. Hamming distances are computed pairwise from the multiple sequence alignment of all 203,452 sequences in the Greengenes database and ignore any site that includes a gap in the pairwise alignment. Clearly, cluster diversity alone is insufficient to judge results (singletons have zero diversity). Instead, we compare methods at the same level of clustering with respect to their diversity. Thus, as we change the threshold α, we compare methods for choices of the threshold where they result in (roughly) equal numbers of clusters. Given the same number of clusters, a method with lower cluster diversity is considered preferable.

We measure the quality of a representative sequence set using two metrics. For a clustering $\{L_1, \ldots, L_N\}$, let $\{L_1, \ldots, L_{N'}\}$ denote all non-singleton clusters. The first metric is the average of average distance to the centroid per cluster, formally defined as:

$$\nu(g, \{L_1, \ldots, L_{N'}\}) = \frac{1}{N'} \sum_{k=1}^{N'} \sum_{i \in L_k} \frac{d(i, g(L_k))}{|L_k|} \tag{2.2}$$

where g is a function that maps a cluster to a (representative) sequence. The second metric is the

average of maximum distance to the representative per cluster, formally defined as:

$$\xi(g, \{L_1, \ldots, L_{N'}\}) = \frac{1}{N'} \sum_{k=1}^{N'} \max_{i \in L_k} d(i, g(L_k)) \tag{2.3}$$

We define these metrics on the set of non-singleton clusters because a trivial clustering which assigns many singletons will trivially have a very low value for ν and ξ (near zero).

Greengenes database is distributed with precomputed representative sequence sets. For centroid selection for TreeCluster, we consider two methods g: consensus and ASR. We perform ASR using TreeTime [206] under GTR model. We use RAxML 8 [220] to infer GTR model parameters from the Greengenes multiple sequence alignment of representative sequences at 15 percent threshold. We compute distance $d(i, j)$ between two elements using Hamming distance.

Application 2: HIV transmission cluster analyses

Biological Problem.

HIV evolves rapidly, so phylogenetic relationships between sequences contain information about the history of transmission [127]. The ability to perform phylogenetic analyses of HIV sequences is critical for epidemiologists who design and evaluate HIV control strategies [2, 93, 95, 124, 158]. The results of these analyses can provide information about the genetic linkage [62] and transmission histories [94], as well as mixing across subpopulations [23]. A recent advancement in computational molecular epidemiology is the use of transmission clustering to predict at-risk individuals and epidemic growth: infer transmission clusters from pairwise sequence distances, monitor the growth of clusters over time, and prioritize clusters with the highest growth rates [241]. In this monitoring framework, two natural questions come about: What is the optimal way to infer transmission clusters from molecular data, and how can transmission cluster inference be performed more efficiently?

Existing methods.

We focus on two popular tools that perform such clustering. Cluster Picker [196] is given a distance threshold, a phylogenetic tree, and sequences. It clusters individuals such that each cluster defines the leaves of a clade in the tree, the maximum pairwise sequence-based distance in each cluster is below the threshold, and the number of clusters is minimized. HIV-TRACE is a tool that, given a distance threshold and sequences, clusters individuals such that, for each pair of individuals u and v, if the Tamura-Nei 93 (TN93) distance [228] between u and v is below the threshold, u and v are placed in the same cluster [110]. Both methods scale worse than linearly with the number of sequences (quadratically and cubically, respectively, for HIV-TRACE and Cluster Picker), and for large datasets, they can take hours, or even days, to run (however, HIV-TRACE enjoys trivial parallelism and is fast in practice).

Formulation as min-cut partitioning.

Transmission clustering is similar to our problem formulation in that it involves cutting edges such that the resulting clusters (as defined by the leafsets resulting from the cuts) must adhere to certain constraints. Both Cluster Picker and HIV-TRACE utilize pairwise distances computed from sequences, but when reformulated to utilize tree-based distances from an inferred phylogeny, Cluster Picker becomes analogous to our Max-diameter min-cut partitioning (with an added constraint that clusters must define clades in the phylogeny), and HIV-TRACE becomes analogous to the Single-linkage min-cut partitioning.

Experiments.

To evaluate the effectiveness of HIV transmission clustering, we first simulate HIV epidemic data using FAVITES [170]. For the simulation parameters, we use the parameters described in Moshiri *et. al.* [170] to model the San Diego HIV epidemic between 2005 and 2014. However, we deviate from the original parameter set in one key way: originally, all HIV patients were sequenced at the end time of the epidemic, yielding an ultrametric tree in the unit of time, but to better capture reality, we instead sequence each patient the first time they receive

Antiretroviral Treatment (ART). In our simulations, we vary two parameters: the expected time to begin ART as well as the expected degree of the social contact network, which underlies the transmission network. Higher ART rates and lower degrees both result in a slower epidemic and change patterns of phylogenetic branch length [170]. The complete FAVITES parameter set can be found in the supplementary materials (List A in Appendix A). We infer phylogenies from simulated sequences under the GTR+Γ model using FastTree-II[188], and we use the MinVar algorithm to root the trees using FastRoot [144].

We use HIV-TRACE [110] as well as multiple clustering modes of TreeCluster to infer transmission clusters. We were unable to use Cluster Picker [196] due to its excessive running time. For HIV-TRACE, we use a clustering threshold of 1.5% as suggested by its authors [241]. Because HIV-TRACE estimates pairwise sequences distances under the TN93 model, [228] which tend to be underestimates of phylogenetic distance estimated under the GTR model, we use a clustering threshold of 3% for Single-Linkage TreeCluster. The default Cluster Picker threshold for Max-diameter clustering is 4.5% [196], so we use this as our clustering threshold for Max-Diameter TreeCluster (both with and without the Clade constraint). For Sum-length TreeCluster (with and without the Clade constraint), we simply double the Max-diameter threshold and use 9%. In addition to using these default thresholds, we also test a wide range of thresholds for each transmission clustering method for robustness.

We measure cluster growth from year 8 to year 9 of the simulation and select the 1,000 highest-priority individuals, where individuals are prioritized in descending order of respective cluster growth. To measure the risk of a given individual u, we count the number of HIV transmission events $u \to v$ between years 9 and 10. To measure the effectiveness of a given clustering, we average the risk of the selected top 1,000 individuals. Higher numbers imply the ability to prevent more transmissions by targeting a fixed number of individuals (1,000) and are thus desirable. As a control, we also show the mean number of transmissions per population, which is what a random selection of 1,000 individuals would give in expectation (we call this "expected" risk).

Application 3: Divide-and-conquer multiple sequence alignment

Algorithmic idea.

Tree-based clustering has also been used for multiple sequence alignment (MSA) using divide-and-conquer. To solve the MSA problem using divide-and-conquer, the tree structure can be used to divide sequences into smaller subsets (i.e., clusters), which can each be aligned separately and then merged. The phylogeny and the MSA can be inferred simultaneously by iterating between tree and MSA inference, and this technique has been used in algorithms such as SATe [138, 139] and PASTA [166]. Divide-and-conquer has been proven to be particularly useful for MSA of very large datasets [138, 167, 176]. We note that not all MSA tools use divide-and-conquer and that we only study the usage of min-cut partitioning in divide-and-conquer methods. We examine the effectiveness of min-cut partitioning in PASTA [166], a scalable software which infers both MSAs and trees for ultra-large datasets (tested for up to 1,000,000 sequences).

PASTA first builds a quick-and-dirty estimate of the phylogeny that is used as a guidance to cluster the sequences. In its "divide" phase, PASTA clusters the input sequences into subsets so that each subset contains less diverse sequences than the full set. Then, an accurate (but often computationally demanding) method is run on the subsets to infer the MSA and/or the tree. Finally, the results on the subsets are merged using various techniques. The accuracy of the output depends not only on the accuracy of the base method used on the subsets and the merging method, but also on the effectiveness of the method used to divide the tree into subsets [139].

PASTA computes an initial alignment using HMMs implemented in HMMER [57] and an initial tree using FastTree-II [188]; then, it performs several iterations (3 by default) of the divide-and-conquer strategy described before using MAFFT [107] for aligning subsets and using a combination of OPAL [244] and a technique using transitivity for merging subalignments. A tree is generated using FastTree-II at the end of each iteration, which is then used as the guide tree for the next iteration. The method has shown great accuracy on simulated and real data, especially in terms of tree accuracy, where it comes very close to the accuracy obtained using

the true alignment, leaving little room for improvement. However, in terms of the alignment accuracy, it has substantial room for improvement on the most challenging datasets.

The clustering used in PASTA is based on the centroid-edge decomposition. Given the *guide tree* (available from the previous iteration), the decomposition is defined recursively: divide the tree into two halves, such that the two parts have equal size (or are as close in size as possible). Then, recurse on each subtree until there are no more than a given number of leaves (200 by default) in each subset.

Formulation as min-cut partitioning.

The centroid edge decomposition involves cutting edges and includes a constraint defined on the subsets. However, it is defined procedurally and does not optimize any natural objective function. The min-cut partitioning can produce a decomposition similar to the centroid decomposition in its constraints but different in outcome. We set all edge weights of the guide tree to 1 and solve the Sum-length min-cut partitioning problem with threshold $\alpha = 2m - 2$; the result is a partition such that no cluster has more than m leaves and the number of subsets is minimized. Thus, this "max-size min-cut partitioning" is identical to centroid decomposition in its constraints but guarantees to find the minimum number of clusters.

Experiments.

To evaluate how our new decomposition impacts PASTA, we run PASTA version 1.8.3 on two datasets, and for each, we compare the accuracy of the two decomposition strategies: centroid and max-size min-cut partitioning. Other parameters (including maximum subset size) are all kept fixed for both decomposition strategies. We used two datasets both from the original PASTA paper: 10 replicates of a simulated RNAsim dataset with 10,000 leaves and a set of 19 real HomFam datasets with 10,099 to 93,681 protein sequences. The RNASim is based on a very complex model of RNA evolution. Here, the true alignment, known in simulations, is used as the reference. For HomFam, since the true alignment is not known, following previous papers, we rely on a very small number of seed sequences with a hand-curated reliable alignment as

reference [167, 217]. In both cases, we measure alignment error using two standard metrics computed using FastSP [162]: SPFN (the percentage of homologies in the reference alignment not recovered in the estimated alignment) and SPFP (the percentage of homologies in the estimated alignment not present in the reference).

2.3 Results

2.3.1 Results for Application 1: OTU clustering

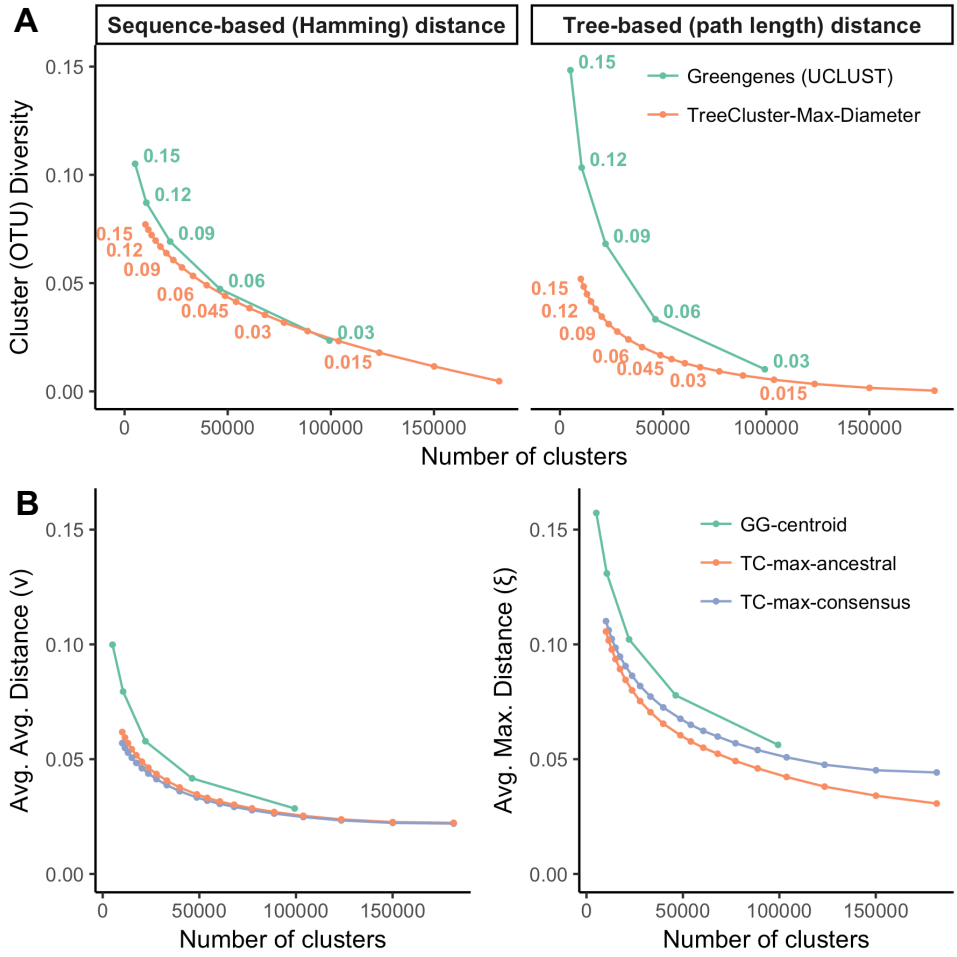


Figure 2.2. Comparing Greengenes and TreeCluster. (A) Cluster diversity (Eq. 2.1) for Greengenes and TreeCluster versus the number of OTUs. Cluster diversity is measured both with respect to hamming distance and tree-based distance. The threshold α is shown for all data points corresponding to Greengenes and for some points of TreeCluster. See Fig. 2.7 for comparison to other TreeCluster modes. (B) Average-average (v) and average-maximum (ξ) distance to the centroid for Greengenes and TreeCluster versus the number of clusters. TreeCluster centroids are computed using ancestral state reconstruction or using consensus.

On the Greengenes dataset, as we change the threshold between 0.005 and 0.15, we get between 181,574 and 10,112 clusters (note that singletons are also counted). The cluster diversity has a non-linear relationship with the number of clusters: it drops more quickly with

higher thresholds where fewer clusters are formed (Fig 2.2A and Fig. 2.7). Comparing the three objective functions that can be used in TreeCluster, we observe that Max-diameter and Sum-length have similar trends of cluster diversity scores, whereas Single-linkage min-cut partitioning has substantially higher diversity compared to the other two methods (Fig. 2.7). This pattern is observed regardless of whether tree distances or sequence distances are used, but differences are larger for tree distances. Finally, note that, even though tree distances are, as expected, larger than sequence distances (Fig. 2.8), the cluster diversity is *lower* when computed using tree distances, showing that clusters are tight in the phylogenetic space.

Compared to the default Greengenes OTUs, which are defined using UCLUST, Max-diameter min-cut partitioning defines tighter clusters for tree-based scores (Fig 2.2A). When distances between sequences are measured in tree distance, the cluster diversity score for Greengenes OTUs is substantially lower for all thresholds, and the gap is larger for higher thresholds. For example, the cluster diversity of Greengenes OTUs is three times higher than TreeCluster OTUs for $\alpha = 0.15$. When distances between sequences are measured in Hamming distance, Greengenes and TreeCluster perform similarly for low threshold values (e.g. $\alpha = 0.03$ for Greengenes, which is similar to $\alpha = 0.02$ for TreeCluster in terms of the number of clusters). However, when the number of OTUs is reduced, remarkably, TreeCluster outperforms Greengenes OTUs by up to 1.4-fold (e.g. $\alpha = 0.15$). This is despite the fact that UCLUST is working based on sequence distances and TreeCluster is not.

Size of the largest cluster in Greengenes is larger compared to TreeCluster (Table 2.1). For example, for $\alpha = 0.09$, both methods have similar number of clusters (22,090 and 23,631 for Greengenes and TreeCluster, respectively) but the size of largest cluster in Greengenes is three times that of TreeCluster (1,659 versus 540). On the other hand, for the same threshold value, the number of singleton clusters comprises 48% of all clusters for Greengenes whereas only 27% of the clusters are singletons for TreeCluster. Thus, GreenGenes has more clusters that are very small or very large, compared to TreeCluster.

Computed using either consensus or ASR method, representative sequences in TreeClus-

Table 2.1. Number of singleton clusters (σ), total number of clusters (Σ), and maximum cluster size (*max*) for TreeCluster and GreenGenes for various thresholds. In the Greengenes database, OTU definitions for thresholds $\alpha = 0.015$ and $\alpha = 0.045$ are not available.

	TreeCluster-Max-Diameter			**GreenGenes-UCLUST**		
α	σ	Σ	*max*	σ	Σ	*max*
0.015	86387	123456	47	(n.a)	(n.a)	(n.a)
0.03	42510	77263	96	70415	99322	527
0.045	24795	54068	171	(n.a)	(n.a)	(n.a)
0.06	15257	39809	305	26485	46256	894
0.09	6396	23631	540	10560	22090	1659
0.12	3003	15052	808	4153	10544	2131
0.15	1525	10112	1209	1735	5088	3765

ter are closer to other sequences of the cluster than Greengenes (Fig 2.2B). Using ASR representative sequences performs slightly worse than consensus centroids according to the v score (e.g. $v = 0.062$ and $v = 0.057$, respectively when $\alpha = 0.15$). When evaluated using ξ score, ASR representative sequences perform slightly better than consensus in all threshold levels (e.g. $\xi = 0.03$ and $\xi = 0.04$ respectively when $\alpha = 0.005$) and the gap again widens as the number of clusters increases. Both types of centroids computed using TreeCluster perform better than Greengenes representative sequences according to both metrics, and the gap increases as the threshold α increases (e.g. up to 1.7-fold when $\alpha = 0.15$ for v).

2.3.2 Results for Application 2: HIV dynamics

Comparing various TreeCluster modes, regardless of the parameters that we vary, Sum-length TreeCluster consistently outperforms the other clustering methods, and the inclusion of the Clade constraint has little impact on effectiveness (Fig 2.3). Compared to a random selection of individuals, the risk of selected individuals can be substantially higher; for example, with expected time to begin ART set to 1 year, the expected risk is 0.55 transmissions, whereas the average risk of the top 1,000 individuals from Sum-length clusters is 0.85. In all the conditions, a close second to TreeCluster Sum-length is TreeCluster Max-diameter. Other methods, however, are substantially less effective than these two modes of TreeCluster.

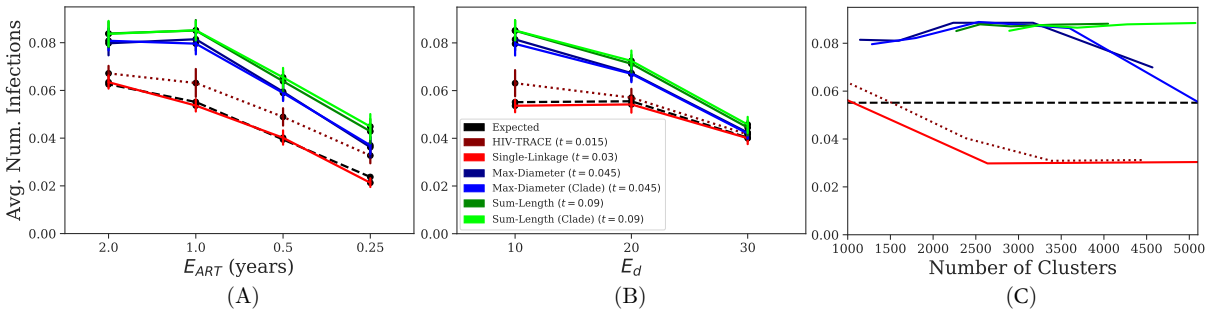


Figure 2.3. Effectiveness of transmission clustering. Effectiveness is measured as the average number of individuals infected by the selected 1,000 individuals. The horizontal axis depicts the expected time to begin ART (A), the expected degree (i.e., number of sexual contacts) for individuals in the contact network (B), and the number of clusters using various thresholds (C).

When varying expected time to begin ART and expected degree, Single-linkage TreeCluster and HIV-TRACE consistently perform lower than the other approaches, with Single-linkage TreeCluster typically performing around the theoretical expectation of a random selection and HIV-TRACE performing slightly better (Fig 2.3a-b). Moreover, these patterns are not simply due to the chosen thresholds: even when the threshold is changed to control the number of clusters, Single-linkage TreeCluster and HIV-TRACE consistently perform worse than expected by random selection (Fig 2.3c). The effectiveness of Sum-length and Max-diameter TreeCluster are maximized when they create 2,000–5,000 and 2,000–3,000 clusters, respectively.

2.3.3 Results for Application 3: improving PASTA

First, we notice a substantial improvement of PASTA 1.8.3 comparing to the original version published in 2015 [166], especially for RNASim where alignment accuracy improved by about 3%. This was due to major updates of the PASTA software and the dependent tools. When we replace centroid decomposition with Max-size min-cut partitioning in PASTA, the alignment error reduces substantially for the RNASim dataset, but less so on the HomFam dataset (Fig 2.4). On the RNAsim data, mean SPFN drops from 0.12 to 0.10, which corresponds to a 17% reduction in error. These drops are consistent across replicates and are substantial given the fact that the only change in PASTA was to replace its decomposition step with our new clustering

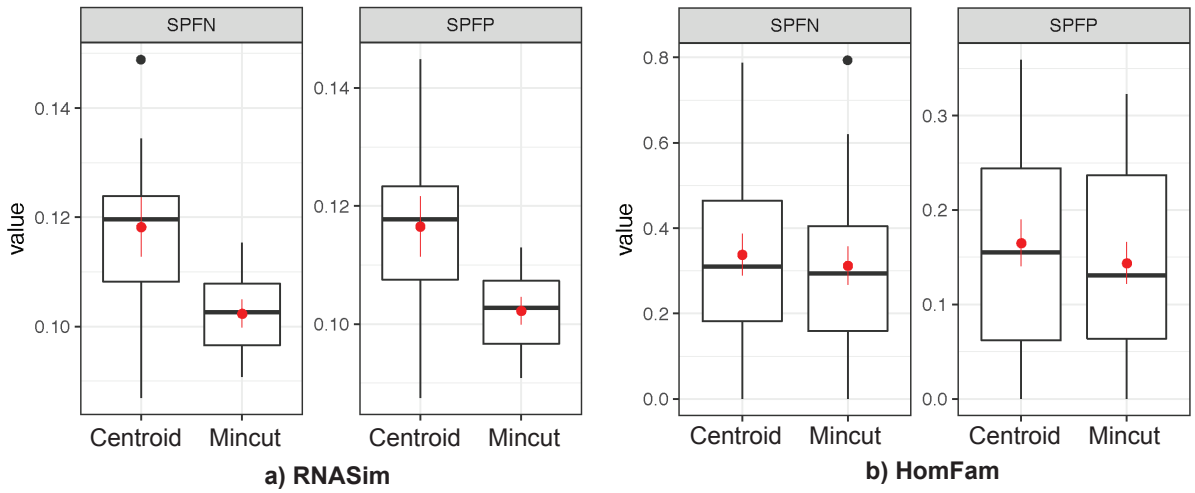


Figure 2.4. Alignment error for PASTA using the centroid and the mincut decompositions. We show Sum of Pairs False Negative (SPFN) and Sum of Pairs False Psotive (SPFP) computed using FastSP [162] over two datasetes: the simulated RNASim dataset (10 replicates) and the biological HomFam dataset (19 largest families; all 20 largest, except "rhv" omitted due to the warning on the Pfam website). We show boxplots in addition to mean (red dot) and standard error (red error bars).

algorithm, keeping the rest of the complex pipeline unchanged. In particular, the method to align subsets, to merge alignments, and to infer trees, were all kept fixed. On the HomFam dataset, too, errors decreased, but the reductions were not substantial (Fig 2.4b). Based on these results, we have now changed PASTA to use Max-size min-cut partitioning by default.

2.4 Discussion

Several theoretical and practical issues should be further discussed.

Mean-diameter min-cut partitioning

Some of the existing methods, such as Cluster Picker [196], can define their constraints based on mean pairwise distance between nodes. Similar to those, we can define a variation of the min-cut partitioning problem in which $f_T(L) = \frac{1}{\binom{|L|}{2}} \sum_{i,j \in L} d(i,j)$. Unfortunately, this "Mean-diameter" min-cut partitioning problem can only be solved in linear time using our greedy algorithm if we also have clade constraints (Algorithm B in Appendix A). As demonstrated

by the counterexample given in Fig. 2.9, the greedy algorithm fails if we do not have clade constraints. More generally, the use of mean as function $f_T(\cdot)$ creates additional complexity, and whether it can be solved in linear time remains unclear. Whether mean diameter is in fact a reasonable criterion is not clear. For example, it is possible that the mean diameter of a cluster is below the threshold while the mean diameter of subclusters embedded in that cluster are not; such scenarios may not make sense for downstream applications.

Set of optimal solutions.

It is possible that multiple distinct partitions with equal number of clusters are all optimal solutions to any of our min-cut partitioning problems. Moreover, as the example given in Fig 2.5 shows, the number of optimal solutions can be exponential with respect to number of leaves in a binary phylogenetic tree. This observation renders listing all optimal solutions potentially impractical as there may be too many of them. However, finding a way to summarize all optimal partitions remains interesting and can have practical utility. We do not currently have such a summarization approach. However, as shown in Lemma A of Appendix A, although the optimal solution space is potentially exponentially large, one can easily determine the set of all edges that could appear in any of the optimal solutions. Thus, we could find absolutely unbreakable edges that will not be cut in any optimal clustering of the data.

Choice of criterion.

Among the three methods that we discussed, we observed that Max-diameter and Sum-diameter are consistently better than the Single-linkage. This observation makes intuitive sense. Single-linkage can increase the diversity within a cluster simply due to the transitive nature of its criterion. Thus, a very heterogeneous dataset may still be collapsed into one cluster, simply due to transitivity. Our desire to solve the Single-linkage problem was driven by the fact that a similar concept is used in HIV-TRACE, arguably the most widely used HIV clustering method. However, we did not detect any advantage in this type of clustering compared to Max-diameter or Sum-length; thus, our recommendation is to use these two criteria instead. Between the

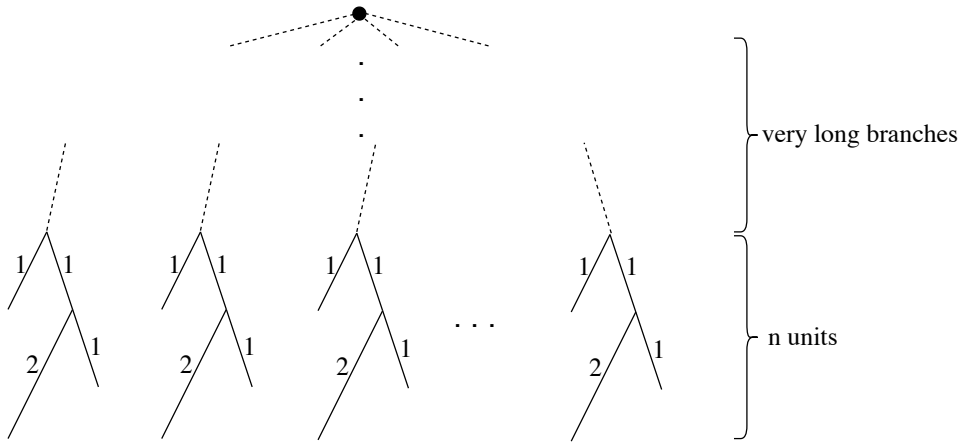


Figure 2.5. An example showing that number of minimal clusterings under a diameter threshold can be exponential of number of leaves. When the threshold is 3.5, each unit has to be split into two clusters, and there are thus three equally-optimal ways of splitting. The minimum number of clusters is therefore $2n$. The total number of distinct optimal solutions is 3^n, whereas there are $3n$ leaves.

two, Max-diameter has the advantage that its α threshold is easier to interpret. Finally, ASR-based selection of representative sequences outperformed consensus sequences, but we note that computing consensus sequences is much easier and faster.

Running time

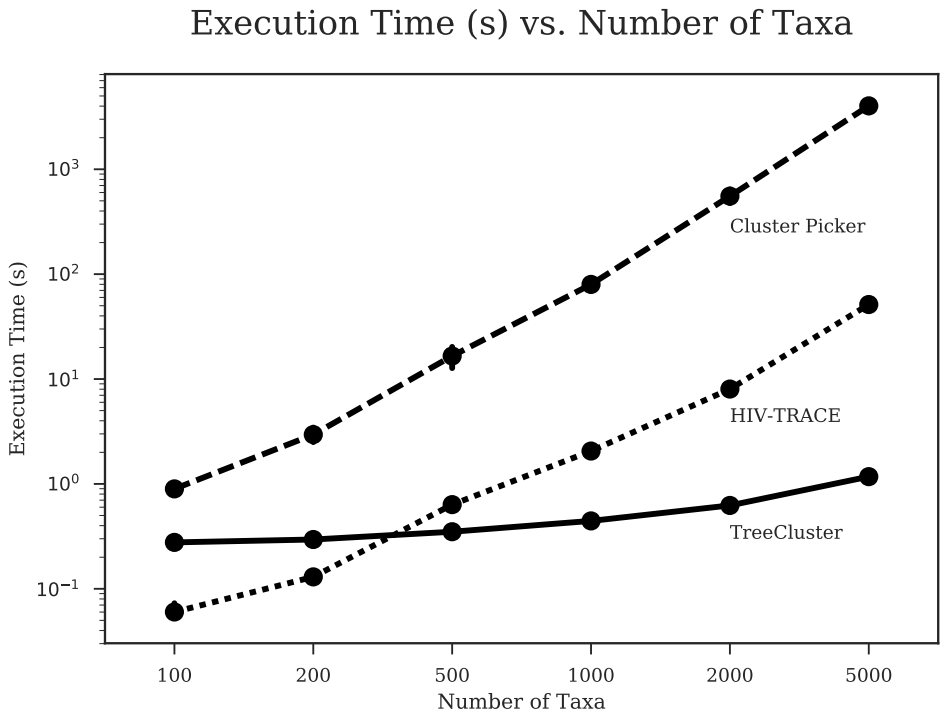


Figure 2.6. Execution times of Cluster Picker, HIV-TRACE, and TreeCluster in log-scale. Execution times (in seconds) are shown for each tool for various values of n sequences, with 10 replicates for each n. The full dataset was obtained by downloading all HIV-1 subtype B *pol* sequences (HXB2 coordinates 2,253 to 3,549) from the Los Alamos National Laboratory (LANL) database. All programs were run on a CentOS 5.8 machine with an Intel Xeon X7560 2.27 GHz CPU.

We focused on comparing the effectiveness of TreeCluster to other methods, but we note that its running time also compares favorably to other clustering methods (once the tree is inferred). For example, on a real HIV dataset, we ran HIV-TRACE, Cluster Picker, and TreeCluster for subsets of the data ranging from 100 to 5,000 sequences (Fig 2.6). On the largest set with 5,000 leaves, the running time of TreeCluster did not exceed 2 seconds. In contrast, the sequence-based HIV-Trace required close to a minute (which is still quite fast), but Cluster Picker needed more than an hour. Even on the Greengenes dataset with more than 200,000 leaves, TreeCluster performed clustering in only 30 seconds. The high speed of TreeCluster makes it possible to quickly scan through a set of α thresholds to study its impact on the outcomes of

downstream applications.

We note that these numbers do not include the time spent for inferring the tree, which should also be considered if the tree is not already available (note that in many applications a tree is inferred for other purposes and is readily available). For example, based on previous studies, MSA and tree inference on datasets with 10,000 sequences can take close to an hour using PASTA and 12 CPUs. Around a third of this time is spent on tree inference (e.g., see Fig 4 of [167]) and the rest is spent on the estimating alignment, which is also needed by most alternative clustering methods.

2.5 Conclusion

We introduced TreeCluster, a method that can cluster sequences at the tips of a phylogenetic tree using several optimization objective functions. We showed that our linear-time algorithms can be used in several downstream applications, including OTU clustering, HIV transmission clustering, and divide-and-conquer alignment. Using the tree to build the clusters increases their internal consistency and improves downstream analyses.

2.6 Acknowledgement

Chapter 2, in full, is a reprint of the material as it appears in "Balaban, M., Moshiri, N., Mai, U., Jia, X., & Mirarab, S. (08 2019). TreeCluster: Clustering biological sequences using phylogenetic trees. PLOS ONE, 14(8), 1–20". The dissertation author was the primary investigator and first author of this paper.

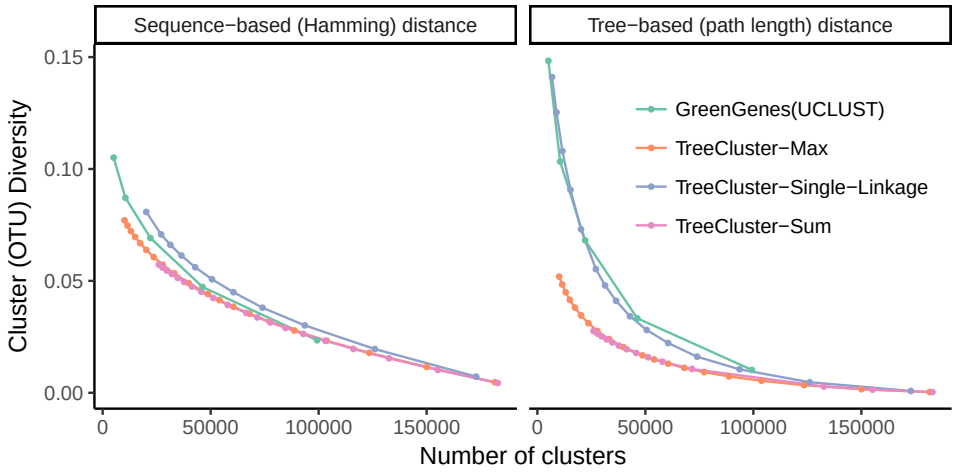


Figure 2.7. Comparison of various TreeCluster modes and Greengenes. Clustering quality of Greengenes and various TreeCluster modes, where quality is measured as average pairwise distance within a cluster (the lower the better). The horizontal axis shows the number of clusters for a given method and a threshold value. TreeCluster OTUs based on Max-diameter and Sum-length options outperform Single-linkage option as well as Greengenes OTUs. Computation of Hamming distance based cluster diversity for $\alpha \geq 0.7$ did not complete within 24 hours and had to be terminated.

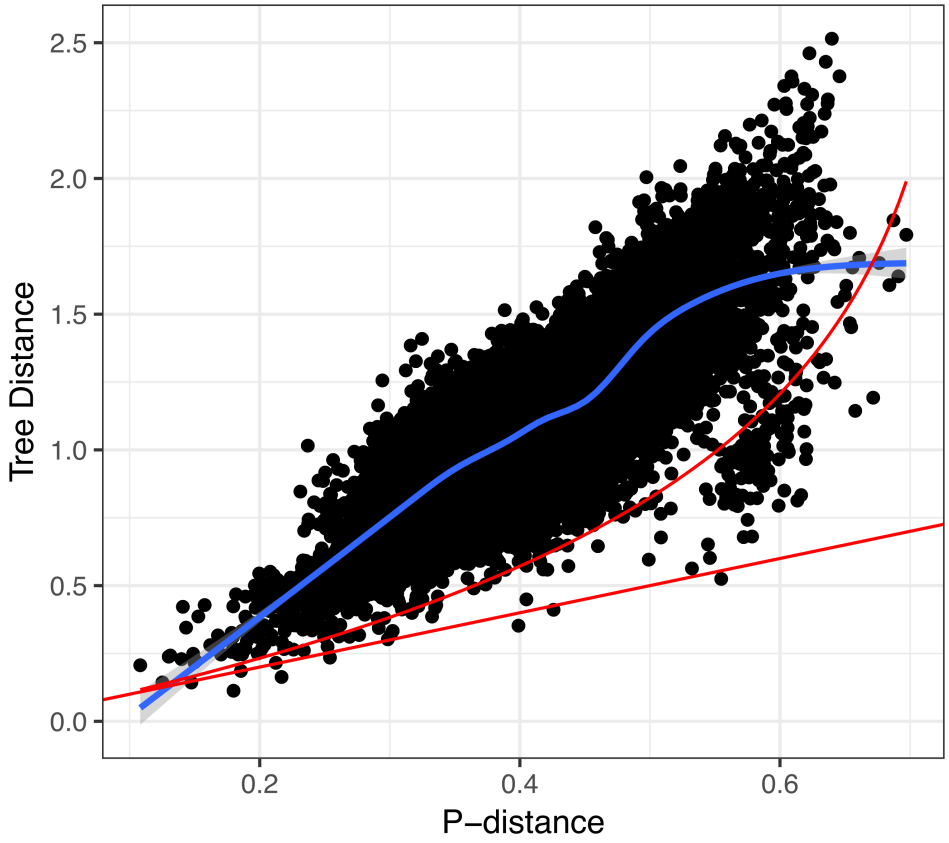


Figure 2.8. Tree distance versus Hamming distance. On 16S data, the relationship between tree distances and Hamming distances cannot be established using the Jukes-Cantor formula (red curve).

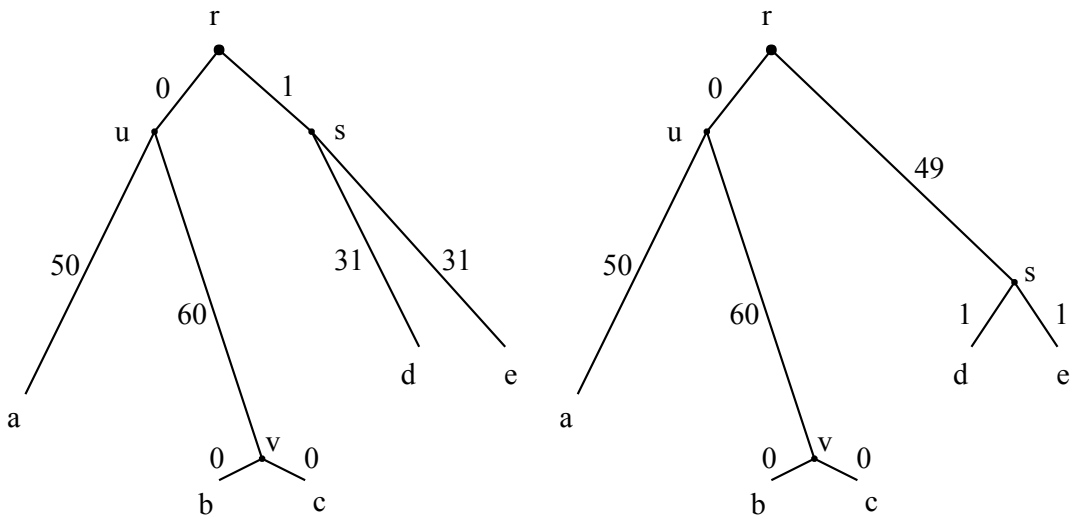


Figure 2.9. An example showing that Mean-diameter min-cut partitioning is not conforming locality when $\alpha = 72$, so it cannot be solved by a greedy algorithm analogous to Algorithm 1. When a greedy algorithm is at the stage where it processes u, it makes the decision for cutting its children edges (u,v) and (u,a) based on the information available at the subtree rooted by u. When $\alpha = 72$, (A) T_1 and (B) T_2 require different cut-sets ($\{(u,v)\}$ and $\{(u,a)\}$ respectively) for the optimal Mean-diameter partitioning despite the fact that the subtree rooted by u remains unchanged in T_1 and T_2.

Chapter 3

APPLES: Scalable Distance-based Phylogenetic Placement with or without Alignments

Placing a new species on an existing phylogeny has increasing relevance to several applications. Placement can be used to update phylogenies in a scalable fashion and can help identify unknown query samples using (meta-)barcoding, skimming, or metagenomic data. Maximum likelihood (ML) methods of phylogenetic placement exist, but these methods are not scalable to reference trees with many thousands of leaves, limiting their ability to enjoy benefits of dense taxon sampling in modern reference libraries. They also rely on *assembled* sequences for the reference set and aligned sequences for the query. Thus, ML methods cannot analyze datasets where the reference consists of unassembled reads, a scenario relevant to emerging applications of genome-skimming for sample identification. We introduce APPLES, a distance-based method for phylogenetic placement. Compared to ML, APPLES is an order of magnitude faster and more memory efficient, and unlike ML, it is able to place on large backbone trees (tested for up to 200,000 leaves). We show that using dense references improves accuracy substantially so that APPLES on dense trees is more accurate than ML on sparser trees, where it can run. Finally, APPLES can accurately identify samples without assembled reference or aligned queries using kmer-based distances, a scenario that ML cannot handle. APPLES is available publically at

github.com/balabanmetin/apples.

3.1 Introduction

Phylogenetic placement is the problem of finding the optimal position for a new *query* species on an existing *backbone* (or, reference) tree. Placement, as opposed to a *de novo* reconstruction of the full phylogeny, has two advantages. In some applications (discussed below), placement is all that is needed, and in terms of accuracy, it is as good as, and even better than [101], *de novo* reconstruction. Moreover, placement can be more scalable than *de novo* reconstruction when dealing with very large trees.

Earlier research on placement was motivated by scalability. For example, placement is used in greedy algorithms that start with an empty tree and add sequences sequentially [49, 65]. Each placement requires polynomial (often linear) time with respect to the size of the backbone, and thus, these greedy algorithms are scalable (often requiring quadratic time). Despite computational challenges [239], there has been much progress in the *de novo* reconstruction of ultra-large trees (e.g., thousands to millions of sequences) using both maximum likelihood (ML) [173, 187] and the distance-based [123] approaches. However, these large-scale reconstructions require significant resources. As new sequences continually become available, placement can be used to update existing trees without repeating previous computations on full dataset.

More recently, placement has found a new application in sample identification: given one or more *query* sequences of unknown origins, detect the identity of the (set of) organism(s) that could have generated that sequence. These identifications can be made easily using sequence matching tools such as BLAST [4] when the query either exactly matches or is very close to a sequence in the reference library. However, when the sequence is novel (i.e., has lowered similarity to known sequences in the reference), this *closest* match approach is not sufficiently accurate [111], leading some researchers to adopt a phylogenetic approach [174, 226]. Sample identification is essential to the study of mixed environmental samples, especially of the

microbiome, both using 16S profiling [75, 116] and metagenomics [237]. It is also relevant to barcoding [85] and meta-barcoding [30, 38] and quantification of biodiversity [67]. Driven by applications to microbiome profiling, placement tools like pplacer [153] and EPA(-ng) [17, 21] have been developed. Researchers have also developed methods for aligning query sequence [20, 164] and for downstream steps [152, 221]. These publications have made a strong case that for sample identification, placement is sufficient (i.e., *de novo* is not needed). Moreover, some studies [101] have shown that when dealing with fragmentary reads typically found in microbiome samples, placement can be *more* accurate than *de novo* construction and can lead to improved associations of microbiome with clinical information.

Existing phylogenetic placement methods have focused on the ML inference of the best placement – a successful approach, which nevertheless, suffers from two shortcomings. On the one hand, ML can only be applied when the reference species are *assembled* into full-length sequences (e.g., an entire gene) and are *aligned*; however, in new applications that we will describe, assembling (and hence aligning) the reference set is not possible. On the other hand, ML, while somewhat scalable, is still computationally demanding, especially in memory usage, and cannot place on backbone trees with many thousands of leaves. As the density of reference substantially impacts the accuracy and resolution of placement, this inability to use ultra-large trees as backbone also limits accuracy. This limitation has motivated alternative methods using local sensitive hashing [29] and divide-and-conquer [164].

Assembly-free and alignment-free sample identification using genome-skimming [52] can also benefit from phylogenetic placement. A genome-skim is a shotgun sample of the genome sequenced at low coverage (e.g., 1X) – so low that assembling the nuclear genome is not possible (though, mitochondrial or plastid genomes can often be assembled). Genome-skimming promises to replace traditional marker-based barcoding of biological samples [39] but limiting analyses to organelle genome can limit resolution. Moreover, mapping reads to reference genomes is also possible only for species that have been assembled, which is a small fraction of the biodiversity on Earth. Sarmashghi *et al.* [209] have recently shown that using

shared k-mers, the distance between two unassembled genome-skims with low coverage can be accurately estimated. This approach, unlike assembling organelle genomes, uses data from the entire nuclear genome and hence promises to provide a higher resolution (e.g., at species or sub-species levels) while keeping the low sequencing cost. However, ML and other methods that require assembled sequences cannot analyze genome-skims, where both the reference and the query species are unassembled genome-wide bags of reads.

Distance-based approaches to phylogenetics are well-studied, but no existing tool can perform distance-based placement of a query sequence on a given backbone. The distance-based approach promises to solve both shortcomings of ML methods. Distance-based methods are computationally efficient and do not require assemblies. They only need distances (however computed). Thus, they can take as input assembly-free estimates of genomic distance estimated from low coverage genome-skims using Skmer [209] or other alternatives [19, 63, 84, 100, 126, 179, 255]. While alignment-based phylogenetics has been traditionally more accurate than alignment-free methods when both methods are possible, in these new scenarios, only alignment-free methods are applicable.

Here, we introduce a new method for distance-based phylogenetic placement called APPLES (Accurate Phylogenetic Placement using LEast Squares). APPLES uses dynamic programming to find the optimal distance-based placement of a sequence with running time and memory usage that scale linearly with the size of the backbone tree. We test APPLES in simulations and on real data, both for alignment-free and aligned scenarios.

3.2 Description

3.2.1 Background and notations

Let an unrooted tree $T = (V, E)$ be a weighted connected acyclic undirected graph with leaves denoted by $\mathscr{L} = \{1 \cdots n\}$. We let T^* be the rooting of T on a leaf 1 obtained by directing all edges away from 1. For node $u \in V$, let $p(u)$ denote its parent, $c(u)$ denote its set of children,

$sib(u)$ denote its siblings, and $g(u)$ denote the set of leaves at or below u (i.e., those that have u on their path to the root), all with respect to T^*. Also let $l(u)$ denote the length of the edge $(p(u),u)$.

The tree T defines an $n \times n$ matrix where each entry $d_{ij}(T)$ corresponds to the path length between leaves i and j. We further generalize this definition so that $d_{uv}(T^*)$ indicates the length of the undirected path between any two nodes of T^* (when clear, we simply write d_{uv}). Given some input data, we can compute a matrix of all pairwise sequence distances Δ, where the entry δ_{ij} indicates the dissimilarity between species i and j. When the sequence distance δ_{ij} is computed using (the correct) phylogenetic model, it will be a noisy but statistically consistent estimate of the tree distance $d_{ij}(T)$ [66]. Given these "phylogenetically corrected" distances (e.g. $\frac{3}{4}\ln(1-\frac{4}{3}h)$ is the corrected hamming distance h under the Jukes-Cantor [105] model), we can define optimization problems to recover the tree that best fits the distances. A natural choice is minimizing the (weighted) least square difference between tree and sequence distances:

$$Q^*(T) = \sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(\delta_{ij} - d_{ij}(T))^2 \,. \tag{3.1}$$

Here, weights (e.g., w_{ij}) are used to reduce the impact of large distances (expected to have high variance). A general weighting schema can be defined as $w_{ij} = \delta_{ij}^{-k}$ for a *constant* value $k \in \mathbb{N}$. Standard choices of k include $k = 0$ for the ordinary least squares (OLS) method of Cavalli-Sforza and Edwards [35], $k = 1$ due to Beyer *et al.* [22] (BE), and $k = 2$ due to itch and Margoliash [68] (FM).

Finding $\arg\min_T Q^*(T)$ is NP-Complete [47]. However, decades of research has produced heuristics like neighbor-joining [207], alternative formulations like (balanced) minimum evolution [35, 49], and several effective tools for solving the problem heuristically (e.g., FastME by Lefort *et al.* [123], DAMBE by Xia [251], and Ninja by Wheeler [243]).

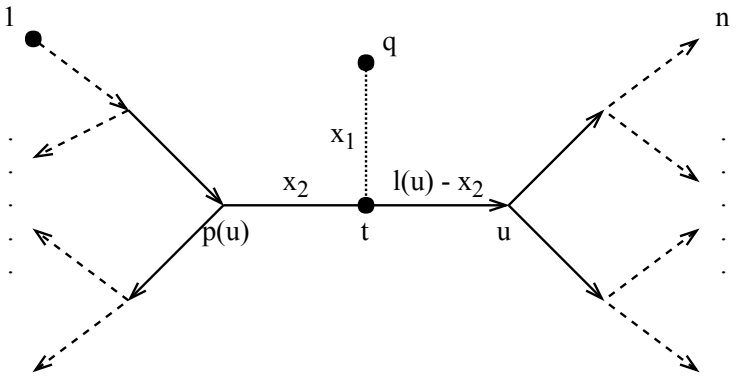


Figure 3.1. Any placement of q can be characterized as a tree $P(u,x_1,x_2)$, shown here. The backbone tree T^* is an arborescence on leaves $\mathscr{L} = \{1 \ldots n\}$, rooted at leaf 1. Query taxon q is added on the edge between u and $p(u)$, creating a node t. All placements on this edge are characterized by x_1, the length of the pendant branch, and x_2, the distance between t and $p(u)$.

3.2.2 Problem Statement

We let $P(u,x_1,x_2)$ be the tree obtained by adding a query taxon q on an edge $(p(u),u)$, creating three edges (t,q), $(p(u),t)$, and (t,u), with lengths x_1, x_2, and $l(u)-x_2$, respectively (Fig. 3.1). When clear, we simply write P and note that P *induces* T both in topology and branch length. We now define the problem.

- **Least Squares Phylogenetic Placement (LSPP):**

Input: A backbone tree T on $\mathscr{L}$, a query species q, and a vector Δ_{q*} with elements δ_{qi} giving sequence distances between q and every species $i \in \mathscr{L}$;

Output: The placement tree P that adds q on T and minimizes

$$Q(P) = \sum_{i=1}^{n} w_{qi}(\delta_{qi} - d_{qi}(P))^2 \tag{3.2}$$

3.2.3 Linear Time Algorithm

The number of possible placements of q is $2n-3$. Therefore, LSPP can be solved by simply iterating over all the topologies, optimizing the score for that branch, and returning the

placement with the minimum least square error. A naive algorithm can accomplish this in $\Theta(n^2)$ running time by optimizing Eq. 3.2 for each of the $2n-3$ branches. However, using dynamic programming, the optimal solution can be found in linear time.

Theorem 3. *The LSPP problem can be solved with $\Theta(n)$ running time and memory.*

The proof (Appendix B.1) follows easily from three lemmas that we next state. The algorithm starts with precomputing a fixed-size set of values for each nodes. For any node u and exponents $a \in \mathbb{Z}$ and $b \in \mathbb{N}^+$, let $S(a,b,u) = \sum_{i \in g(u)} \delta_{qi}^a d_{ui}^b$ and for $b=0$, let $S(a,0,u) = S'(a,u) = \sum_{i \in g(u)} \delta_{qi}^a$. Note that $S'(0,u) = |g(u)|$. Similarly, for $u \in V \setminus \{1\}$, let $R(a,b,u) = \sum_{i \notin g(u)} \delta_{qi}^a d_{p(u)i}^b$ for $b > 0$ and let $R(a,0,u) = R'(a,u) = \sum_{i \notin g(u)} \delta_{qi}^a$.

Lemma 1. *The set of all $S(a,b,u)$ and $R(a,b,u)$ values can be precomputed in $\Theta(n)$ time with two tree traversals using the dynamic programming given by:*

$$S(a,b,u) = \begin{cases} \delta_{qu}^a & u \in \mathscr{L} \setminus \{1\} \ \ \& \ \ b = 0 \\ 0 & u \in \mathscr{L} \setminus \{1\} \ \ \& \ \ b \neq 0 \\ \sum_{j=0}^{b} \sum_{v \in c(u)} l(v)^j \binom{b}{j} S(a, b-j, v) & u \notin \mathscr{L} \setminus \{1\} \end{cases} \tag{3.3}$$

$$R(a,b,u) = \begin{cases} \delta_{q1}^a & u = 1' = c(1) \ \ \& \ \ b = 0 \\ 0 & u = 1' = c(1) \ \ \& \ \ b \neq 0 \\ \sum_{j=0}^{b} \Big(l(p(u))^j \binom{b}{j} R(a, b-j, p(u)) & + \sum_{v \in sib(u)} l(v)^j \binom{b}{j} S(a, b-j, v) \Big) \quad u \notin \{1, 1'\} \end{cases} \tag{3.4}$$

Lemma 2. *Equation 3.2 can be rearranged (Appendix Eq. B.2) such that computing $Q(P)$ for a given $P = P(u, x_1, x_2)$ requires a constant time computation using $S(a,b,u)$ and $R(a,b,u)$ values for $-k \leq a \leq 2-k$ and $0 \leq b \leq 2$.*

Thus, after a linear time precomputation, we can compute the error for any given placement in constant time. It remains to show that for each node, the optimal placement on the branch above it (e.g., x_1 and x_2) can be computed in constant time.

Lemma 3. *For a fixed node $u \in V \setminus \{1\}$, if $(\hat{x}_1, \hat{x}_2) = \arg\min_{x_1,x_2} Q(P(u,x_1,x_2))$, then*

$$\begin{bmatrix} R'(-k,u)+S'(-k,u) & R'(-k,u)-S'(-k,u) \\ R'(-k,u)-S'(-k,u) & R'(-k,u)+S'(-k,u) \end{bmatrix} \cdot \begin{bmatrix} \hat{x}_1 \\ \hat{x}_2 \end{bmatrix} = \begin{bmatrix} R'(1-k,u)+S'(1-k,u)-l(u)S'(-k,u)-R(-k,1,u)-S(-k,1,u) \\ R'(1-k,u)-S'(1-k,u)+l(u)S'(-k,u)-R(-k,1,u)+S(-k,1,u) \end{bmatrix} \tag{3.5}$$

and hence $\hat{x}_1, \hat{x}_2$ can be computed in constant time.

Non-negative branch lengths.

The solution to Equation 3.5 does not necessarily conform to constraints $0 \leq x_1$ and $0 \leq x_2 \leq l(u)$. However, the following lemma (proof in Appendix B) allows us to easily impose the constraints by choosing optimal boundary points when unrestricted solutions fall outside boundaries.

Lemma 4. *With respect to variables x_1 and x_2, $Q(P(u,x_1,x_2))$ is a convex function.*

Minimum evolution

An alternative to directly using MLSE (Eq. 3.1) is the minimum evolution (ME) principle [35, 205]. Our algorithm can also optimize the ME criterion: after computing x_1 and x_2 by optimizing MLSE for each node u, we choose the placement with the minimum total branch length. This is equivalent to using $\arg\min_u x_1$, since the value of x_2 does not contribute to total branch length. Other solutions for ME placement exist [49], a topic we return to in the Discussion section.

Hybrid.

We have observed cases where ME is correct more often than MLSE, but when it is wrong, unlike MLSE, it has a relatively high error. This observation led us to design a hybrid approach. After computing x_1 and x_2 for all branches, we first select the top $log_2(n)$ edges with

minimum $Q(P(u,x_1,x_2))$ values (this requires $\Theta(n \log \log n)$ time). Among this set of edges, we place the query on the edge satisfying the ME criteria.

3.2.4 APPLES Software

We implemented the algorithm described above in a software called APPLES. APPLES uses Treeswift [169] for phylogenetic operations, and it generates the output in the jplace format [154]. APPLES can compute distances using vectorized numpy [178] operations but can also use input distance matrices (e.g. generated using FastME or Skmer). When computing distances internally, APPLES ignores positions that have a gap in at least one of the two sequences. By default, APPLES uses the JC69 model to compute phylogenetic distances [105] without Γ model of rate variation. It computes distances independently for all pairs, and not simultaneously as suggested by Tamura *et al.* [229].

By default, APPLES uses FM weighting, the MLSE selection criterion, enforcement of non-negative branch lengths, and JC69 distances. When not specified otherwise, these default parameters are used (the default setting is referred to as APPLES*).

3.3 Benchmark

3.3.1 Datasets

We benchmark accuracy and scalability of APPLES in two settings: sample identification using assembly-free genome-skims on real biological data and placement using aligned sequences on simulated data.

Real genome-skim datasets

Columbicola genome-skims.

We use a set of 61 genome-skims by Boyd *et al.* [27], including 45 known lice species (some represented multiple times) and 7 undescribed species. We generate lower coverage skims of 0.1Gb or 0.5Gb by randomly subsampling the reads from the sequence read archives (SRA)

provided by the original publication (NCBI BioProject PRJNA296666). We use BBTools [31] to filter subsampled reads for adapters and contaminants and remove duplicated reads. Since this dataset is not assembled, the coverage of the genome-skims is unknown; Skmer estimates the coverage to be between 0.2X and 1X for 0.1Gb samples (and 5 times that coverage with 0.5Gb).

Anopheles and Drosophila datasets.

We also use two insect datasets used by Sarmashghi *et al.* [209]: a dataset of 22 *Anopheles* and a dataset of 21 *Drosophila* genomes (Appendix B), both obtained from InsectBase [256]. For both datasets, genome-skims with 0.1Gb and 0.5Gb sequence were generated from the assemblies using the short-read simulator tool ART, with the read length $l = 100$ and default error profile. Since species have different genome sizes, with 0.1Gb data, our subsampled genome-skims range in coverage from 0.35X to 1X for Anopheles and from 0.4X to 0.8X for Drosophila.

More recently, Miller *et al.* [161] sequenced several Drosophila genomes, including 12 species shared with the InsectBase dataset. Sarmashghi *et al.* [209] subsampled the SRAs from this second project to 0.1Gb or 0.5Gb and, after filtering contaminants, obtained artificial genome-skims. We can use these genome-skims as query and the genome-skims from the InsectBase dataset as the backbone. Since the reference and query come from two projects, the query genome-skim can have a non-zero distance to the same species in the reference set, providing a realistic test of sample identification applications.

Backbone trees.

For all genome-skimming datasets, we inferred the backbone tree using FastME from the JC69 distance matrix computed from genome-skims using Skmer.

Simulated alignment-based datasets

GTR.

We use a 101-taxon dataset available from Mirarab *et al.* [163]. Sequences were simulated under the General Time Reversible (GTR) plus the Γ model of site rate heterogeneity using

INDELible [69] on gene trees that were simulated using SimPhy [149] under the coalescent model evolving on species trees generated under the Yule model. Note that the same model is used for inference under ML placement methods (i.e., no model misspecification). We took all 20 replicates of this dataset with mutation rates between 5×10^{-8} and 2×10^{-7}, and for each replicate, we selected five genes at random among many candidates that satisfy the condition that RF distance between the true tree and the tree inferred from the sequence is at most 20%. Thus, we have a total of 100 backbone trees. This dataset is the simplest test case where model violation or mis-alignment is not a concern.

RNASim.

Guo *et al.* [80] designed a complex model of RNA evolution that does not make usual i.i.d assumptions of sequence evolution. Instead, it uses models of energy of the secondary structure to simulate RNA evolution by a mutation-selection population genetics model. This model is based on an inhomogeneous stochastic process without a global substitution matrix. The model complexity of RNASim allows us to test both ML and APPLES under a substantially misspecified model. An RNASim dataset of one million sequences (with E.coli SSU rRNA used as the root), which consists of a multiple sequence alignment and true phylogeny, is available from Mirarab *et al.* [167]. We created several subsets of the full RNASim dataset.

i) Heterogeneous: We first randomly subsampled the full dataset to create 10 datasets of size $10,000$. Then, we chose the largest clade of size at most 250 from each replicate; this gives us 10 backbone trees of mean size 249.

ii) Varied diameter: To evaluate the impact of the evolutionary diameter (i.e., the highest distance between any two leaves in the backbone), we also created datasets with low, medium, and high diameters. We sampled the largest five clades of size at most 250 from each of the 10 replicates used for the heterogeneous dataset. Among these 50 clades, we picked the bottom, middle, and top five clades in diameter, which had diameter in $[0.3, 0.4]$ (mean: 0.36), $[0.5, 0.52]$ (mean: 0.51), and $[0.65, 1.07]$ (mean: 0.82), respectively.

iii) Varied size (RNASim-VS): We randomly subsampled the full dataset to create 5 replicates of datasets of size (n): 500, 1000, 5000, 10000, 50000, and 100000, and 1 replicate (due to size) of size 200000. For replicates that contain at least 5000 species, we removed sites that contain gaps in 95% or more of the sequences in the alignment.

iv) Query scalability (RNASim-QS): We first randomly subsampled the full dataset to create a dataset of size 500. Then for k =1 to 49,152 queries (choosing all $k = 3 \times 2^i, 0 \leq i \leq 14$) we created 5 replicates of k query sequences, again randomly subsampling from the full alignment with one million sequences.

v) Alignment Error (RNASim-AE): Mirarab *et al.* [167] used PASTA to estimate alignments on subsets of the RNASim dataset with up to 200,000 sequences. We use their reported alignment with 200,000 or 10,000 sequences (taking only replicate 1 in this case).

Backbone Alignment and Tree

Backbone alignments.

We present results both based on true backbone alignments (for all datasets) and PASTA-estimated alignments (for large datasets). The true alignments are known from the simulations. To test the accuracy in the presence of alignment error, we use the available PASTA backbone alignment for RNASim-AE dataset. The alignments have considerable error as measured by FastSP [162]: 11.5% and 12.7% SPFN, 10.9% and 11.7% SPFP, and only 165 and 848 fully correctly aligned sites (2.2% and 6.4%), respectively for 10,000 and 200,000 sequences. As before, here, we remove sites with more than 95% gaps in the estimated alignments.

Backbone trees.

For true alignments, we ran RAxML [220] using GTRGAMMA model for all datasets to estimate the topology of the backbone tree except for RNASim-AE and RNASim-VS dataset, where due to the size, we used FastTree-2 [187]. When estimated alignment is used (RNASim-AE dataset), we used the co-estimated tree output by PASTA, which is itself computed using FastTree-2. We always re-estimated branch lengths and model parameters on that fixed topology

using RAxML (switching to GTRCAT for $n \geq 10,000$) before running ML methods. For APPLES, we re-estimated branch lengths using FastTree-2 under the JC69 model to match the model used for estimating distances.

Alignment of queries.

For analyses with true backbone alignment, we use the true alignment of the queries to the backbone. In analyses with estimated backbone alignments, we align the query sequences to the estimated backbone alignment using SEPP [164], which is a divide-and-conquer method that internally uses HMMER [56, 57], with alignment subset size set to 10% of the full set (default setting). We use the resulting extended alignment, after masking unaligned sites, to run both APPLES and EPA-ng, in both cases, placing on the full backbone tree. We also report results using default SEPP (which runs pplacer internally); however, here, due to limitations of pplacer, we use the default setting of SEPP, which set the placement subset size to 10% of the full set.

3.3.2 Methods Compared

For aligned data, we compare APPLES to two ML methods: pplacer [153] and EPA-ng [17]. Matsen *et al.* found pplacer to be substantially faster than EPA [20] while their accuracy was similar. EPA-ng improves the scalability of EPA; thus, we compare to EPA-ng in analyses that concerned scalability (e.g., RNASim-VS). We run pplacer and EPA-ng in their default mode using GTR+Γ model and use their best hit (ML placement). We also compare with a simple method referred to as CLOSEST that places the query as the sister to the species with the minimum distance to it. CLOSEST is meant to emulate the use of BLAST (if it could be used). For the assembly-free setting, existing phylogenetic placement methods cannot be used, and we compared only against CLOSEST.

To run APPLES on assembly-free datasets, we first compute genomic distances using Skmer [209]. We then correct these distances using the JC69 model, without Γ model of rate variation. For APPLES on alignment-based analyses, we let APPLES compute distances for

JC69. We also use FastME (see Supplementary methods) to compute distances according to four more models: JC69+Γ [104], the six-parameter TN93 [228] model, TN93+Γ [238], and the 12-parameter general Markov model [141]. Pairing Gamma with GTR is theoretically possible in the absence of noise; however, the method can run into problems on real data [238]. Thus, we do not include a GTR model directly. Instead, we use the log-det approach that can handle the most general (12-parameter) Markov model [141]; however, log-det cannot account for rate across sites heterogeneity [238]. In JC69+Γ and TN93+Γ models, we used the α parameter computed by RAxML [220] run on the backbone alignment and given the backbone tree.

3.3.3 Evaluation Procedure

To evaluate the accuracy, we use a leave-one-out strategy. We remove each leaf i from the backbone tree T and place it back on this $T \setminus i$ tree to obtain the placement tree P. On the RNAsim-VS dataset, due to its large size, we only removed and added back 200 randomly chosen leaves per replicate. On the RNAsim-AE dataset, we remove 200 queries from the backbone at the same time (leave-many-out). Finally, for RNAsim-QS, we place k queries *in one run*, allowing the methods to benefit from optimizations designed for multiple queries, but note that queries are not selected from the backbone tree but are instead selected from the full dataset. In all cases, placement of queries is with respect to the backbone and not other queries.

Delta error.

We measure the accuracy of a placement tree P of a single query q on a backbone tree T on leafset $\mathscr{L}$ with respect to the true tree T^* on $\mathscr{L} \cup \{q\}$ using delta error:

$$\Delta e(P) = |B(T^*) \setminus B(P)| - |B(T^* \restriction_{\mathscr{L}}) \setminus B(T)| \,. \tag{3.6}$$

where $B(.)$ is the set of bipartitions of a tree and $T^* \restriction_{\mathscr{L}}$ is the true tree restricted to $\mathscr{L}$. Note that $\Delta e(P) \geq 0$ because adding q cannot decrease the number of missing branches in T. We report delta error averaged over all queries (denoted as Δe). Backbone tree is estimated from the same

Table 3.1. Assembly-free placement of genome-skims. We show the percentage of placements into optimal position (those that do not increase Δe), average delta error (Δe), and maximum delta error (e_{max}) for APPLES, assignment to the CLOSEST species, and the placement to the position in the backbone (DE-NOVO) over the 61 (a), 22 (b), and 21 (c) placements. Results are shown for genome skims with 0.1Gbp of reads. Delta error is the increase in the missing branches between the true tree (or the gold standard for biological data) and the backbone tree after placing each query.

	(a) Columbicola			(b) Anopheles			(c) Drosophila		
	%	Δe	e_{max}	%	Δe	e_{max}	%	Δe	e_{max}
APPLES*	97	0.03	1	95	0.05	1	71	0.29	1
APPLES-ME	84	0.28	5	95	0.05	1	67	0.42	2
APPLES-HYBRID	87	0.16	2	95	0.05	1	67	0.33	1
CLOSEST	54	1.15	7	91	0.09	1	57	0.62	3
DE-NOVO	98	0.02	1	95	0.05	1	71	0.29	1

data used in distance calculation, whereas true tree is either the ground truth or the gold standard that approximates the most to the truth. In leave-one-out experiments, placing q to the same location as the backbone before leaving it out can still have a non-zero delta error because the backbone tree is not the true tree. We refer to the placement of a leaf into its position in the backbone tree as the *de novo* placement. In leave-many-out experiments, we measure delta error of each query separately (not the delta error of the combination of all queries). On biological data, where the true tree is unknown, we instead use a published phylogenetic tree as the gold standard (Fig. 3.6). For Drosophila and Anopheles, we use the tree available from the Open Tree Of Life [90] and for Columbicola, we use the ML concatenation tree published by Boyd *et al.*.

3.4 Benchmark Results

3.4.1 Assembly-free Placement of Genome-skims

On our three biological genome-skim datasets, APPLES* successfully places the queries on the optimal position in most cases (97%, 95%, and 71% for Columbicola, Anopheles, and Drosophila, respectively) and is never off from the optimal position by more than one branch. Other versions of APPLES are less accurate than APPLES*; e.g., APPLES with ME can have

up to five wrong branches (Table 3.1). On genome-skims, where assembly and alignment are not possible, existing placement tools cannot be used, and the only alternative is the CLOSEST method (emulating BLAST if assembly was possible). CLOSEST finds the optimal placement only in 54% and 57% of times for Columbicola and Drosophila; moreover, it can be off from the best placement by up to seven branches for the Columbicola dataset. On the Anopheles dataset, where the gold standard tree is unresolved (Fig. 3.6), all methods perform similarly.

APPLES* is less accurate on the Drosophila dataset than other datasets. However, here, simply placing each query on its position in the backbone tree would lead to identical results (Table 3.1). Thus, placements by APPLES* are as good as the *de novo* construction, meaning that errors of APPLES* are entirely due to the differences between our backbone tree and the gold standard tree. Moreover, these errors are not due to low coverage; increasing the genome-skim size 5x (to 0.5Gb) does not decrease error (Appendix Table B.4).

On Drosophila dataset, we next tested a more realistic sample identification scenario using the 12 genome-skims from the separate study (and thus, non-zero distance to the corresponding species in the backbone tree). As desired, APPLES* places all of 12 queries from the second study as sister to the corresponding species in the reference dataset.

3.4.2 Alignment-based Placement

We first compare the accuracy and scalability of APPLES* to ML methods and then compare various settings of APPLES. For ML, we use pplacer (shown everywhere) and EPA-ng (shown only when we study scalability and work on large backbones).

Comparison to Maximum Likelihood (ML) without Alignment Error

GTR dataset.

On this dataset, where it faces no model misspecification, pplacer has high accuracy. It finds the best placement in 84% of cases and is off by one edge in 15% (Fig. 3.2a); its mean delta error (Δe) is only 0.17 edges. APPLES* is also accurate, finding the best placement in 78% of cases and resulting in the mean Δe =0.28 edges. Thus, even though pplacer uses ML and faces

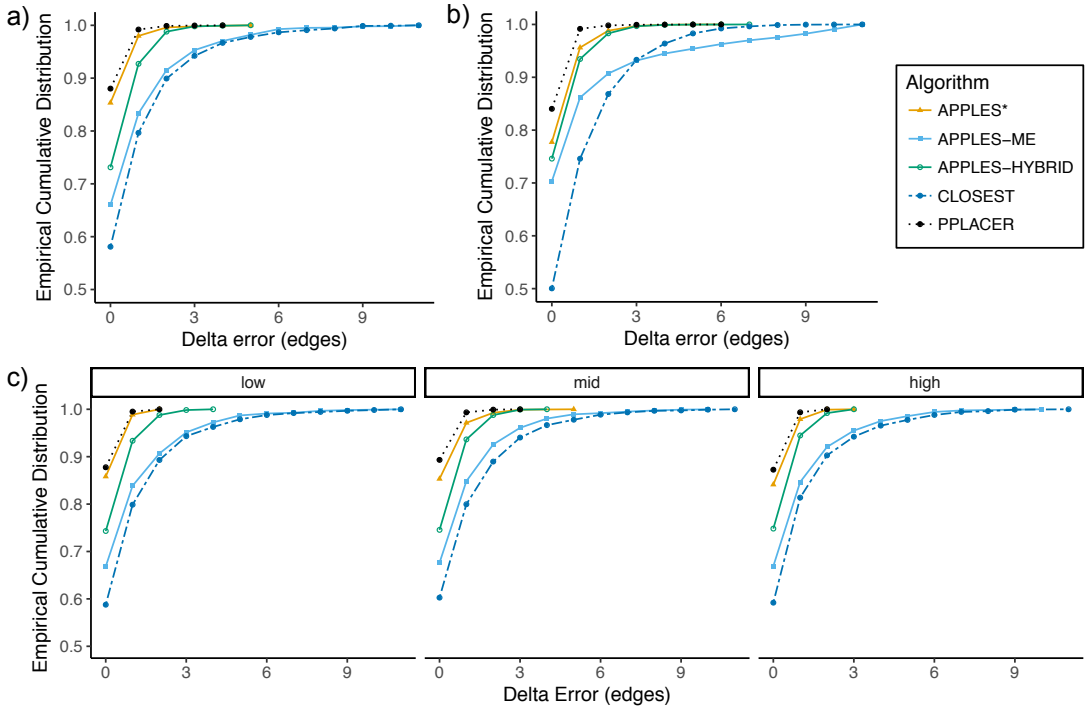


Figure 3.2. Accuracy on simulated data. We show empirical cumulative distribution of the delta error, defined as the increase in the number of missing branches in the tree compared to the true tree after placement. We compare pplacer (dotted), CLOSEST match (dashed), and APPLES with FM weighting and JC69 distances and MLSE (APPLES*), ME, or Hybrid optimization. (a) GTR dataset. (b) RNASim-Heterogeneous. (c) RNASim-varied diameter, shown in boxes: low, medium (mid), or high. Distributions are over 10,000 (a), 2450 (b), and 3675 (c) points.

no model misspecification and APPLES* uses distances based on a simpler model, the accuracy of the two methods is within 0.1 edges on average. In contrast, CLOSEST has poor accuracy and is correct only 50% of the times, with the mean Δe of 1.0 edge.

Model misspecification.

On the small RNASim data with subsampled clades of $\approx$ 250 species), both APPLES* and pplacer face model misspecification. Here, the accuracy of APPLES* is very close to ML using pplacer. On the heterogeneous subset (Fig. 3.2b and Table 3.2), pplacer and APPLES* find the best placement in 88% and 85% of cases and have a mean delta error of 0.13 and 0.17 edges, respectively. Both methods are much more accurate than CLOSEST, which has a delta error of 0.87 edges on average.

Table 3.2. The delta error for APPLES*, CLOSEST match, and pplacer on the RNASim-varied diameter dataset (low, medium, or high) and the RNA-heterogeneous dataset. Measurements are shown over 1250 placements for each diameter size category, corresponding to 5 backbone trees and 250 placements per replicate.

	Low			Medium			High			Heterogeneous		
	%	Δe	e_{max}	%	Δe	e_{max}	%	Δe	e_{max}	%	Δe	e_{max}
APPLES*	86	0.15	2	85	0.18	5	84	0.18	3	85	0.17	5
CLOSEST	59	0.88	13	60	0.88	13	60	0.85	14	60	0.87	14
pplacer	88	0.13	2	89	0.11	3	87	0.13	3	88	0.13	3

Table 3.3. Percentage of correct placements (shown as %) and the delta error (Δe) on the RNASim datasets with various backbone size (n). % and Δe is over 1000 placements (except $n = 200,000$, which is over 200 placements). Running pplacer and EPA-ng was not possible (n.p) for trees with at least $10,000$ leaves and failed in some cases (number of fails shown) for $5,000$ leaves.

	n = 500		n = 1,000		n = 5,000		n = 10000		n = 100,000		n = 200,000	
	%	Δe	%	Δe	%	Δe	%	Δe	%	Δe	%	Δe
APPLES-2	74	0.33	76	0.31	78	0.35	83	0.25	89	0.13	92	0.11
APPLES*	75	0.32	71	0.43	77	0.37	79	0.33	84	0.25	87	0.25
CLOSEST	52	1.16	53	1.18	54	1.15	59	0.90	61	0.69	63	0.70
EPA-ng	73	0.33	73	0.31	78	0.24	79	0.22	n.p	n.p	n.p	n.p
pplacer	80	0.23	81	0.20	n.p	n.p	n.p	n.p	n.p	n.p	n.p	n.p

Impact of diameter.

When we control the tree diameter, APPLES* and pplacer remain very close in accuracy (Fig. 3.2c). The changes in error are small and not monotonic as the diameters change (Table 3.2). The accuracies of the two methods at low and high diameters are similar. The two methods are most divergent in the medium diameter case, where pplacer has its lowest error ($\Delta e = 0.11$) and APPLES* has its highest error ($\Delta e = 0.18$).

To summarize results on small RNASim dataset with model misspecification, although APPLES* uses a parameter-free model, its accuracy is extremely close to ML using pplacer with the GTR+Γ model.

Impact of taxon sampling.

The real advantage of APPLES* over pplacer becomes clear for placing on larger backbone trees (Fig. 3.3 and Table 3.3). For backbone sizes of 500 and 1000, pplacer continues to

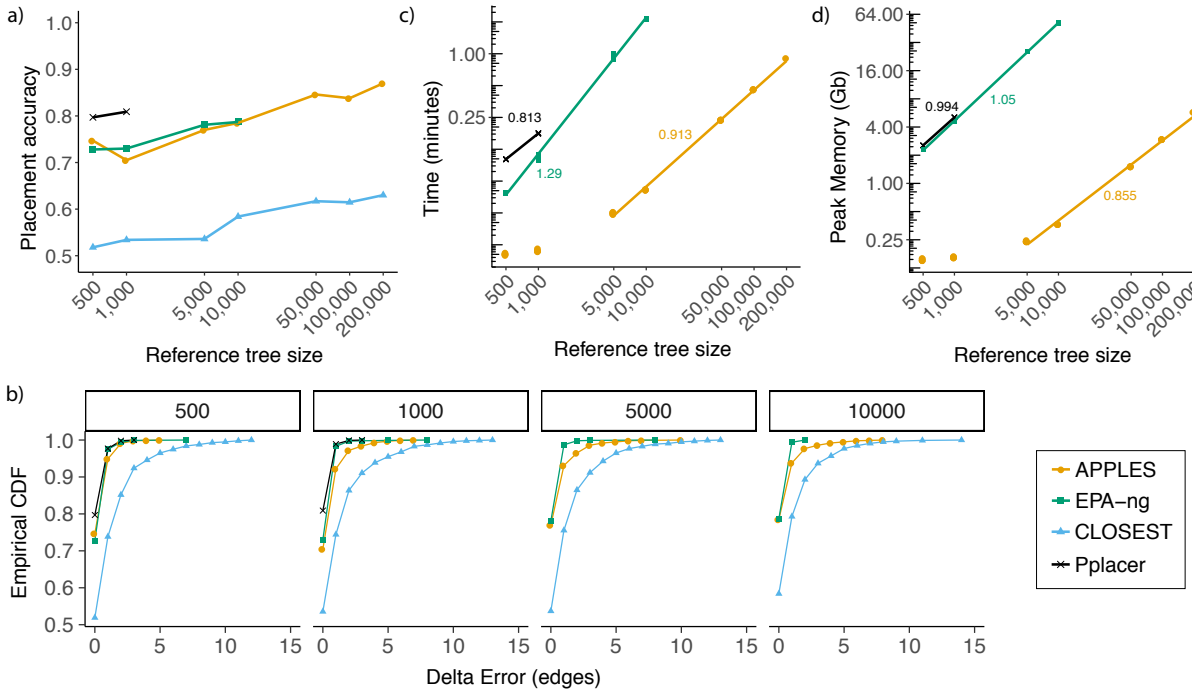


Figure 3.3. Results on RNASim-VS. (a) Placement accuracy with taxon sampling ranging from 500 to 200,000. (b) The empirical cumulative distribution of the delta error, shown for $500 \leq n \leq 10000$ where EPA-ng can run. (c,d) Running time and peak memory usage of placement methods for a single placement. Lines are fitted in the log-log scale and their slope (indicated on the figure) empirically estimates the polynomial degree of the asymptotic growth ($t = n^a \Rightarrow \log t = a \log n$). APPLES lines are fitted to $\geq 5,000$ points because the first two values are small and irrelevant to asymptotic behavior. All calculations are on 8-core, 2.6GHz Intel Xeon CPUs (Sandy Bridge) with 64GB of memory, with each query placed independently and given 1 CPU core and the entire memory.

be slightly more accurate than APPLES* (mean Δe of pplacer is better than APPLES* by 0.09 and 0.23 edges, respectively). However, with backbones of 5000 leaves, pplacer fails to run on 449/1000 cases, producing infinity likelihood (perhaps due to numerical issues) and has 41 times higher error than APPLES* on the rest (Fig. 3.7). Since pplacer could not scale to 5,000 leaves, we also test the recent method, EPA-ng [17]. Given the 64GB of memory available on our machine, EPA-ng is able to run on datasets with up to $10,000$ leaves. EPA-ng finds the correct placement less often than pplacer but is close to APPLES* (Fig. 3.3a). However, when it has error, it tends to have a somewhat lower distance to the correct placement, making its mean error slightly better than APPLES* (Fig. 3.3b and Table 3.3).

For backbone trees with more than $10,000$ leaves, pplacer and EPA-ng are not able to run given computational resources at hand, and CLOSEST is not very accurate (finding the best placement in only 59% of cases). However, APPLES* continues to be accurate for all backbone sizes. As the backbone size increases, the taxon sampling of the tree is improving (recall that these trees are all random subsets of the same tree). With denser backbone trees, APPLES* has increased accuracy despite placing on larger trees (Fig. 3.3a, Table 3.3). For example, using a backbone tree of $200,000$ leaves, APPLES* is able to find the best placement of query sequences in 87% of cases, which is better than the accuracy of either APPLES* or ML tools on any backbone size. Thus, an increased taxon sampling helps accuracy, but ML tools are limited in the size of the tree they can handle given relatively powerful machines (e.g., 64GB of memory).

Running time and memory.

As the backbone size increases, the running times of pplacer and APPLES grow close to linearly with the size of the backbone tree, n, whereas running time of EPA-ng seems to grow with $n^{1.3}$ (Fig. 3.3c). APPLES is on average 13 times faster than pplacer and 7.5 times faster than EPA-ng on backbone trees with 1000 leaves, and is 41 times faster than EPA-ng with 10,000-taxon backbones.

The memory consumption of all methods increases close to linearly as n increases, but

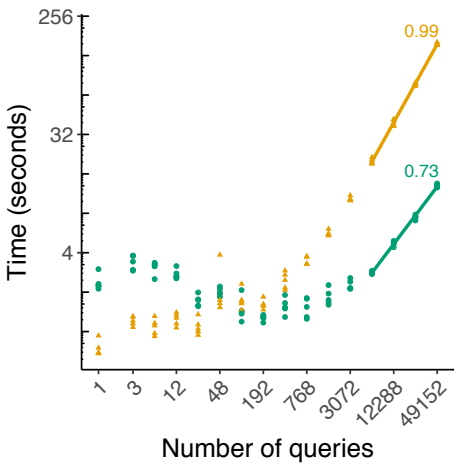


Figure 3.4. Scalability with respect to the number of queries. We show wall-clock running time with respect to increased numbers of queries (k) in one execution given 28 CPU cores and 28 threads on a Intel Xeon E5 CPU with 64 GB of memory. We fit a line to running times in log-log scale only for $k \geq 6144$ because otherwise, the preprocessing time would distort estimates (note: $t = n^a + b \not\Rightarrow \log t = a \log n + \varepsilon$ except in approximation if $b \ll t$).

APPLES requires dramatically less memory (Fig. 3.3d). For example, for placing on a backbone with 10,000 leaves, EPA-ng requires 51GB of memory, whereas APPLES requires only 0.4GB. APPLES easily scales to a backbone of $200,000$ sequences, running in only 1 minute and using 6GB of memory per query (including all precomputations in the dynamic programming). These numbers also include the time and memory needed to compute the distance between the query sequence and all the backbone sequences.

We next test the scalability of APPLES* and EPA-ng with respect to the number of queries, k, in one run given 28 CPU cores and 28 threads on RNASim-QS dataset. Both methods spend time on preprocessing steps that will be amortized over a large number of queries. The running time of APPLES*, as expected, increases linearly with $k > 200$ and grow more slowly for $k < 200$ (due to the preprocessing) (Fig. 3.4). The patterns of running time of EPA-ng are surprising. Increasing k from 1 to 768 *decreases* the running time instead of increasing it. While the exact reasons for the reductions are not clear to us, we note that EPA-ng developers have taken great care to implement various optimizations, especially for utilizing multiple cores. Only with $k > 768$ we start to see increasing running times for EPA-ng. With $k > 3072$, EPA-ng

seems to grow in running with $k^{0.73}$; however, we suspect further increasing k would increase the exponent (asymptotic running time cannot theoretically be less than $O(k)$ as all queries need to be read). Aside from the fluctuation due to small sample size for small k values, the number of queries do not seem to affect the accuracy for either method as expected since both methods treat queries independently (Fig. 3.8).

Comparing parameters of APPLES.

Comparing five models of sequence evolution that can be used with APPLES, we see similar patterns of accuracy across all models despite their varying complexity, ranging from 0 to 12 parameters (Fig. 3.9). Since the JC69 model is parameter-free and results in similar accuracy to others, we opted to use it as the default. Next, we ask whether imposing the constraint to disallow negative branch lengths improves the accuracy. The answer depends on the optimization strategy. Forcing non-negative lengths marginally increases the accuracy for MLSE but dramatically reduces the accuracy for ME (Fig. 3.10ab). Thus, we always impose non-negative constraints on MLSE but never for ME. Likewise, our Hybrid method includes the constraint for the first MLSE step but not for the following ME step (Fig. 3.10c).

The next parameter to choose is the weighting scheme. Among the three methods available in APPLES, the best accuracy belongs to the FM scheme closely followed by the BE (Fig. 3.11). The OLS scheme, which does not penalize long distances, performs substantially worse than FM and BE. Thus, the most aggressive form of weighting (FM) results in the best accuracy. Fixing the weighting scheme to FM and comparing the three optimization strategies (MLSE, ME, and Hybrid), the MLSE approach has the best accuracy (Fig. 3.2), finding the correct placement 84% of the time (mean error: 0.18), and ME has the lowest accuracy, finding the best placement in only 67% of cases (mean error: 0.70). The Hybrid approach is between the two (mean error: 0.34) and fails to outperform MLSE on this dataset. However, when we restrict the RNASim backbone trees to only 20 leaves, we observe that Hybrid can have the best accuracy (Fig. 3.12).

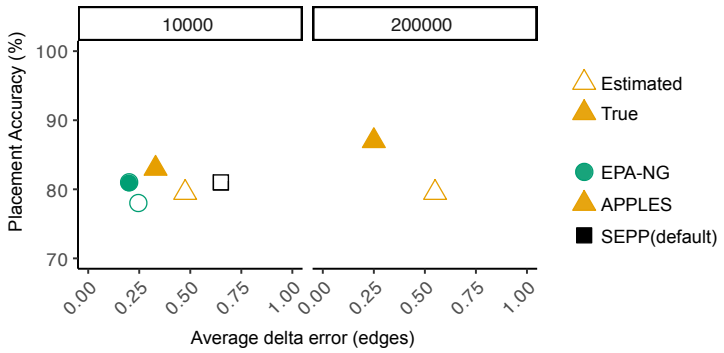


Figure 3.5. Impact of alignment error on placement accuracy. Left and right panels show placement accuracy and mean delta error of leave-many-out experiments for backbone size of $10,000$ and $200,000$ (200 queries each).

3.4.3 Comparison to ML with Alignment Error

We next test the impact of alignment errors. On RNASim-AE dataset with $n = 200,000$ sequences, we observe 80% placement accuracy using SEPP+APPLES* (Fig. 3.5), which is a statistically significant ($p = 0.005$ according to McNemar's test) but relatively modest 7% reduction compared to placing using the true alignments. On the $n = 10,000$ backbone, placement accuracy drops from 83% on the true alignment to 80% using SEPP+APPLES*, and from 81% to 78% using SEPP+EPA-ng; neither change is statistically significant ($p = 0.14$ and $p = 0.08$, respectively). Despite the relatively small drops on placement accuracy, the impact of alignment error on delta error can be more pronounced (Fig. 3.5). The mean delta error goes up from 0.33 to 0.48 for $n = 10,000$, which is statistically significant ($p = 0.018$ according to a one-sided paired t-test). For $n = 200,000$, the error also increases significantly ($p = 0.004$) from 0.25 to 0.55 edges.

Recall that SEPP+APPLES* eliminates the need for decomposing the backbone tree into smaller placement subtrees, as default SEPP must do to deal with memory requirements of pplacer (which it internally uses). Comparison of default SEPP and SEPP+APPLES* shows that incorporating APPLES* inside SEPP reduces the mean delta error from 0.65 to 0.48 while it hurts placement accuracy from 81% to 80%; however, neither change is statistically significant

(p=0.15, one-sided paired t-test, and p = 0.66, McNemar's test, respectively). To summarize, highly accurate placement is possible even with estimated alignments with backbones of size up to 200,000 using the divide-and-conquer methods PASTA and SEPP for estimating alignments of the backbone and query, respectively.

3.5 Discussion

We introduced APPLES: a new method for adding query species onto large backbone trees using both unassembled genome-skims and aligned data. We now provide further observations on our results and on distance-based placement.

3.5.1 Further observations on the results

The accuracy of APPLES was very close to ML in most settings where we could run ML; the accuracy advantages of ML were particularly small for the RNASim dataset where both methods face model misspecification. As expected by the substantial evidence from the literature [89, 266], improved taxon sampling increased the accuracy of placement. Thus, overall, the best accuracy on RNASim dataset was obtained by APPLES* run on the full reference dataset, further motivating its use when large backbones are available. Despite many strides made in terms of scalability by the new method EPA-ng, ML methods still have to restrict their backbone to at most several thousand species given reasonable amounts of memory (up to 64GB in our case). We also note that it is possible to follow the APPLES* placement with a round of ML placement on smaller trees, but the small differences in accuracy of ML and APPLES* on smaller trees did not give us compelling reasons to try such hybrid approaches.

APPLES was an order of magnitude or more faster and less memory-hungry than ML tools (pplacer and EPA-ng) for single query runs. However, for placing large numbers of queries (e.g., as found in metagenomic datasets) on a relatively small sized backbone ($n = 500$), EPA-ng had an advantage since it is specifically designed to tackle scalability of multiple queries.

Advantages in memory consumption and scalability to large backbone trees remain for

APPLES* regardless of the number of queries. The python APPLES code is not optimized nearly as much as EPA-ng and can also benefit from some of the heuristic techniques used by EPA-ng. We plan for the future versions of the code to focus on improved scalability as the number of queries increases.

By incorporating APPLES inside SEPP, we were able to create a method that can do both alignment and placement on very large backbones with reasonable computational requirements and high accuracy. We observed relatively small reductions in accuracy as a result of alignment error, a pattern that we find remarkable given the size of the tree and the amount of error in the estimated alignment. The default SEPP method deals with large backbone trees by dividing the backbone tree into "placement" subtrees and choosing which subtree to place on using bit-scores produced by HMMs trained on subsets of sequences. The original paper had shown that the best accuracy is obtained with the largest possible backbone subtrees given computational limitations. APPLES now enables us to eliminate the need for decomposition into placement subsets, and in doing so, reduces placement error.

In our analyses, we observed no advantage in using models more complex than JC69+Γ for distance calculation inside APPLES. However, these results may be due to our pairwise estimation of model parameters (e.g., base compositions). More complex models may perform better if we instead estimate model parameters on the backbone alignment/tree and reuse the parameters for queries (or simultaneously among all queries and the reference sequences). Simultaneous estimation of distances has many advantages over using independent distances for the *de novo* case [229, 250]; these results give us hope that using simultaneous distances inside APPLES can further improve its accuracy.

Branch lengths of our backbone trees were computed using the same distance model as the one used for computing the distance of the query to backbone species. Using consistent models for the query and for the backbone branch lengths is essential for obtaining good accuracy (see Fig. 3.13 for evidence). Thus, in addition to having a large backbone tree at hand, we need to ensure that branch lengths are computed using the right model. Fortunately, FastTree-2 can

compute both topologies and branch lengths on large trees in a scalable fashion, without a need for quadratic time/memory computation of distance matrices [187].

3.5.2 Observations on distance-based placement

Phylogenetic insertion using the ME criterion has been previously studied for the purpose of creating an algorithm for greedy minimum evolution (GME). Desper and Gascuel [49] have designed a method that given the tree T can update it to get a tree with $n+1$ leaves in $\Theta(n)$ after precomputation of a data-structure that gives the average *sequence* distances between all adjacent clusters in T. The formulation by Desper and Gascuel has a subtle but consequential difference from our ME placement. Their algorithm does not compute branch lenghts for inserted sequence (e.g., x_1 and x_2). It is able to compute the optimal placement topology *without* knowing branch lengths of the backbone tree. Instead, it relies on pairwise distances among backbone *sequences* (Δ), which are precomputed and saved in the data-structure mentioned before. In the context of the greedy algorithm for tree inference, in each iteration, the data structure can be updated in $\Theta(n)$, which does not impact the overall running time of the algorithm. However, if we were to start with a tree of n leaves, computing this structure from scratch would still require $\Theta(n^2)$. Thus, computing the placement for a new query would need quadratic time, unless if the $\Theta(n^2)$ precomputation is allowed to be amortized over $\Omega(n)$ queries. Our formulation, in contrast, uses branch lengths of the backbone tree (which is assumed fixed) and thus never uses pairwise distances among the backbone sequences. Thus, using tree distances is what allows us to develop a linear time algorithm. Finally, we note that in our experimental analyses, we were not able to test the distance-based algorithm of Desper *et al.* [49] because it is available only as part of a the greedy algorithm inside FastME but is not available as a stand-alone feature to place on a given tree.

We emphasize that our results in assembly-free tests do not advocate for the use of assembly-free methods when assemblies are available. Moreover, we have no evidence that assembly-free methods are effective in inferring deep branches of a phylogeny. Instead, our

results show that assembly-free phylogenetic placement is effective in sample identification where assembly is not possible due to low coverage. In assembly-free analyses, we used Skmer to get distances because alternative alignment-free methods of estimating distance generally either require assemblies [84, 125, 126] or higher coverage than Skmer [19, 179, 255]; however, combining APPLES with other alignment-free methods can be attempted in future (finding the best way of computing distances without assemblies was not our focus). Moreover, the Skmer paper has described a trick that can be used to compute log-det distances from genome-skims. Future studies should test whether using that trick and using GTR instead of JC69 improves accuracy.

3.6 Availability

The APPLES software is publicly available in open-source from github.com/balabanmetin/apples.

3.7 Acknowledgment

Chapter 3, in full, is a reprint of the material as it appears in "Balaban, M., Sarmashghi, S., & Mirarab, S. (09 2019). APPLES: Scalable Distance-Based Phylogenetic Placement with or without Alignments. Systematic Biology, 69(3), 566–578". The dissertation author was the primary investigator and first author of this paper.

Figure 3.6. The reference biological trees obtained from Open Tree of Life (Drosophila and Anopheles) and from Boyd *et al.* (Columbicola).

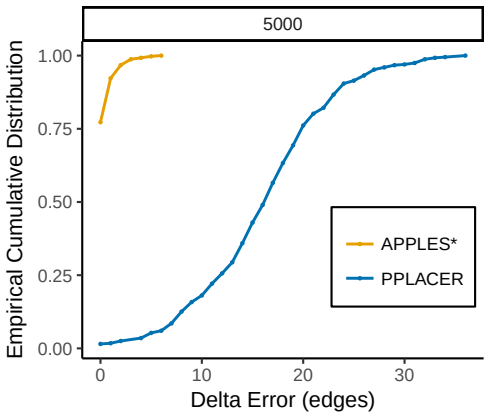


Figure 3.7. APPLES versus pplacer on 5,000 backbone trees. The empirical cumulative distribution of the delta error is shown. We compare pplacer and APPLES* on RNASim-VS dataset with 5000 leaves. Distributions is over 551 cases where pplacer could run for the panel.

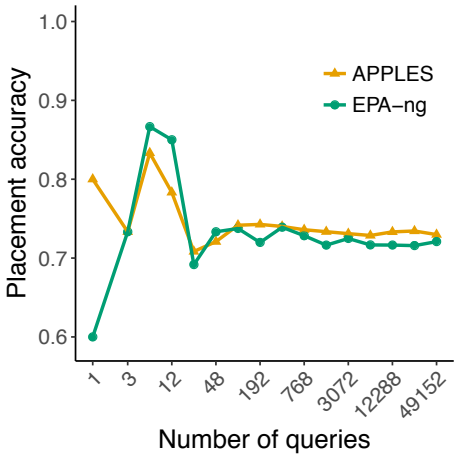


Figure 3.8. Scalability with respect to the number of queries. Accuracy with respect to increased numbers of queries (k) in one execution given 28 CPU cores and 28 threads on a Intel Xeon E5 CPU with 64 GB of memory.

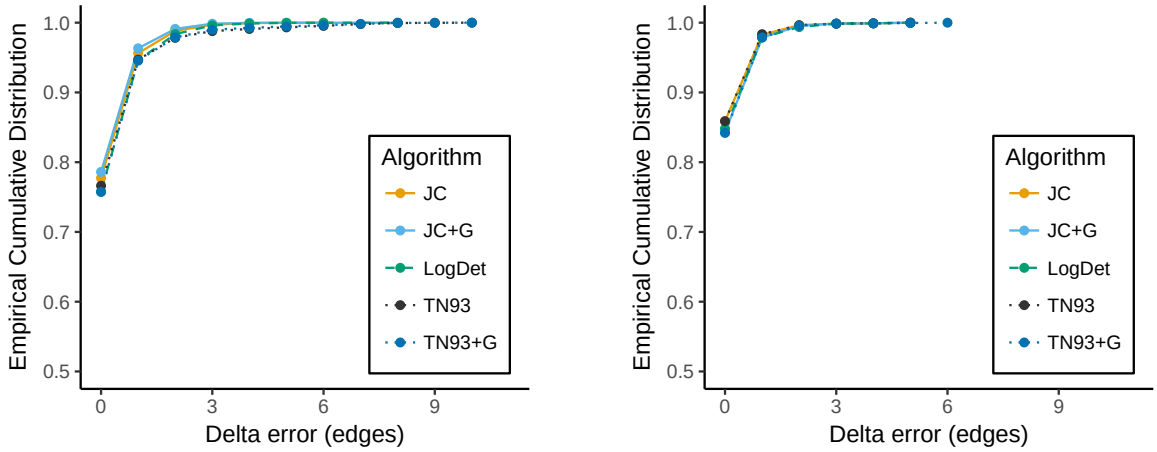


Figure 3.9. Comparing various models of DNA evolution. For the GTR (a) and RNASim-heterogeneous (b) datasets, we show the delta error (edges) of APPLES* run with five distance matrices calculated based on different models of DNA evolution. All model parameters are estimated per pair of sequences. The five models have similar accuracy.

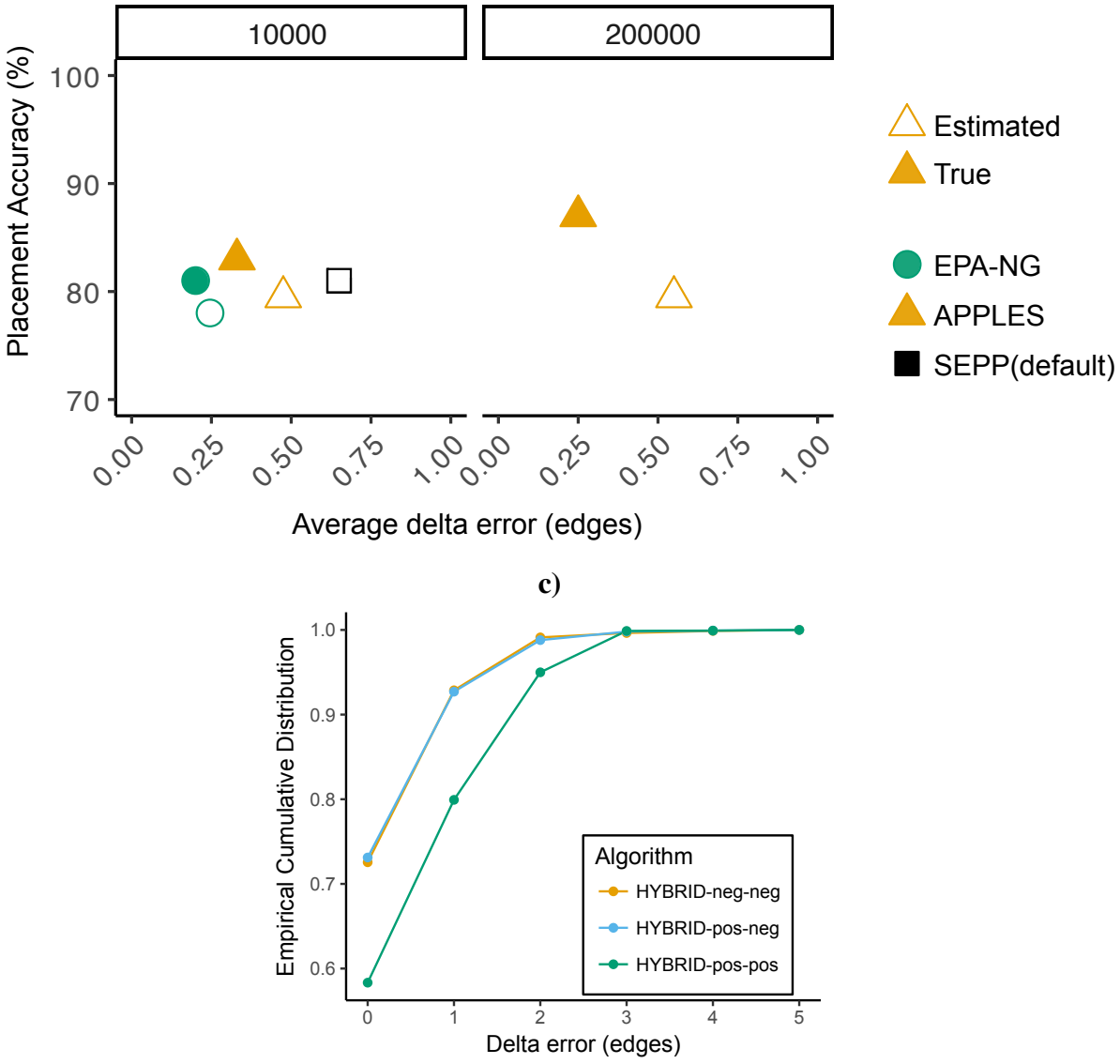


Figure 3.10. (a,b) The effect of imposing positivity constraint on error. We show the error of (a) APPLES-MLSE and (b) APPLES-ME run both with and without enforcement of non-negative branch lengths on RNASim heterogeneous dataset. Accuracy improves substantially for MLSE whereas it reduces drastically for ME. **(c) The effect of imposing positivity constraint on accuracy on Hybrid.** The HYBRID approach does not benefit from imposing positivity constraint on its second (ME) stage.

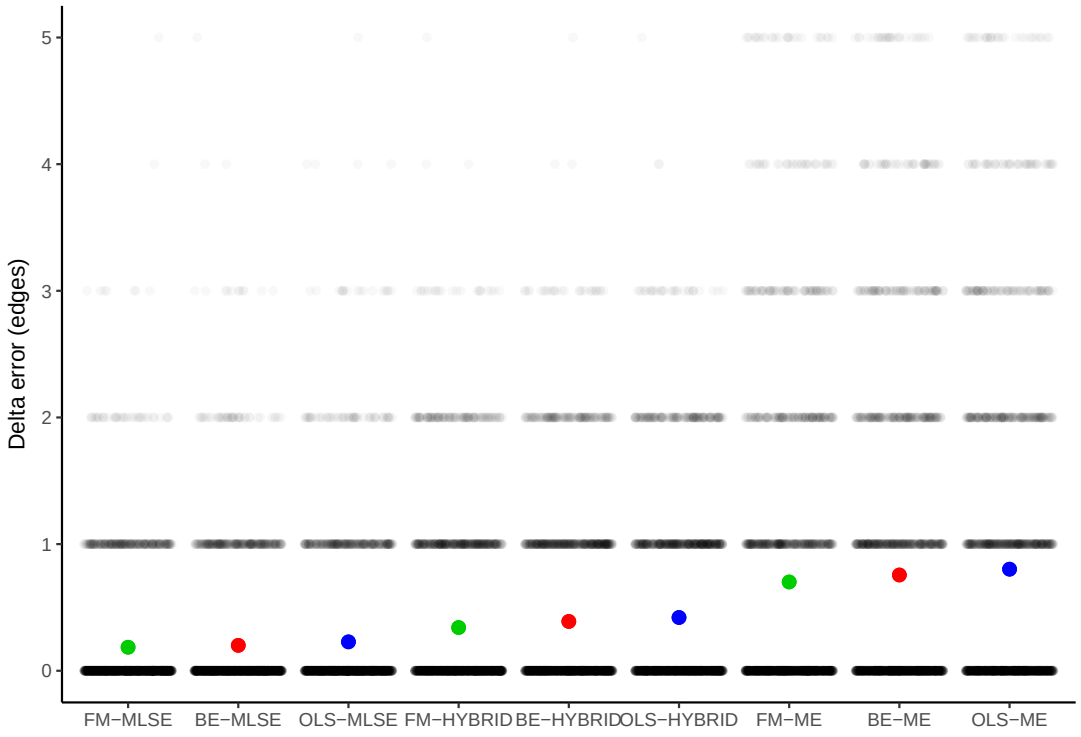


Figure 3.11. Comparing APPLES versions. For the RNASim dataset (without controlling for the diameter), we show the delta error (edges) of APPLES run with three options for weighting: FM (green), BE (red), and OLS (blue), and three options for selection strategy (MLSE, ME, and Hybrid). For each method, the mean (colored circle) and standard errors (lines; too small to see) are shown over 2500§ data points, each shown as dots. Some of the methods occasionally have error above 5 branches, but for better resolution, we cap the y-axis at 5.

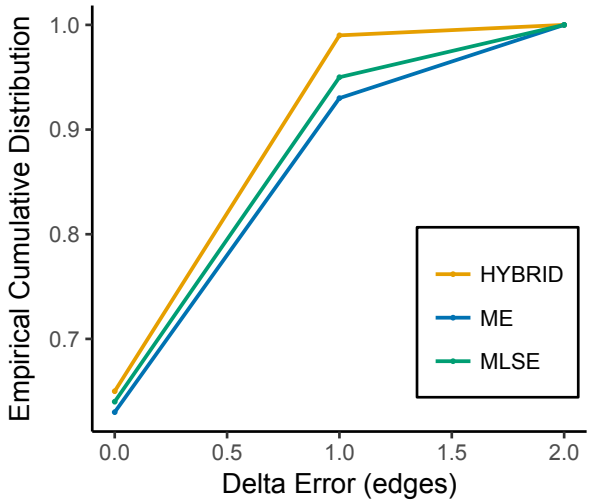


Figure 3.12. APPLES-HYBRID has higher accuracy on sparse RNAsim dataset On the RNAsim dataset, we chose 20 sequences randomly from the larger RNAsim-heterogeneous dataset; here, APPLES-HYBRID has higher accuracy than APPLES* (MLSE).

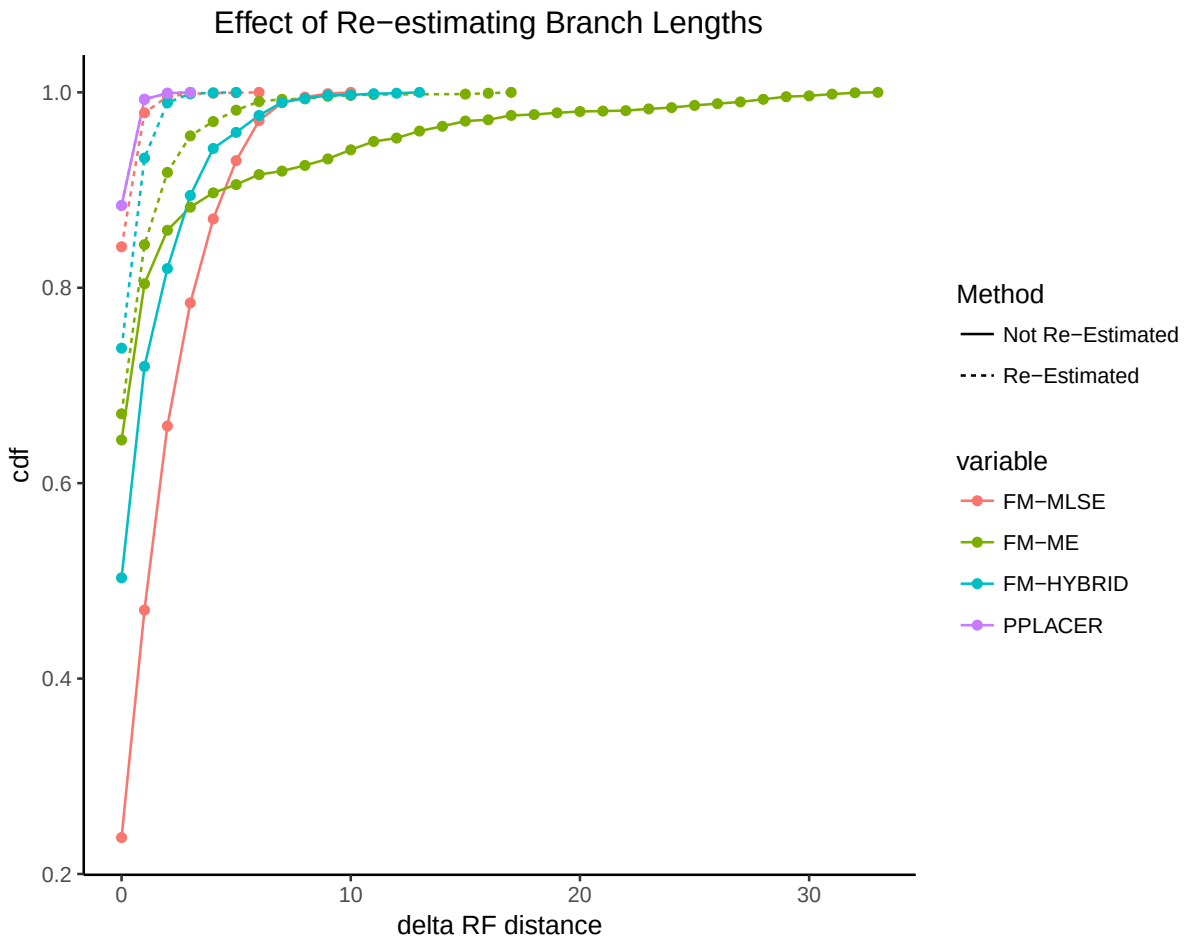


Figure 3.13. The effect of reestimating branch lengths of the backbone tree on accuracy. We show the accuracy of pplacer and APPLES-FM with its three optimization criteria. APPLES is run both with (dotted) and without (solid) re-estimating branch lengths in the backbone tree using the same model (here, TN93+Γ) used for computing distances of query sequences to backbone sequences. FastME* is used to re-estimate branch lengths. Accuracy improves dramatically by recomputing backbone branch lengths using the same model. The case labeled "Not re-estimated" uses branch lengths produced using RAxML under the GTR+Γ model.

Chapter 4

Fast and Accurate Distance-based Phylogenetic Placement using Divide and Conquer

Phylogenetic placement of query samples on an existing phylogeny is increasingly used in molecular ecology, including sample identification and microbiome environmental sampling. As the size of available reference trees used in these analyses continues to grow, there is a growing need for methods that place sequences on ultra-large trees with high accuracy. Distance-based placement methods have recently emerged as a path to provide such scalability while allowing flexibility to analyze both assembled and unassembled environmental samples. In this paper, we introduce a distance-based phylogenetic placement method, APPLES-2, that is more accurate and scalable than existing distance-based methods and even some of the leading maximum likelihood methods. This scalability is owed to a divide-and-conquer technique that limits distance calculation and phylogenetic placement to parts of the tree most relevant to each query. The increased scalability and accuracy enables us to study the effectiveness of APPLES-2 for placing microbial genomes on a data set of 10,575 microbial species using subsets of 381 marker genes. APPLES-2 has very high accuracy in this setting, placing 97% of query genomes within three branches of the optimal position in the species tree using 50 marker genes. Our proof of concept results show that APPLES-2 can quickly place metagenomic scaffolds on ultra-large backbone trees with high accuracy as long as a scaffold includes tens of marker genes. These

results pave the path for a more scalable and widespread use of distance-based placement in various areas of molecular ecology.

4.1 Introduction

Phylogenetic placement of query samples on an existing phylogeny is increasingly used in diverse downstream applications such as microbiome profiling [8, 43, 101, 151, 152, 174, 233], genome skimming [26], and epidemic tracking [133, 234]. A main attraction of placing new sequences onto an existing phylogeny is computational expediency: the running time of phylogenetic placement is a fraction of the time needed for *de novo* reconstruction and can grow linearly with the number query samples assuming they are placed independently. To take advantage of this potential, many methods have been developed using a wide range of algorithmic techniques [e.g., 10, 17, 29, 103, 135, 153, 164, 191, 221, 261].

A major attraction of phylogenetic placement is that it enables placement of sequences on very large trees. In applications of placement for microbiome analyses, sequences obtained from amplicon sequencing or metagenomic samples are placed into a reference phylogeny composed of known organisms. Depending on the datatype and pipeline, we may decide to place reads directly [164, 174] or may place marker genes obtained from metagenome-assembled genomes (MAGs) [8]. Large 16S databases have existed for more than a decade [48, 190] and genome-wide references trees with ten thousand species and more have been developed recently [e.g., 182, 263]. Moreover, close to a million microbial genomes are available in the RefSeq and GenBank databases. Although there is much redundancy among assembled genomes, we can expect that even larger and more diverse reference trees will be available in the near future. Development of bigger reference sets has a strong motivation: the density of reference set has been known to play a crucial role in the accuracy of downstream analyses [156, 172, 185]. Thus, if downstream methods can handle them, we should ideally use these dense reference data sets.

Despite their promise, two types of challenges emerge when reference data sets increase

in size: scalability and accuracy. The issue of scalability is well-understood: placement methods may not be able to place on ultra-large reference trees with reasonable running time, and equally important, with reasonable amounts of memory. Moreover, handling ultra large reference trees can be subject to numerical issues. Less appreciated is the observation that as the data set size increases, the accuracy of the algorithms may reduce and updated strategies may be needed. Thus, for placement methods to reach their full potential and take advantage of the available ultra-large reference trees, both scalability and accuracy of existing methods need to improve.

One recent advance in phylogenetic placement on ultra-large reference trees is the development of distance-based placement method, implemented in a method called APPLES [14]. Distance-based placement relies on computing distances between the query and references and finding the placement most congruent with these distances. In extensive simulation studies, [14] found APPLES to come very close in accuracy to a leading maximum likelihood method, pplacer [153], but unlike ML methods, was able to scale to trees with up to 200,000 taxa. Moreover, APPLES is more useful for studying ecological data because it allows assembly-free and alignment-free placement of genome skims. Despite the relatively high accuracy and scalability of APPLES, it has room for improvement. Its memory usage and speed both grow linearly with the size of the data set, which can start to become slow for references with many hundreds of thousands of species. A bigger challenge is that computing distances across very diverse species found in ultra-large trees can lead to low accuracy, an issue that APPLES only tried to address using weighted distances. A more direct algorithm that accounts for very diverse sequences in the backbone has the potential to further improve accuracy. Moreover, APPLES lacked several features that help usability (including handling of amino acids and building precomputed reference packages). Finally, APPLES has not been tested in the context of microbiome analyses with large backbone trees where it has the most potential.

In this paper, we introduce APPLES-2, a method that, compared to APPLES, improves both accuracy and scalability by adding a divide-and-conquer mechanism and several other features. We test APPLES-2 on both simulated and empirical data sets representative of microbiome

analyses. We show that it can place scaffolds from a metagenomic sample onto a large reference tree of more than 10,000 species both given marker genes found in assembled scaffolds.

4.2 Materials and Methods

4.2.1 APPLES-2 Algorithm

Background

[14] introduced a Least Squares Phylogenetic Placement (LSPP) framework and a method called APPLES for distance-based placement. In this framework, the input to APPLES is a reference (*a.k.a* backbone) phylogenetic tree T with n leaves and a vector of distances δ_{qi} between a query taxon q and every taxon i on T. Although machine-learning based methods shows substantial promise [103], typically, input distances are obtained by calculating sequence distances between query and backbone taxa followed by a phylogenetic correction using a statistically consistent method under a model such as Jukes-Cantor (JC69) [105]. APPLES introduced a dynamic programming algorithm to find a placement of q that minimizes weighted least squares error $\sum_{i=1}^{n} w_{qi}(\delta_{qi} - d_{qi}(T))^2$ where $d_{qi}(T)$ represents the path distance from q to backbone taxon i on T. APPLES, by default, sets $w_{qi} = \delta_{qi}^{-2}$ following [68] (FM) weighting.

Divide-and-conquer Placement Algorithm

The most consequential change in APPLES-2 is that it adopts a divide-and-conquer approach to improve both accuracy and scalability using two inter-related techniques. There is strong evidence in distance-based phylogenetics literature that correction for high variance occurring in estimation of long distances can result in dramatic improvements in accuracy [50, 66, 246]. For example, the DCM family of methods that result in fast converging methods [61, 97, 98] mostly rely on dividing taxa into smaller subsets with lower distances. To take advantage of this insight, we enable APPLES-2 to use distances that are either smaller than a threshold d_f or among the lowest b distances. Ignoring distances larger than the d_f threshold also gives us an opportunity to avoid computing all n distances so that the running time could

grow sublinearly with the size of the reference tree. To do so, we divide the backbone tree T into subsets that are somewhat larger than d_f in diameter (maximum pairwise path distance between any two leaves), choose one representative from each subset, and compute distances of the query only to these representatives. Then, we compute all distances in the cluster with the least distance to our query taxon.

More formally, without loss of generality, we assume that for a certain query taxa q, $\delta_{q1} \leq \delta_{q2} \leq \delta_{q3} \leq \ldots \delta_{qn}$ holds. The first parameter we introduce is $d_f \in \mathbb{R}_{\geq 0}$ which sets $w_{qi} = 0$ (i.e. ignore backbone taxon i) when $\delta_{qi} \geq d_f$ for a query taxon q. In addition, we introduce a second parameter $b \in \mathbb{N}_{\geq 0}$ which forces to retain the standard weighting (i.e. $w_{qi} = \delta_{qi}^{-2}$) for backbone taxa $1 \leq i \leq b$, regardless of δ_{qi}. Consequently the new LSPP objective function becomes $\sum_{i=1}^{b} w_{qi}(\delta_{qi} - d_{qi}(T))^2 + \sum_{i=b+1}^{n} \mathbf{1}(d_f - \delta_{qi}) w_{qi}(\delta_{qi} - d_{qi}(T))^2$ where $\mathbf{1}(x)$ is the unit step function: $\mathbf{1}(x) = 0$ for $x < 0$ and $\mathbf{1}(x) = 1$ for $x \geq 0$. We discuss default values below.

To avoid computing all distances, during preprocessing of the reference set, we cluster the backbone alignment and tree T using the linear-time TreeCluster algorithm [12] to find the minimum number of clusters such that the maximum pairwise distance in each cluster is no more than $1.2 \times d_f$. The threshold 1.2 is chosen empirically, and APPLES-2 is robust to this choice (see Fig. 4.7). Then, we select a representative sequence per partition by computing consensus sequence among all sequences belonging to the partition. Let $P_1, P_2, \ldots, P_k$ denote partitions of leaves of T and $C(j)$ denote centroid sequence of partition P_j. Without loss of generality, we assume that $\delta_{qC(1)} \leq \delta_{qC(3)} \leq \delta_{qC(3)} \leq \ldots \delta_{qC(k)}$. The distance between q and a backbone taxa $i \in P_j$ is computed only if either $\delta_{qC(j)} \leq d_f$ or $\sum_{i=1}^{j-1} |P_i| < b$ holds. The time complexity of distance calculation per query is in the order of $O(\max(b,m)L)$ where L is alignment length and m is number of backbone taxa whose distance to the query is less than or equal to d_f.

Since in APPLES-2 a subset of distances are calculated, we have re-designed its dynamic programming algorithm so that it automatically works on the backbone tree induced to the taxa for which distances are computed. The updated dynamic programming algorithm scales with the number of edges in the induced tree, which can be as low as $O(\max(b,m))$ (if the chosen leaves

are a connected subtree) and as high as $O(n)$ (when chosen leaves span all of a caterpillar tree).

New features in APPLES-2 Software

Protein distances. Several tools [123, 200, 218, 247] offer distance calculation from protein sequences using analytical [105, 108, 218] and maximum likelihood (ML) [122, 245] models. In order to provide support for protein alignments, we implement Scoredist algorithm, which has achieved better accuracy than other analytical models in previous tests [218]. Scoredist computes pairwise distances according to the BLOSUM62 [86] matrix, normalizes the distances with respect to expected distance and minimum possible distance, applies a logarithmic correction, and scales distances using empirically derived coefficients. Like JC69 distances, in APPLES-2, Scoredist distance calculation is powered by Numpy [82] vectorized operations and is extremely fast.

BME weighting. APPLES implemented three weighting schemes FM [68], BE [22], and OLS [35]. [14] demonstrated that FM weighing given by $w_{qi} = \delta_{qi}^{-2}$ results in the best placement accuracy among these methods. However, it did not implement balanced minimum evolution (BME) weighting [51], which has been among the most promising methods. APPLES-2 implements BME which corresponds to setting $w_{qi} = 2^{-(1+p_{qi})}$ where p_{qi} is the topological distance between q and a backbone taxa i. Note that BME weights are much more challenging to incorporate into the dynamic programming because a BME weight is not simply a function of calculated distances but is rather a function of the placement on the tree. Thus, unlike the previous weighting schemes, the BME weight changes as we examine different placements. Overcoming this hurdle required implementing a more complex dynamic programming.

Database features. We allow precomputation of a database (called APPLES database) that consists of a backbone alignment and tree, including centroid sequences and the leaf clustering, which can be stored and distributed. The database can be reused for the analysis of different query data sets. Moreover, when a backbone alignment is provided, APPLES-2 can re-estimate branch lengths of the input tree using FastTree-2 [188] under the JC69 model to match the model

used for estimating distances.

4.2.2 Experiments

Data sets

RNASim data set.

We reuse the RNASim-VS simulated RNA data set from [14], which consists of subsets of a simulated RNASim data set [80], but change the query selection strategy. We begin with randomly selecting 200 queries with various novelty levels; to control novelty, 10 taxa are randomly selected from each of 20 bins determined by dividing terminal branch length of the phylogeny on 200,000 taxa into 20 quantiles. The remaining 199,800 taxa are designated as backbone. Then, we create data sets with size (n): 100000, 50000, 10000, 5000, 1000, and 500 by successive random subsampling. The procedure is replicated 5 times and query taxa are identical within a replicate across different size data sets. Each replicate contains a 1596-sites multiple-sequence alignment of a single gene and the true tree. 200 query are placed on the backbone independently for each replicate. We also adopted the RNASim-QS data set from [14] also based on the RNASim data set [80]; this data set comprises five replicates with varying numbers of queries, ranging from 1 to 49,152 with backbones of size $n = 500$. In both RNASim data sets, backbone tree topology and maximum likelihood branch lengths are estimated from true MSA using FastTree-2 [188] according to GTR+Γ model. In all cases, branch length is re-estimated using FastTree-2 [188] to be consistent minimum evolution units.

Web of Life (WoL) data set.

[263] built a species tree of 10,575 prokaryotic genomes from 381 marker genes using ASTRAL [258]. We first determine a set of marker genes according to several selection strategies which will be discussed later. We remove sites that contain gaps in 95% or more of the sequences in the protein MSA using TrimAl [34]. The trimming is only to speed up analyses and has no positive impact on accuracy; in fact, it very slightly *decreases* accuracy (see Fig. 4.8). Then, we create three concatenated alignments; the amino acid alignment, a nucleotide alignment with all

three codon positions (C123), and another with third codon position removed (C12). Unless it is stated otherwise, we use the C12 nucleotide MSA in our analyses.

We analyze the WoL data set in four ways (Table. 4.1). In WoL-main, three data sets of size (n) 9000, 3000, and 1000 with 10 replicates are created by successively subsampling the protein MSA of the selected marker genes at random. From the remaining 1575 species, 1000 are randomly subsampled from the protein MSA of the selected marker genes and designated as query. For all data set sizes, we use the ASTRAL tree available from the original publication induced to backbone species as the backbone tree. However, we recompute its branch lengths using FastTree-2 [188] in the minimum evolution branch length unit. We determine a marker gene set by controlling for two parameters; the number of genes (k) and a selection strategy. Two selection strategies are *random* (among all 381) and *best*, which means top k marker genes with the lowest quartet distance [208] to the species tree are selected. In WoL-main, we choose $k = 50$ coupled with the *best* strategy (which results in lowest, median, and highest quartet-distance to be $0.058, 0.125, 0.17$ respectively). Concatenated MSA using the default marker gene set contains $71,798$ nucleotide sites. In WoL-random (Table 4.1), we create 1000-species backbone alignments by selecting $k \in \{10, 25, 50, 381\}$ coupled with the *best* and *random* strategies. Additionally, marker gene set selection is replicated five times for the *random* strategy. In the previous two data sets, the backbone was inferred with queries included, which were then removed, because repeating the complex backbone inference pipeline for all analyses was not doable. However, we did add a smaller analysis that avoids this information leakage. In WoL-denovo, we reuse a single replicate under data set sizes 1000 and 3000 from WoL-main and fully reproduce WoL pipeline [263] to obtain de-novo MSA and species tree instead of removing queries from the full tree. All query genes are then independently aligned to de-novo MSA of the 50 backbone marker genes using UPP [176].

Table 4.1. WoL based data sets. In Wol-random data set, only random selection of marker genes is replicated 5 times. *best* marker selection strategy indicates choosing the marker genes whose gene tree has the lowest topological discordance with the species tree. An alignment or backbone tree is induced when it is taken from a larger data set (e.g. full data set). C12: nucleotide alignment with first and second codon positions. C123: nucleotide alignment with all codon positions. AA: amino-acid alignment.

data set name	Backbone Size	Number of markers	Marker strategy	Backbone tree & MSA	Query alignment	Replicates	Character
WoL-main	1000, 3000, 9000	50	best	induced	induced	10	C12, C123, AA
WoL-random	1000	10, 25, 50, 381	best, random	induced	induced	5*	C12
WoL-denovo	1000, 3000	50	best	de-novo	UPP	1	C12
WoL-metagenomic	10375	381	all	induced	UPP	1	C12

Data set of Simulated genome assemblies and scaffolds.

In WoL-metagenomic data set, we utilize a protocol for generating simulated genome sequencing data which begins with randomly selecting 200 test genomes from the WoL data set (10 genomes are randomly selected from each of 20 genome bins of equal genome count with the bins determined by ascending terminal branch length). Next, we run InSilicoSeq [79] v1.5.1 (using NovaSeq settings) to simulate 3 M 150 bp paired-end reads per genome. For assembly, first we run PEAR [260] v0.9.11 to merged read pairs, then run SPAdes [16] v3.14.1 with a *k*-mer size cascade of 21,33,55,77,99 to assemble them into scaffolds. We then run Prodigal [99] v2.6.3 to identify open reading frames (ORFs) from the scaffolds, and finally run PhyloPhlAn [215] commit 2c0e61a to identify the same 381 marker genes.

Selected test genomes are removed from the backbone set, which leaves us with 10375 species in the backbone. All the genes were then independently aligned to the backbone marker genes using UPP [176], and markers from the same assembly or scaffold were concatenated. We try to place the samples on the backbone using either 1) the assembly (i.e., which can be fragmented and can include errors, compared with the genome from which it is simulated), or 2) individual scaffolds (small portions of the genome). We only include scaffolds that are ≥ 10 kbp in our analyses. Note that here, instead of testing on microbial communities, we use an *in-silico* approach and simply generate reads from individual microbial genomes and assemble

them separately. We leave it to future work to simulate mixed metagenomic reads and evaluate accuracy under such scenarios.

Biological TD-metagenomic dataset.

We use the metagenome-assembled genomes (MAGs) from a study by [262], which identified novel pathogenic profiles from the fecal samples of 22 Traveler's Diarrhea (TD) patients and seven healthy traveler (HT) controls. The dataset consists of 320 manually curated MAGs (bins) and 6653 scaffolds that are 50kb or longer. The 381 marker genes in the dataset are identified using the same protocol as the WoL study. We use the species tree in WoL study as the backbone tree and align the sequences from Traveler's Diarrhea dataset to the WoL dataset using UPP independently for each marker gene. We then concatenate the 381 marker genes from the same bins or scaffolds and use them for placement. We also study the case where we filter out the scaffolds with less than or equal to 10, 20, 30, or 40 marker genes, which reduce the number to 4522, 1608, 668, and 320 scaffolds, respectively.

Methods compared

For APPLES-2, we explored various options for d_f and b in an experiment performed on the WoL-main data set (Fig. 4.9) . As a result, we set $d_f = 0.2$ and $b = 25$ by default and keep these values fixed across all of our other experiments. For RNASim-VS and WoL-main data set, in addition to APPLES, we compare APPLES-2 to two ML methods, pplacer [153] and EPA-ng [17]. We run pplacer (v1.1.alpha19-0-g807f6f3) and EPA-ng (v0.3.8) in their default mode using GTR+Γ model and use their best hit (ML placement). Unlike the procedure used by [14], we do not perform branch length re-estimation on backbone tree using RAxML-8 [220]. Instead, we input inferred FastTree-2 tree and model parameters without modification to EPA-NG and pplacer. We compare to EPA-ng in analyses that concerned scalability (e.g., RNASim-QS).

Evaluation Criteria

In RNASim analyses, we use the known true tree as the gold standard whereas on the empirical data, we use the ASTRAL tree on the full set of species as the gold standard with an exception of WoL-denovo data set in which the ASTRAL tree is computed denovo for each data set size. In all WoL data sets except WoL-denovo, we measure the accuracy of a placement using the number of edges between the position on the gold-standard tree and the inferred placement (i.e., node distance [136]). In the simulated RNAsim data set, because true tree is known and we place on the estimated tree and not the true tree, we need to slightly update the metric of the error: We use *delta error*, which measures the increase in the number of false-negative bipartitions after placement compared to before placement [164]. We use delta error in WoL-denovo data set as well, treating the published phylogeny on the full set as the true tree.

4.3 Results

4.3.1 Single-gene Placement

We start with leave-one-out experiments on an existing single-gene simulated RNASim data set where all methods face model misspecification. Despite the model misspecification, APPLES-2 is able to find the best placement of query sequences with up to 91% accuracy (placement on the correct branch) when the backbone size is $n = 200,000$ (Fig.4.1a, Table 4.2). APPLES-2 has lower mean delta error (-0.08 edges on average) and higher accuracy (+2.5% on average) compared to APPLES for all cases except for $n \leq 5000$, where they are essentially tied in accuracy but APPLES has slightly higher mean delta error. Across all cases, pplacer is the most accurate method. In particular, pplacer has 10% better accuracy and 0.13 less mean delta error than APPLES-2 for $n = 500$. However, the difference in accuracy and mean error gradually decrease as n increases and diminish to only 2% and 0.04 respectively for $n = 200,000$. Compared to the other ML method, EPA-NG, APPLES-2 either matches (for $n \leq 5000$) or improves the accuracy (up to 3%) on instances where EPA-NG manages to

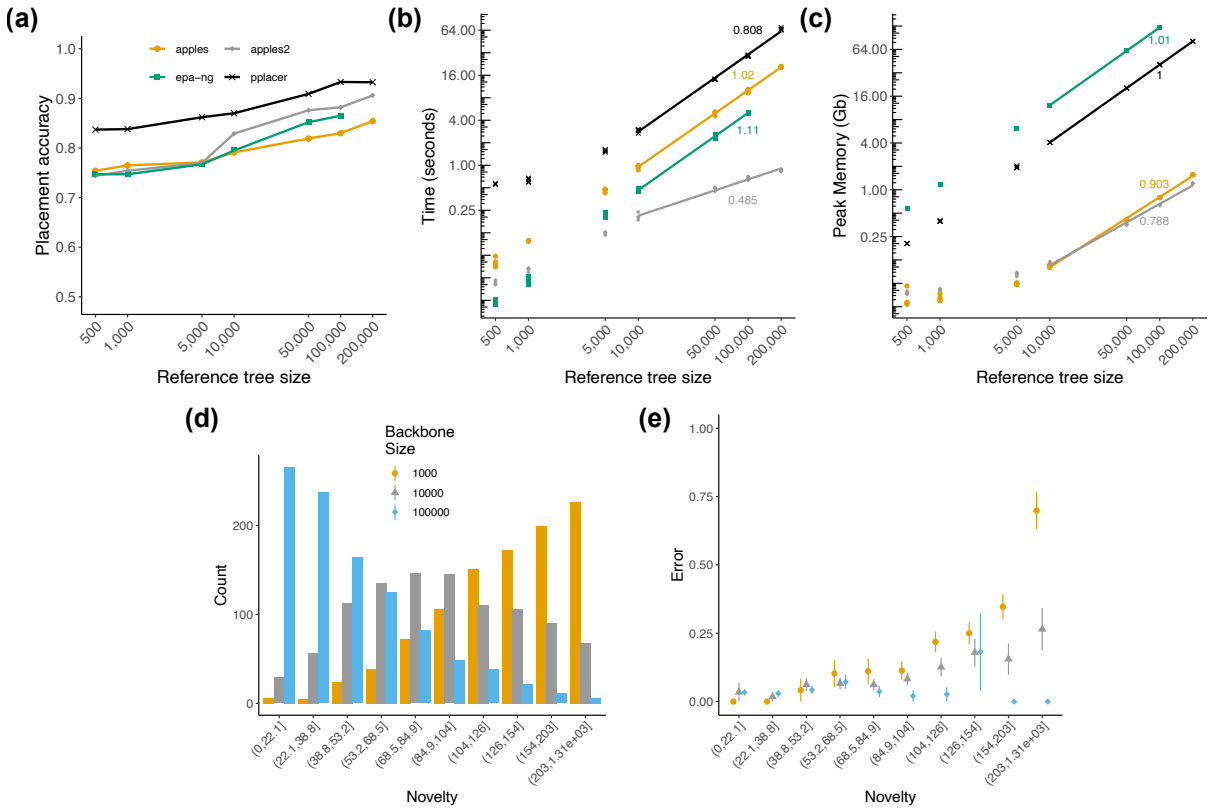


Figure 4.1. Results on RNASim-VS. Placement accuracy (a), running time (b), and peak memory usage (c) for a single placement with taxon sampling ranging from 500 to 200,000. (b,c) Lines are fitted in the log–log scale and their slope (indicated on the figure) empirically estimates the polynomial degree of the asymptotic growth. Lines are fitted to ≥ 10000 points because the earlier values are small and irrelevant to asymptotic behavior. All calculations are on 36-core, 2.6GHz Intel Xeon CPUs (Sandy Bridge) with 128GB of memory, with each query placed independently and given 1 CPU core and the entire memory. Missing results (EPA-NG on tree size $200,000$) indicate that the tool fails to run or complete in 48h. (d) Queries are grouped into deciles based on their novelty with respect to backbone set of species, defined as the terminal branch length of the query in the gold-standard tree, induced to backbone and query species. (e) Mean placement error of APPLES-2 across increasing level of query novelty for all three backbone sizes.

complete ($n \leq 100,000$). Thus, APPLES-2 matches or improves the accuracy of one ML method (EPA-ng) and is slightly below the accuracy of the other (pplacer).

Placement accuracy of APPLES-2 is 17% higher on largest tree than the smallest tree. To examine the reason, we first observe that the novelty of the test set (defined as terminal branch length in the true tree) decreases as the backbone size increases (Fig 4.1D). To test the impact of

Table 4.2. Percentage of correct placements (shown as %) and the average placement error (Δe) on the RNASim-VS with various backbone size (n). % and Δe is over 1000 placements (except $n = 200,000$, which is over 200 placements). *n.p* indicates tool failed to run in the case.

	n = 500		n = 1,000		n = 5,000		n = 10000		n = 100,000		n = 200,000	
	%	Δe	%	Δe	%	Δe	%	Δe	%	Δe	%	Δe
APPLES-2	74	0.31	75	0.33	77	0.34	83	0.24	88	0.14	91	0.11
APPLES	75	0.36	77	0.34	77	0.36	79	0.34	83	0.28	85	0.22
EPA-ng	75	0.29	75	0.28	77	0.24	80	0.21	87	0.14	n.p	n.p
pplacer	84	0.18	84	0.18	86	0.14	87	0.14	93	0.07	93	0.07

novelty on error, we measure the mean error for each decile of novelty for all backbone sizes after larger trees are pruned so that backbone trees are identical to those of the smallest tree. This pruning ensures that errors are always computed with respect to trees of the same size and are therefore comparable. Two patterns stand out. First, increasing novelty does increase the error, especially for smaller backbone sizes (Fig 4.1E). Thus, the error with larger backbone trees is reduced simply because fewer queries are novel (Fig 4.1D). More interestingly, it appears that at higher levels of novelty, the error is reduced with larger backbones even after the novelty level is controlled. Thus, the results show improved accuracy with the increased taxon sampling even when the novelty of the test set does not change. We note that larger backbone trees include fewer long branches in the backbone and that processes such as long branch attraction need at least two close long branches (e.g., one in the backbone and one for the query) to impact results.

Our benchmarking indicates that the running time of APPLES-2 grows empirically as $O(n^{0.45})$ (Fig. 4.1b); this sublinear running time growth with the backbone size is consistent with our theoretical expectations. APPLES-2 is the fastest method on backbones with 5,000 or more taxa, offering up to 24$\times$ speed-up in average compared to APPLES on a 200,000-taxon tree. EPA-NG is faster than pplacer and APPLES but slower than APPLES-2 (with running time that grows super-linearly). On the 100,000 taxon data set, EPA-NG and pplacer take 7.3$\times$ and 77$\times$ longer than APPLES-2 on average respectively. APPLES-2 and APPLES consistently uses less memory than ML tools (Fig. 4.1c) and are the only tools with sublinear memory complexity (empirically close to $O(n^{0.8})$ for APPLES-2). On the largest backbone tree with $200,000$ taxa,

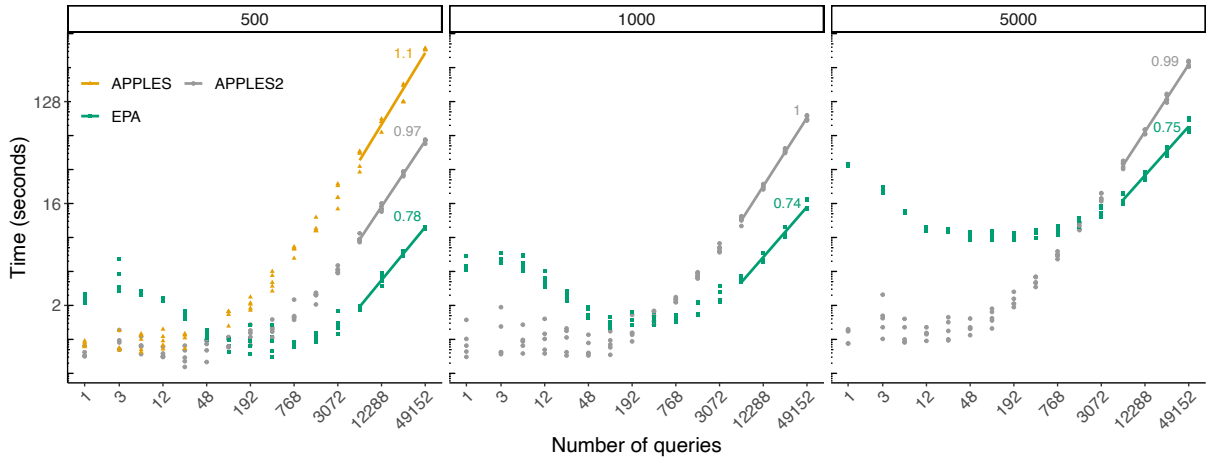


Figure 4.2. Scalability with respect to the number of queries. We show wall-clock running time with respect to increased numbers of queries placing on a tree with 500 taxa in one execution of the tool given 28 CPU cores and 28 threads on an Intel Xeon E5 CPU with 64 GB of memory. Lines are fitted to $x \geq 6144$ points because the earlier values are small and irrelevant to asymptotic behavior.

APPLES-2 requires only 1.2GB of memory compared to 81GB needed by pplacer. EPA-NG uses 192× more memory then APPLES-2 on the largest backbone tree with 100,000 taxa where both tools successfully run.

We also evaluated the impact of the number of queries on the running time (Fig. 4.2), comparing APPLES, APPLES-2, and EPA-NG, all run in the parallel mode. On 500 taxa backbone, all three methods finish placement of up to 1,536 queries in less than 4 seconds given 28 CPU cores with no clear trend in running time. EPA-NG is able to place 49152 queries in 10 seconds on average, 5.8 times faster than the second best method APPLES-2, which takes 57 seconds and is 6.5 times faster than APPLES. The comparison of EPA-NG and APPLES-2, the fastest two of the three methods, on backbone trees with 1000 and 5000 taxa shows that EPA-NG is 6 and 3.4 faster than APPLES-2 respectively on the largest query set. While both methods complete in less than 36 seconds, APPLES-2 is faster than EPA-NG when number of queries is less than or equal to 1536 for 5000 taxa tree. Running times of EPA-NG, which is designed specifically for very large numbers of query sequences, can surprisingly decrease when given more queries. For any backbone size, APPLES and APPLES-2 start to have substantial

increase in running time after placing 6144 queries or, scaling linearly with respect to the number of queries; surprisingly, EPA-NG grows at a sublinear rate, likely indicating that it requires more queries to display its asymptotic behavior. To summarize, while APPLES-2 is faster than EPA-NG given hundreds of queries, EPA-NG scales better as the number of queries increases.

4.3.2 Multi-gene Web of Life (WoL) data set

We next test the utility of distance-based phylogenetic placement on a real WoL biological data set [263] with 381 marker genes and 10575 microbial taxa. When we concatenate the best 50 marker genes, APPLES-2 achieves outstanding accuracy, placing query sequences with 75% accuracy and 0.50 edges of error on average on backbones with 1000 taxa (Fig. 4.3a). A striking 97% of the queries are placed within three or fewer branches away from the optimal branch (in a tree with a diameter of 58.3 branches on average). Note that here we are using 50/381 marker genes and a much simpler methodology compared to the original study. In comparison, APPLES achieves 60% accuracy with 1.1 average error on the same data set. As in the single-gene RNASim dataset, pplacer is the most accurate method with 80% accuracy for $n = 1000$. EPA-NG has slightly lower accuracy (-1%) and mean error (-0.06) than APPLES-2. As the size of the reference increases from 1000 to 3000 and 9000, APPLES-2 and APPLES are only methods that run successfully due to large memory requirements of ML-based methods (more on performance below). APPLES-2 is able to maintain high accuracy, placing within three branches of the optimal placement in 97% and 96% of cases, respectively, for backbones of size 3000 (avg. diameter: 77.2 edges in average) and 9000 (avg. diameter: 105.5 edges). Increasing the backbone size also amplifies the gap between APPLES and APPLES-2, going from a difference of 0.57 edges of error on average for $n = 1000$ to 0.81 and 1.11 for $n = 3000$ and $n = 9000$.

APPLES-2 places queries with 0.1 higher error in average for $n = 9000$ compared to $n = 1000$; however, it should be noted that the largest tree has 9 times more branches than the smallest one. Therefore one branch of error in the smallest tree indicates a larger degree of misplacement. In order to establish a fair comparison between trees with different number

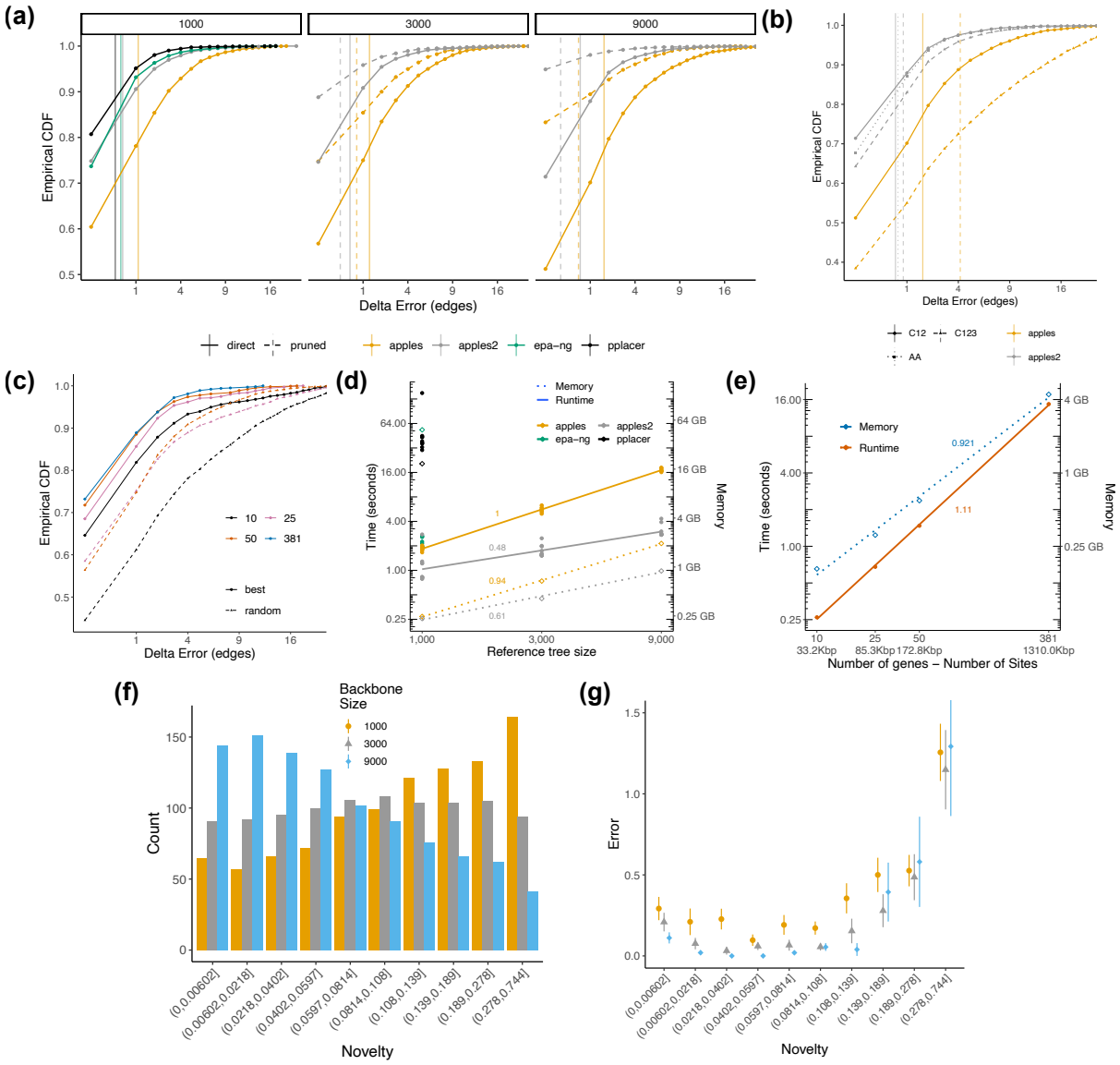


Figure 4.3. (a) Empirical cumulative distribution function (CDF) of placement error on backbones ranging from 1000 to 9000 taxa (10000 queries each). Vertical lines show mean error. (b) Placement accuracy versus alignment type. C12: only the first two codon positions are retained in the alignment; C123: all three positions used. (c) Impact of marker gene selection on placement accuracy. (d,e) Running time (solid lines and solid points) and memory (dotted lines, hollow points) performance with respect to backbone tree size (d) and number of marker genes in the backbone tree (e) in log-log scale. Each run has 32 cores and 56GB memory in a shared node with 2.25 GHz AMD EPYC 7742 processor with each query placed independently and given 1 CPU core. Missing results indicate that the tool fails to run or complete in 48h. (f,g) Novelty (defined as in Fig. 4.1) of queries and mean placement error of APPLES-2 for all backbone sizes.

of backbone species, after placement (i.e. before measuring the error), we prune trees with $n = 3000$ and $n = 9000$ to include only those present in the smallest tree with $n = 1000$. The comparisons on pruned trees shows that the placement accuracy for APPLES-2 becomes 90% and 95% on 3000 and 9000-taxa backbone tree respectively, which are much higher than 75% with 1000-taxa backbone tree (Fig. 4.3a). These increases in accuracy show that accuracy of APPLES-2, does, in fact, improve with better taxon sampling.

The reasons behind improved accuracy with improved taxon sampling parallel the simulated data. Again, we observe reduced novelty in the query set (Fig. 4.3f) as backbone size increases. Impact of novelty on error is not uniform. When the query is extremely similar to backbone species, correct placement is difficult. Thus, initially, the error slightly decreases as novelty increases. However, after reaching a sweet spot, the error increases dramatically as novelty increases. The better accuracy with larger trees, therefore, is a function of having fewer very novel queries. Controlling for novelty of query, in the first seven deciles, results show a negative correlation between the error and backbone size (Fig. 4.3g).

In WoL-main data set, backbone and query alignment and backbone tree is directly induced from full WoL data set, which may potentially "leak" information about query location since query sequences were present in the full data set during MSA and tree inference. We test this scenario on WoL-denovo data set where two MSAs and trees with $n = 1000$ and $n = 3000$ are de-novo inferred using the identical methodology described in the original publication [263]. In addition, query sequences are aligned to backbone MSA using UPP [176] to prevent leakage of information through alignment. We find a slight absolute reduction (-5%) in placement accuracy of APPLES-2 on 3000-taxa de-novo backbone compared to induced backbone (Fig. 4.10). However, the percentage of queries placed with no more than three edges error is 98% for both de-novo and induced backbone trees. The mean delta experiences very slight changes between de-novo and induced backbone trees.

APPLES-2 is the fastest method in WoL-main data set, managing to place a query in 1.1 second on average on the smallest backbone tree using a single CPU core (Fig. 4.3d). For

comparison, APPLES, EPA-NG, and pplacer take 1.86, 2.18, and 49.47 seconds per query respectively on the same data set. APPLES and APPLES-2 achieve the best memory efficiency by using 250Mb of memory whereas EPA-NG and pplacer use 48.6 and 18.6 GB of memory on the same instances. As backbone size increase to $n = 3000$ and $n = 9000$, APPLES and APPLES-2 become the only methods that complete the benchmark given a 56GB memory machine as ML-based methods terminate due to insufficient memory. Our benchmark indicates that running time and memory use of APPLES-2 grow sublinearly, achieving empirical time and memory complexity of $O(n^{0.5})$ and $O(n^{0.6})$ respectively.

Next, testing the impact of data type used for placement, we observe that removing the third codon position from nucleotide alignments improves placement accuracy substantially for both versions of APPLES (Fig. 4.3b). Interestingly, APPLES-2 seems to be more robust to inclusion of third codon position as the increase in the average error is 0.26 and 2.44 for APPLES-2 and APPLES, respectively. The third codon position often poses a stronger violation on stationarity assumption than the first and second codon positions [102, 186] and saturates faster, especially among very divergent taxa. Recall that APPLES-2 ignores distances among very divergent sequences, which is consistent with its higher robustness to the third codon position. Note that the original study [263] that built our gold-standard in these analyses inferred gene trees using amino acid data. We do not observe a substantial error difference between using nucleotide (first two codon positions) and amino acid sequences (Fig. 4.3b) for APPLES-2. Although APPLES-2 has 3% higher accuracy on the former data type, the number of queries with at most three edges of error is 96% on both data types. We remind the reader that amino acid distances are computed under the Scoredist algorithm, which is different from the models used in the original study to infer the reference tree [263].

Next, we test the impact of varying the number of marker genes and the type of genes used (randomly chosen or the *best* genes) on WoL-random data set (Fig. 4.3c) . While Using all the marker genes has the highest accuracy (mean edge error: 0.52; placement accuracy: 73%), using as few as 50 of best genes (i.e., those with gene trees with the lowest quartet distance

the species tree) comes very close. With 50 genes, APPLES-2 places 958 out of 1000 queries (96% rate) within three branches away from the optimal branch; in contrast, using all genes, 972 queries are within three branches (97% rate). Using the best 50 genes results in 0.63 average delta error, which is 0.11 more than using all genes in the data set. However, reducing the number of best genes to 25 and 10 increases error to 0.79 and 1.28 edges, respectively. Our benchmark indicates that runtime and memory use of APPLES-2 empirically grow near linearly with number of marker genes and number of sites in the backbone alignment (Fig. 4.3e). When all 50 marker genes used, placement of a query takes 1.5 seconds whereas using all 381 marker genes, placement takes 10 times longer on the 1000 taxa backbone. Thus, 50 best genes is the sweet spot in terms of accuracy among levels we test considering computational requirements.

There is a large difference between selecting genes randomly or using best genes (Fig. 4.3c). A random selection of 10 genes results in lower accuracy (within 3 edges from the optimal branch only 74% of the time) and a high average edge error of 3.29, whereas 10 best genes result in 1.28 edges of error on average. Going to 25 randomly selected genes provide acceptable placement accuracy where 87% of queries are placed within 3 edges from the optimal location (error: 1.8 edges on average); yet, 25 best genes continue to be better (error: 0.79 edges on average). With 50 genes, the error is 0.80 less when using best genes compared to randomly selected genes. The mean placement error using random genes decreases as the number of genes increases, culminating in 0.53 edges when all genes are selected (Fig. 4.11). Note that *best* and *random* strategies are equivalent when all 381 genes in the data set are used. Overall, the difference between random and best genes is wider when few genes are available and diminishes as more genes are added.

4.3.3 Placement of assemblies and scaffolds

While our previous analyses showed that APPLES-2 has outstanding accuracy using best or random subsets of marker genes sampled across microbial genomes, we often do not have entire genomes. Instead, we have MAGs and scaffolds from which MAGs are generated. We

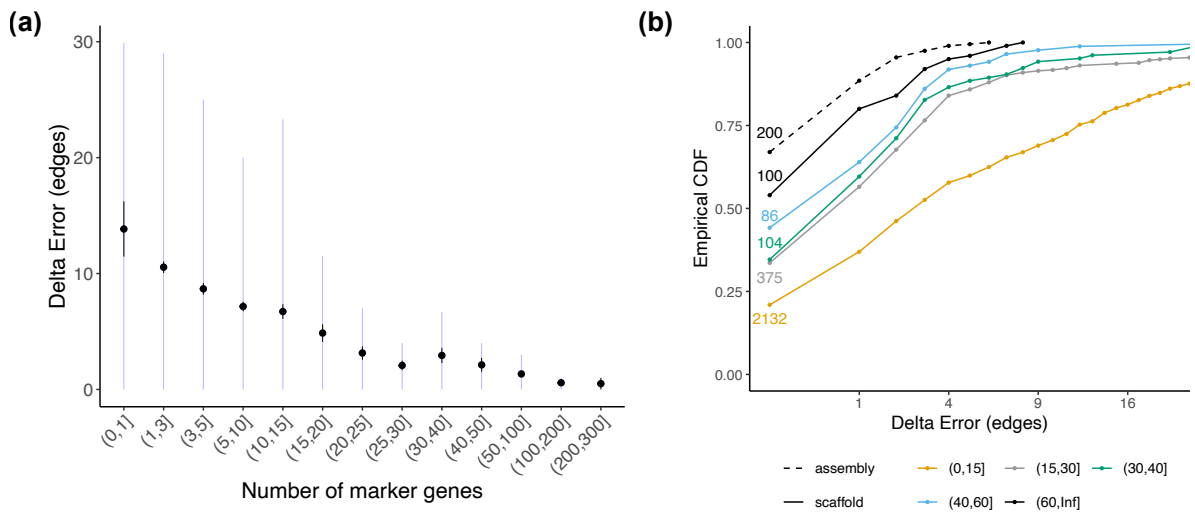


Figure 4.4. Results on WoL-metagenomic data set. (a) The relationship between number of marker genes in a scaffold and the error. Dots show mean, black error bars show standard error, and light blue error bars show the central 80% range. X-axis is binned non-linearly. (b) Placement error CDF for simulated metagenomic assemblies and scaffolds. Each bin indicates the number of genes in the scaffold or assembly. We show the number of queries in each bin next to its curve. The backbone has the diameter (the largest pairwise distance) of 106 edges.

next test APPLES-2 in a simulation that generated scaffolds and assemblies, similar to MAGs, by assembling reads simulated from a subset of 200 genomes in the WoL data set.

Our simulated assemblies included 105 to 365 marker genes (Fig. 4.12a). With these number of markers, APPLES-2 achieves 67% accuracy and places 195 of 200 simulated assemblies with error no larger than three edges (Fig. 4.4b). The error is never more than 6 edges. The placement error has a weak but statistically significant anti-correlation with the number of marker genes available in the assembly ($p = 0.002$ according to Pearson's correlation; $\rho = -0.216$; see Fig. 4.13).

Our assembly procedure produces 3318 unique scaffolds of $\geq$ 10kbp (Fig. 4.12b) , among which 665 has more than fifteen marker genes and 290 has more than 30 marker genes. The placement error is clearly a function of the number of genes in each scaffold (Fig. 4.4a). Scaffolds with less than 15 genes have high error on average (8.51 edges), but also have high variance (with 53% of such scaffolds leading to error up to three edges). Once scaffolds start to have more than approximately 20 genes, the error becomes consistently low (Fig. 4.4a). The placement

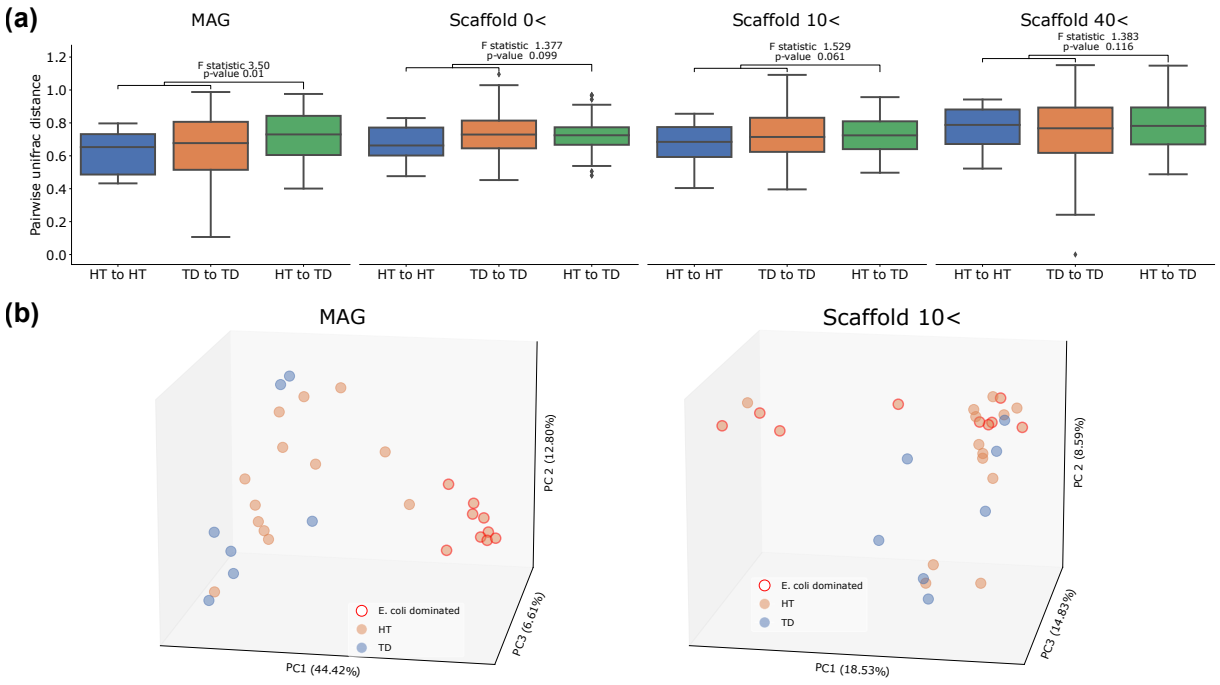


Figure 4.5. Results on the real TD metagenomic dataset. (a) Distribution of UniFrac distances among pairs of samples within HT or TD group and across the groups, using MAGs and scaffolds with more than 0, 10, or 40 marker genes present. F-statistic and p-values are calculated using PERMANOVA test. (b) The PCoA visualization of microbiome profiles of samples using MAGs and Scaffolds with more than 10 marker genes, highlighting samples known to be dominated by *E. coli*.

accuracy for scaffolds that contains 30 to 40 genes is 35%, and 83% can be placed with error no more than 3 edges (Fig. 4.4b). As the number of genes in the scaffold increases, the accuracy also increases; on average, placement error for scaffolds with 50 or more genes is only 1.19 edges, 92% are within three edges of the optimal placement, and the maximum error observed is 12 edges.

Both multiple sequence alignment using UPP and the phylogenetic placement step using APPLES-2 used in the scaffold placement workflow are fast. Running UPP to align all 3318 scaffolds for each gene to the backbone alignment takes 89 seconds in average (lowest 18 and highest 388 seconds) using 6 CPU cores. APPLES-2 takes 2.77 seconds per query scaffold on the backbone tree of size of nearly 10000 species using 28 CPU cores.

4.3.4 Placement of real MAGs and scaffolds onto WoL tree

Next, we study the [262] metagenomic dataset composed of gut microbiomes of 22 patients with Traveller's Diarrhea (TD) and 7 healthy controls (HT). For each subject, we obtain six placement profiles by placing MAGs and scaffolds with five marker occupancy thresholds. We compare two profiles by computing weighted Unifrac distance [142]. We observe a statistically significant difference between intra-group (HT and HT, TD and TD) and inter-group (TD and HD) distances with MAGs (p-value 0.01 using standard PERMANOVA test) (Fig. 4.5a). Using all scaffolds (not including MAGs) intra- and inter-group distances between the samples cannot be distinguished with statistical significance ($p = 0.099$). However, F-statistic increases after filtering out scaffolds with less than or equal to 10 marker genes. Increasing the scaffold filtering threshold to 40 results in a decrease in the F-statistic (Fig. 4.5a), indicating that a large proportion of the signal in the data is lost due to over-filtering. Using placement, we can also visualize MAG- and Scaffold-informed community structures of all samples using a Principal Coordinates Analysis (PCoA) (Fig. 4.5b). MAG-informed community structures provide better delineation of communities dominated by *Escherichia coli* compared to Scaffolds with at least 10 marker genes. Thus, our results show that placing MAGs using APPLES-2 enables inference about community structure of metagenomic samples but the utility of scaffolds is less clear.

4.4 Discussion

We presented APPLES-2: an improved distance based phlogenetic placement tool for inserting new taxa on large phylogenetic trees. Inspired by DCM-like methods [98], our divide-and-conquer approach improved placement accuracy beyond its predecessor APPLES and made it comparable to or better than ML-based tool EPA-NG on single gene data sets. Furthermore, we showed that APPLES-2 is even more scalable than APPLES, reducing running time and memory consumption, and can achieve high accuracy on diverse multi-gene data sets.

Some of the new features of APPLES-2 increase usability and completeness of the

tool but have limited impact on accuracy and scalability. For example, we implemented BME weighting. However, despite the previous literature suggesting BME weighting is preferable to alternatives [51], we observed that BME is less accurate than the default FM weighting scheme for all data set sizes (Fig. 4.14); ; difference between FM and BME mean error is 0.85 edges on average). Based on these results, we continued to use FM as the default weighting method everywhere but provide BME as a new option to the users. Similarly, using amino acid sequences did not show any improvements, but we enable it for cases when only amino acid data is available. Despite declining opportunities, future changes could seek to further improve the accuracy. For example, at the expense of higher computational cost, one can select centroid sequences for partitions of the backbone MSA via ancestral state reconstruction instead of consensus — a technique used by [12] (also see ancestral k-mers [135]).

Previously, [14] reported that ML-based method pplacer failed to place queries on backbone trees with 5000 taxa or larger in RNASim-VS data set due to a numerical error (infinity likelihood values). We find that re-estimating backbone branch lengths and model parameters using RAxML-8 and inputting the RAxML info file to pplacer causes a bug in pplacer. We overcome this issue by creating a taxtastic package (https://github.com/fhcrc/taxtastic) using Fasttree-2 tree and info file and using this package as the input. Note that creating taxtastic package from re-estimated RAxML-8 tree and info file also produces the aforementioned error. As a result of discovery, we do not perform branch length re-estimation using RAxML-8 in any of our data sets.

While accuracy is typically high, on a minority of queries results of APPLES-2 are far from the correct placement. A reasonable question is whether these highly inaccurate instances can be identified by APPLES-2. While we leave a more elaborate exploration to future work, we have identified several interesting patterns (Fig. 4.6). First, we observe a correlation between APPLES-2's objective function value, the Minimum Least Square Error (MLSE; denoted by Q), and placement error (Fig. 4.6a). In addition, variance of error dramatically increases as Q increases. Even for the same level of Q, selecting marker genes strategically instead of randomly

reduces the placement error. Therefore, Q itself does not seem sufficient to predict the degree of placement error. Note that high MLSE (e.g $Q \geq 1$) does not indicate that APPLES-2 fails to optimize its objective function — APPLES-2 solves the objective problem *exactly* (i.e., is not heuristic). High MLSE can result from sequence data and tree distances being very incompatible. This incompatibility may be due to several reasons such as lack of signal, model violation, and horizontal gene transfer (HGT). Despite its reduced *mean* accuracy, APPLES-2 can still find a good placement for many queries with $Q \geq 1$: Approximately 75% of such queries have at most 3 edges error on the backbone consisting of random marker genes. Secondly, when a query is placed with zero distal and pendant edge length, the placement error is significantly higher than otherwise ($p < 5.5 \times 10^{-13}$, two-sample Wilcoxon test). The average error is 11.74 when both pendant and distal edge length is zero (i.e. when query is placed on an internal node) whereas it is only 2.04 on average when pendant edge length is larger than zero (Fig. 4.6b). We have also noticed that erroneous placements with zero pendant edge lengths are more prevalent in the query sequences with fewer genes. Out of 15 occurrences of this pattern, 13 are found in test cases with 10 marker genes in the backbone. Thus, users of APPLES-2 should be skeptical of the placements with zero pendant and distal branch length and/or high MLSE error (which APPLES-2 outputs). A warning is produced by APPLES-2 when such placement are produced. In future work, these features can be used to develop a predictive value indicating possible errors in placement.

Our studies on microbial data showed that APPLES-2 can phylogenetically place and hence identify genome-wide shotgun data with promising accuracy after they are assembled. Using both simulated and real data sets, we showed that assembled genomes (MAGs) can be placed on the species tree with great accuracy. Patterns are more intricate for scaffolds: On real data, scaffolds with very few (as few as one) or large number of marker genes (40) were insufficient to portray the community structure of the metagenomic sample. Filtering scaffolds with fewer than ten marker genes provided the optimal signal-to-noise ratio, despite being inferior to MAGs. On simulated microbial data, the accuracy tends to be low on small scaffolds with few

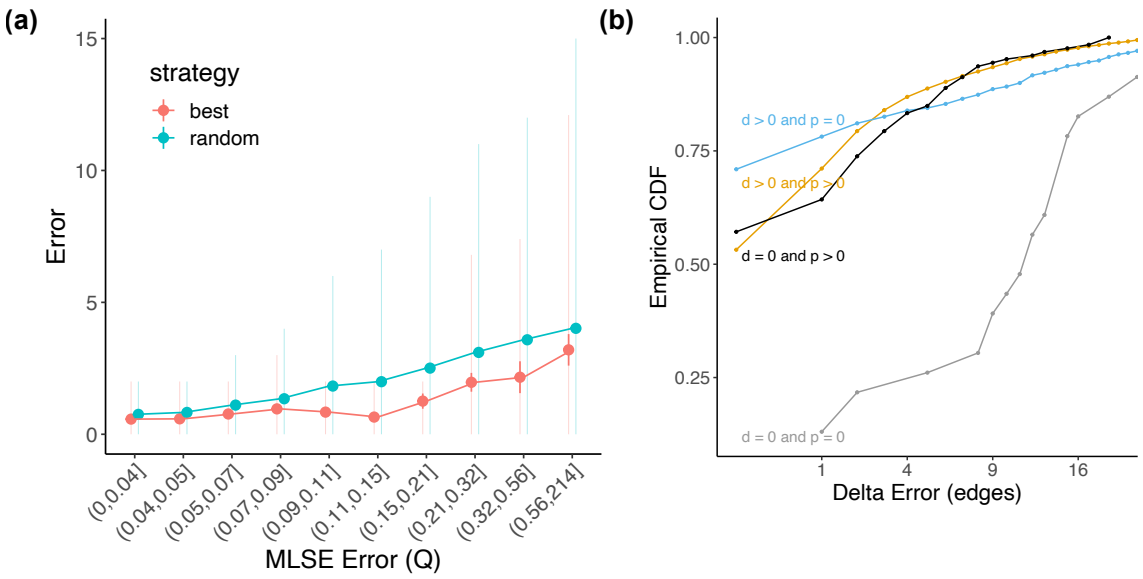


Figure 4.6. Detecting erroneous placements. Results are based on 18879 queries in WoL-random dataset. (a) The relationship between optimized MLSE objective function (Q) and the error. Dots show mean, thick error bars show standard error, and light error bars show the central 80% range. (b) Empirical CDF of placement error with distal (d) and/or pendant (p) edge equal to zero. Queries with zero objective function (i.e. queries that have an exact match to a backbone species) are omitted.

genes but improve for scaffolds that have moderately large number of marker genes. Besides their reduced numbers of genes, scaffolds present several challenges that may contribute to their lower accuracy: (*i*) In comparison to assemblies, scaffolds in a metagenomic sample are more prone to assembly errors and chimeras. (*ii*) Genes located on the same syntenic block have similar gene trees, which can introduce a bias in the placement. Thus, factors such HGT and may have a bigger impact on scaffolds. (*iii*) Even when a scaffold has many genes, it may not include the best marker genes; i.e., those genes with maximum signal and concordance to the species tree.

Our results clearly showed that the choice of genes matters. While a random selection of 25 marker genes were adequate for placing queries in most cases, a targeted gene selection strategy outperformed random selection (e.g., $p = 6.6 \times 10^{-13}$ for 25 marker genes, two-sample Wilcoxon test). The results indicate that certain marker genes serve better at predicting location of a query species in the backbone tree. This observation leads to two related questions. Given fully

assembled genomes, should we use all or a subsample of available genes? Our data supports the idea that using a subset of genes has very similar accuracy to using all available genes. However, a more fruitful approach may be weighting genes (or even sites within genes) differently to further improve accuracy. Such a goal seems amenable to machine learning approaches that can learn optimal weights.

The second question is how to handle scaffolds from metagenomic assemblies, which include only a handful of genes. There are always more scaffolds with few genes than those with many genes. Thus, requiring a large number of genes would reduce the number of scaffolds placed, which has the potential to reduce the accuracy of downstream analyses. Our results indicate scaffolds with a modest number of genes (e.g., with 30 or more) are enough to place them phylogenetically. But the vast majority of scaffolds have fewer than 15 marker genes, and *some* of these *can* be placed accurately. We leave it to future work to design a more principled framework for deciding which scaffolds can be placed accurately and which cannot. We also leave to the future work to answer a more challenging question: for downstream applications, is it better to place a few scaffolds that have many genes (or perhaps binned contigs) with high confidence or is it better to place all or most scaffolds with lower confidence hoping that noise will be overcome by the large number of placements? Answering these questions requires careful experimental procedures that are outside the scope of the present study.

While in this paper we focused on applications of APPLES-2 to microbiome data, our earlier work has demonstrated the utility of distance-based placement for assembly-free and alignment-free identification of genome skims [14]. While reference sets available for genome skimming are not currently large enough to challenge APPLES in terms of scalability, the divide-and-conquer step in APPLES-2 may lead to increase accuracy. Essentially, the divide-and-conquer mechanism will allow building reference databases that include genome skims from very diverse set of organisms (e.g., all insects) without reducing accuracy due to high levels of divergence. We leave the exploration of such applications and the choice of best thresholds for genome skimming to future work.

Finally, in this paper, we focused on single query placement and observed that given multiple marker genes, APPLES-2 can insert a new genome into backbone tree with high accuracy. These results open up an exciting opportunity. By spending less computational budget than *de novo* phylogenetics, successive insertion of genomes can enable expanding the existing large microbial phylogenies [e.g., 263] to contain hundreds of thousands of sequences. Future work should explore the best pipelines for achieving this goal.

4.5 Acknowledgement

Chapter 4, in full, is a reprint of the material as it appears in "Balaban, M., Jiang, Y., Roush, D., Zhu, Q., & Mirarab, S. (2021). Fast and Accurate Distance-based Phylogenetic Placement using Divide and Conquer. Molecular Ecology Resources, (March), 1–15". The dissertation author was the primary investigator and first author of this paper.

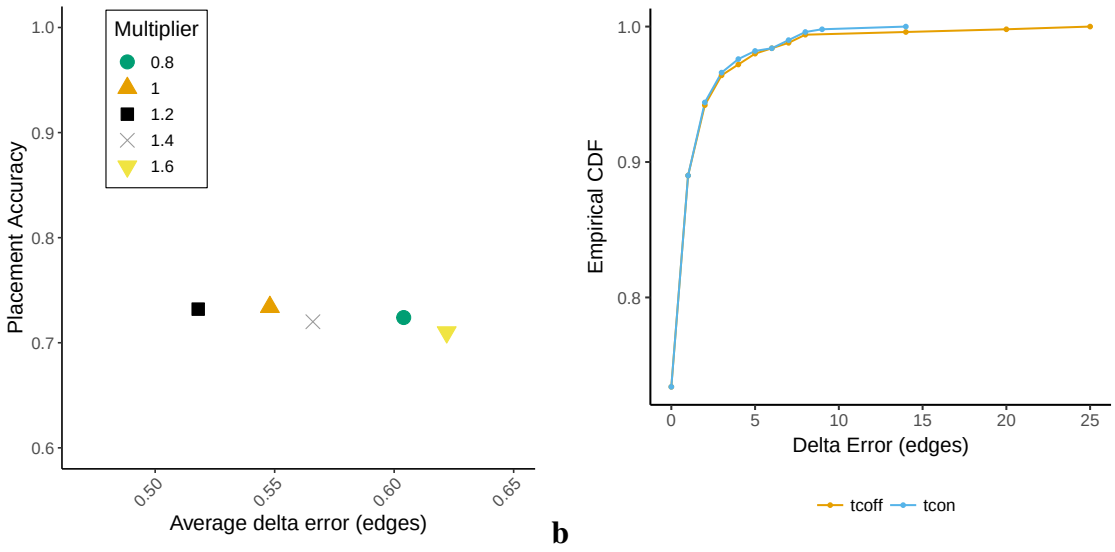


Figure 4.7. Clustering the backbone using TreeCluster. a) Exploring the multiplier for TreeCluster maximum diameter threshold. Results are based on 500 placements on WoL species tree using APPLES-2. Placement accuracy does not change as this threshold changes and the average error increases slightly if other values are used. b) Using TreeCluster to divide the backbone tree into smaller subsets (shown as *tcon* in the figure) not only improves the running time but also slightly improves the placement accuracy compared to using APPLES-2 on the full backbone tree (shown as *tcoff* in the figure) .

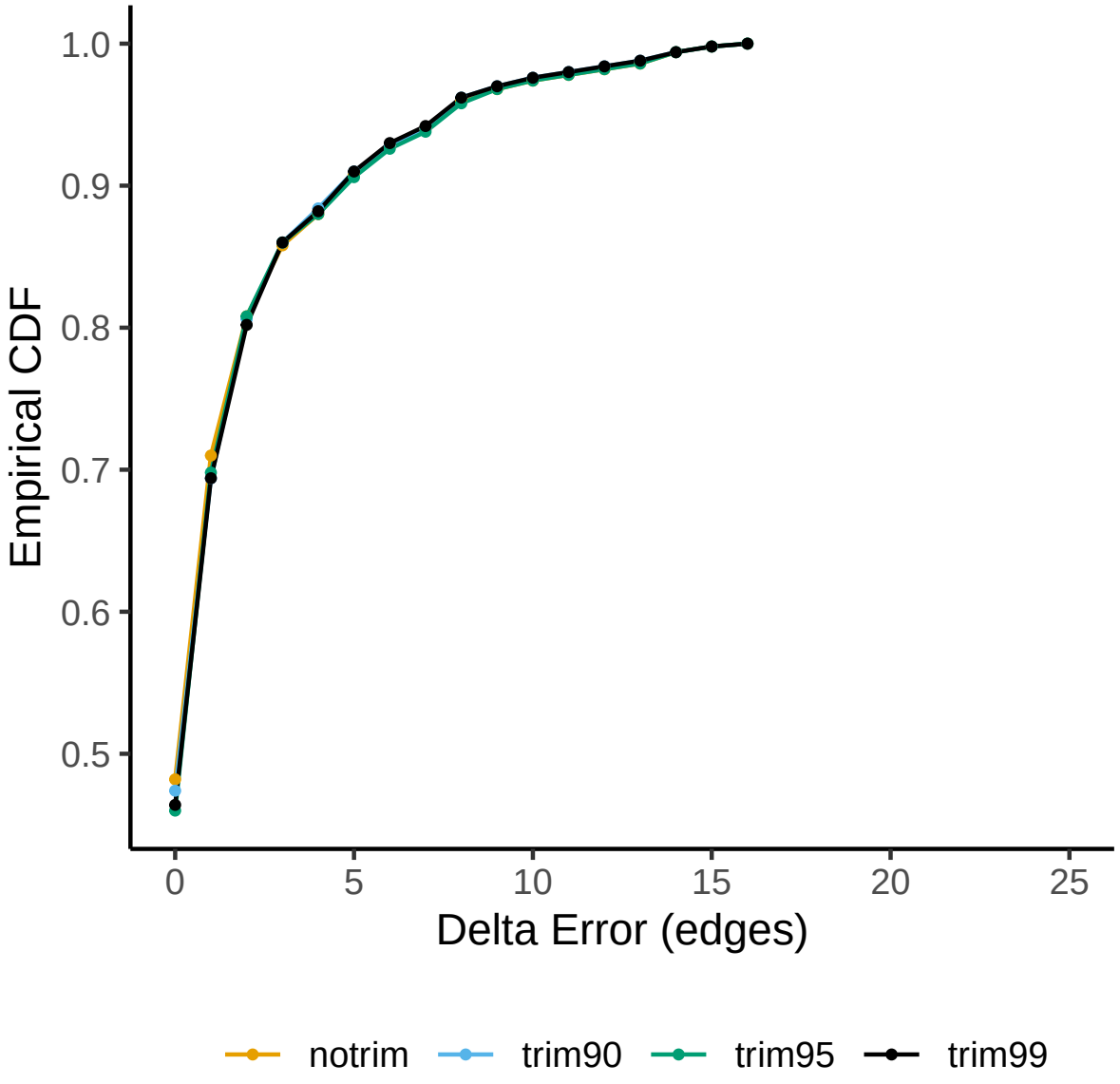


Figure 4.8. Impact of alignment trimming. Results are based on 500 placements on WoL backbone tree using APPLES. Trimming sites with 90, 95, or 99 percent gaps or more does not dramatically impact placement accuracy compared to using untrimmed alignment. Trimming reduces accuracy only slightly. For improved speed and memory usage, we therefore trim the sites with 95% or more gaps.

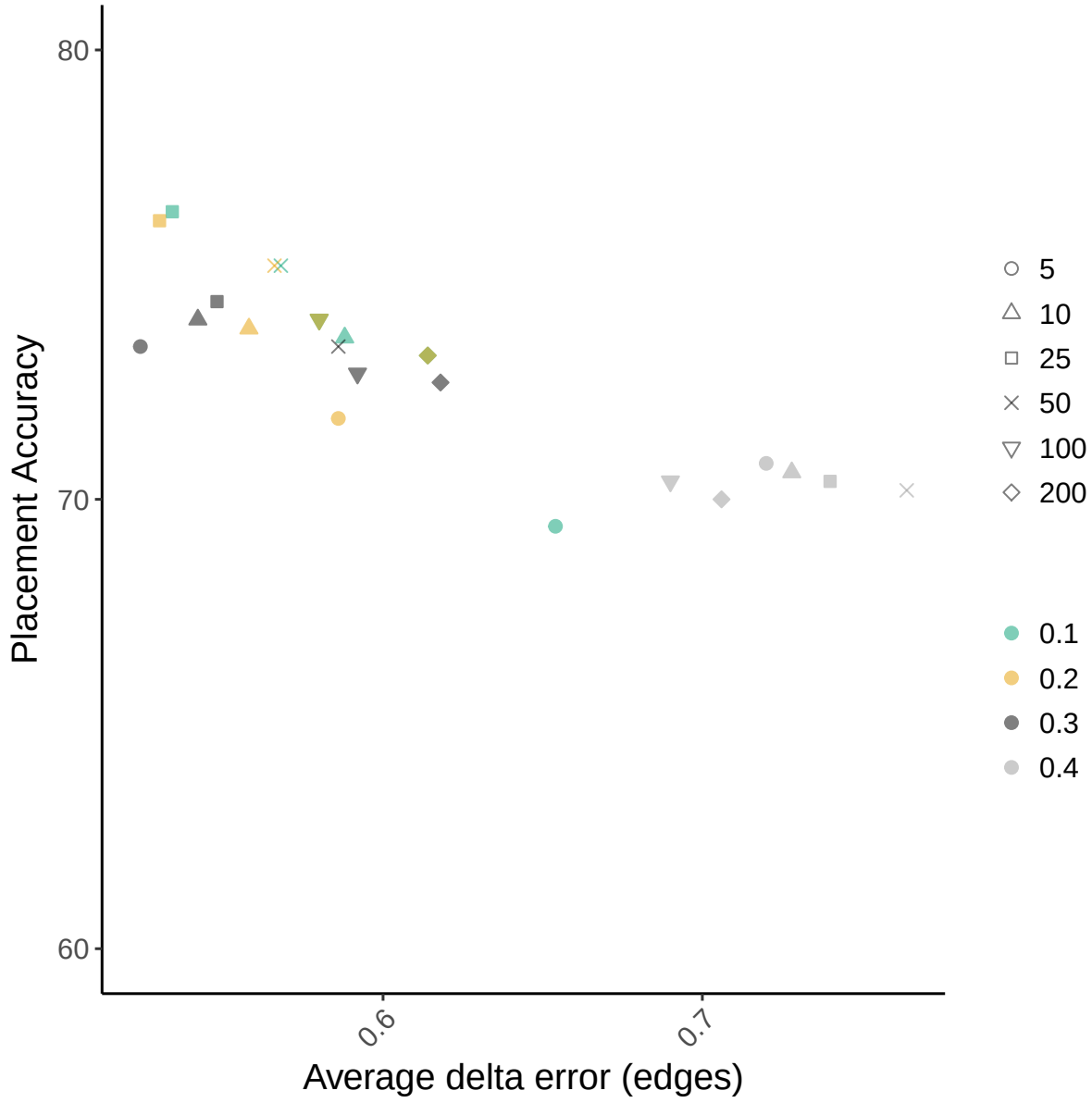


Figure 4.9. Exploring the best choice of d_f and b parameters. Results are based on 500 placements on WoL data set using APPLES-2. Color indicates d_f parameter and shape indicates b parameter.

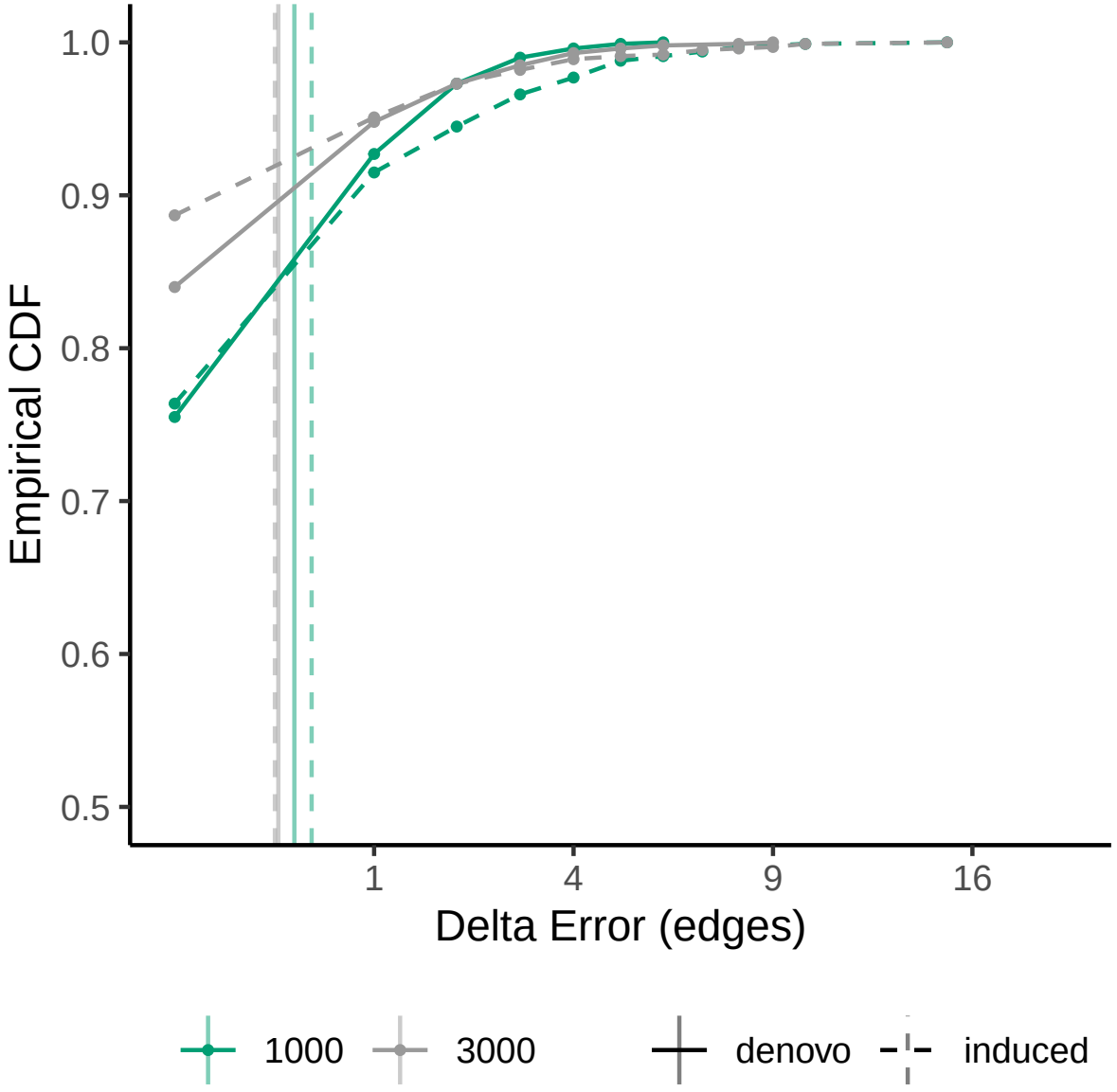


Figure 4.10. De-novo versus induced backbone. For $n = 3000$, delta error is calculated after pruning the placement tree to species in the $n = 1000$ tree and queries.

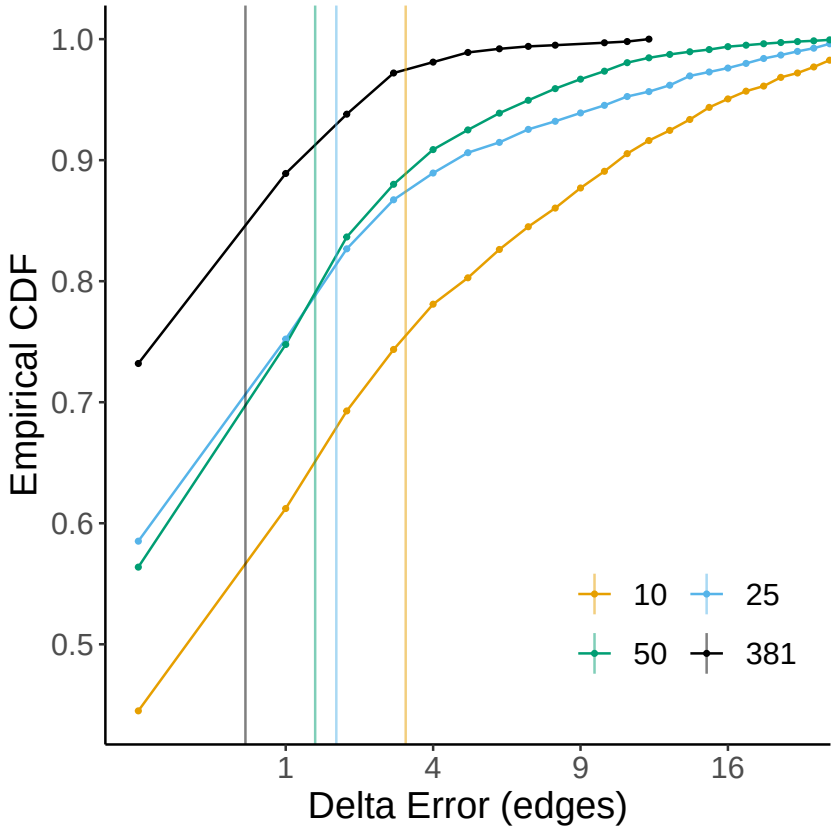


Figure 4.11. Placement accuracy vs size of the marker gene set coupled with random selection strategy. As the number of genes increases, average error decreases consistently.

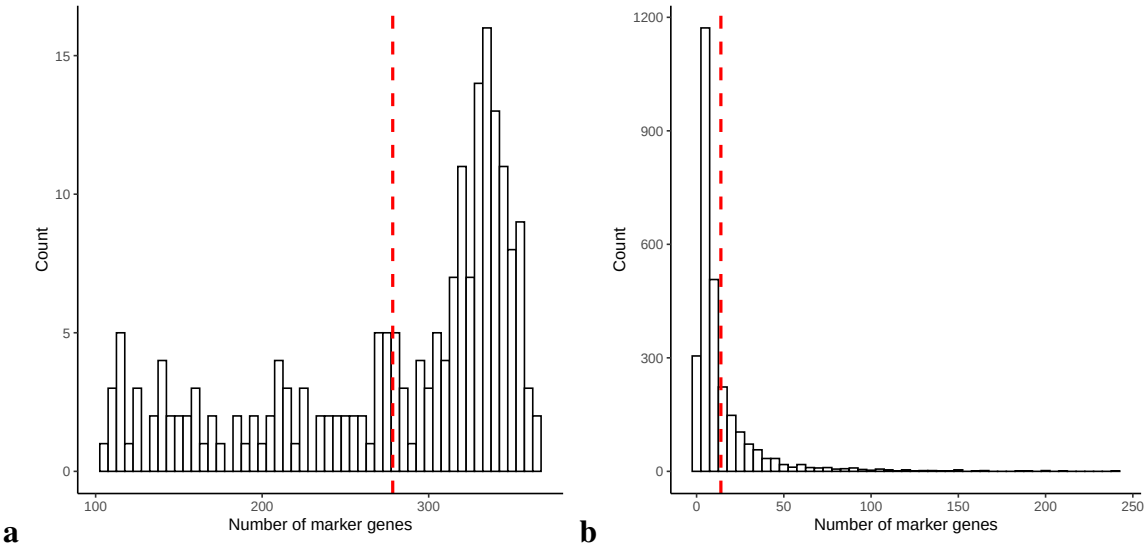


Figure 4.12. Distribution of the number of marker genes in assemblies and scaffolds. Red dashed line shows the mean. Left: assemblies. Right: scaffolds. Note that most scaffolds have relatively few genes.

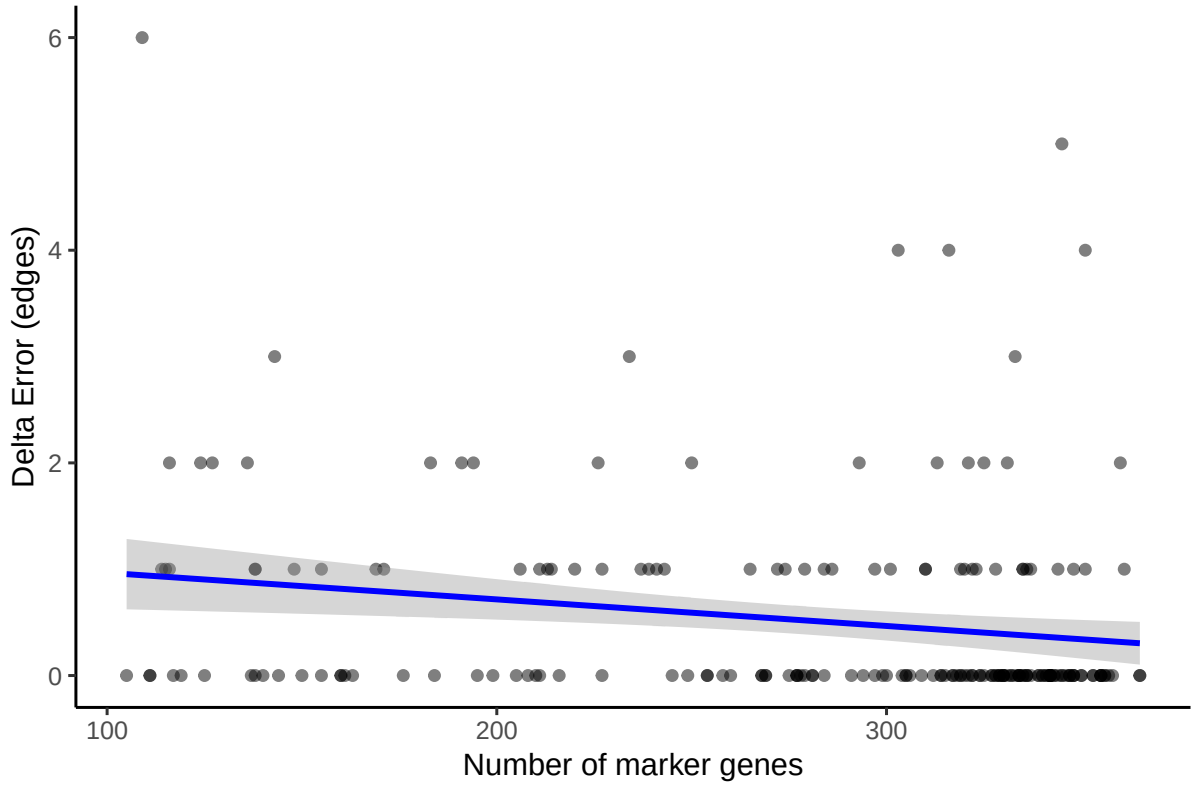


Figure 4.13. Number of marker genes vs placement error for assemblies. There is a weak but statistically significant correlation between the number of genes and error for placing assemblies.

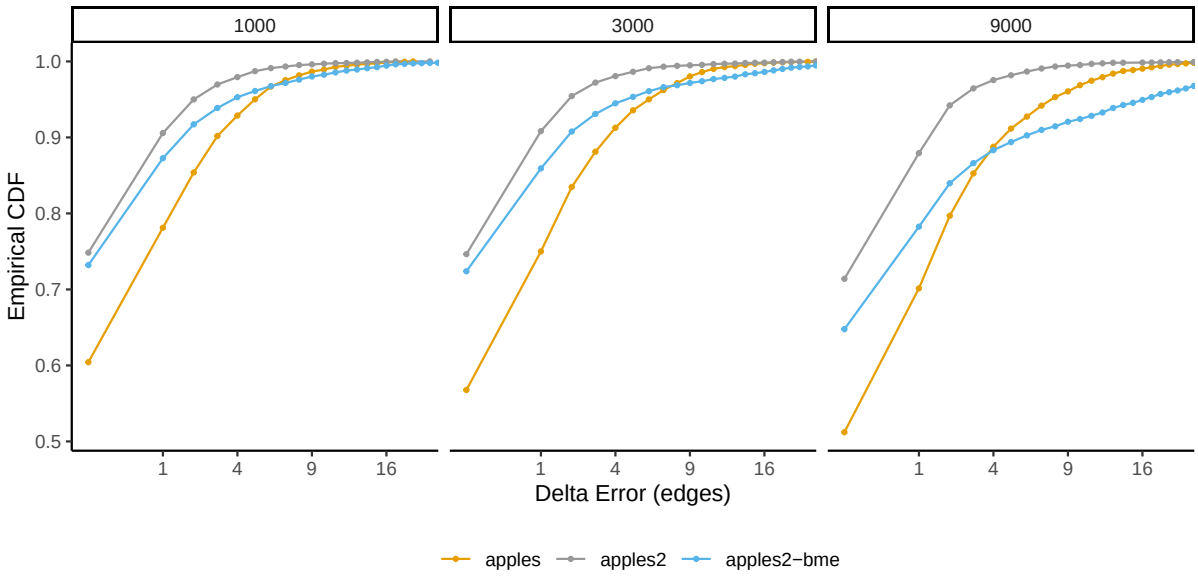


Figure 4.14. We compare balanced minimum evolution (BME) weighting to Fitch-Margoliash (FM), which is the default weighting scheme in APPLES and APPLES-2.

Chapter 5

Phylogenetic double placement of mixed samples

Consider a simple computational problem. The inputs are (i) the set of mixed reads generated from a sample that combines two organisms and (ii) separate sets of reads for several reference genomes of known origins. The goal is to find the two organisms that constitute the mixed sample. When constituents are absent from the reference set, we seek to phylogenetically position them with respect to the underlying tree of the reference species. This simple yet fundamental problem (which we call phylogenetic double-placement) has enjoyed surprisingly little attention in the literature. As genome skimming (low-pass sequencing of genomes at low coverage, precluding assembly) becomes more prevalent, this problem finds wide-ranging applications in areas as varied as biodiversity research, food production and provenance, and evolutionary reconstruction. We introduce a model that relates distances between a mixed sample and reference species to the distances between constituents and reference species. Our model is based on Jaccard indices computed between each sample represented as k-mer sets. The model, built on several assumptions and approximations, allows us to formalize the phylogenetic double-placement problem as a non-convex optimization problem that decomposes mixture distances and performs phylogenetic placement simultaneously. Using a variety of techniques, we are able to solve this optimization problem numerically. We test the resulting method, called

MISA, on a varied set of simulated and biological datasets. Despite all the assumptions used, the method performs remarkably well in practice.

5.1 Introduction

Comparing a set of unassembled reads sequenced from a *query* biological sample against a *reference* library of assembled genomes or unassembled reads can reveal much about the query sample. For example, mapping reads to a closely related assembled genome and variant calling enables population genetic analyses. For more diverse collections of species, genomic distances can be estimated, and distances can allow phylogenetic placement [11]. Sample identification at the population level or higher taxonomic/phylogenetic levels is crucial in many applications, such as characterizing biodiversity, studying food provenance, and detecting toxic contamination. When both the reference and the query are unassembled, as is the case for low-pass sequenced genome skims that do not avail themselves to assembly, we can still compute genomic distances [63, 179] even when the coverage is low [209, 230]. These scalable assembly-free methods have the potential to enable large-scale yet cost-effective genome-wide sample identification because they do not require the genome in the reference library to be assembled. Several tools for assembly-free genome comparison have pursued this ambition [e.g., 42, 63, 204, 235, 252, 255]. However, this methodology has to contend with challenges such the presence of contamination [193] and the potential for mixed samples.

Mixed sample identification is the problem of identifying what species are present in a mixed biological sample of unknown origin. While the metagenomics literature has grappled with a similar conceptual challenge, here, we are specifically focusing on Eukaryotic genomes and mixtures of a small number of species with large genomes (only two in this paper). The two problems are quite different. Here, unlike metagenomics, our samples are not a mixture of large numbers of species with small genomes. Instead, we have a mixture of a handful of large genomes (two in this work). Also, unlike microbes, Eukaryotic genomes do not present certain

difficulties, such as an unclear definition of species and horizontal gene transfer.

The ability to identify the constituents of a mixed sample has obvious applications in food provenance where the goal is to detect adulteration. For example, given a herbal food supplement, can we pinpoint the exact ingredients used, as opposed to those advertised? It also is important for many analyses of ecology and biodiversity where cells of multiple species are intertwined in ways that make physical separation difficult or impossible. For example, bee-breads are mixtures of pollen and fungi; understanding the makeup of these mixes can reveal the pollen composition (indicative of floral diversity), which been linked to local land used for farming [53]. Finally, even when biologists aim to obtain pure single-species samples, technical issues can lead to the sequencing of what is, in reality, a mixed sample. Missing these cases can lead to invalid downstream analyses and false conclusions.

A related concept is recent hybridization. Hybrid speciation is abundant both in the wide and in agricultural and industrial use [148]. The genome of a recent hybrid, especially for alloploids, can be modeled similarly to a mixed sample with two constituents. Such recent hybrids are both abundant and consequential. For example, recent hybridization in yeast species has been hypothesized to contribute to the development of lager beer [54], among other food products.

Little is known about the optimal way to identify the constituents of a mixed sample in the scenario we described. When the genomes of constituents are available in a reference library, pipelines based on read mapping can seek to find the signature of the mixture [e.g., 119]. However, as an exact match to constituents is not always present in the reference set, read mapping is not a general solution. Alignment-based methods have been developed to place a single sequence (e.g., a read) on a phylogeny of assembled references [e.g., 18, 153, 164, 221]. More broadly, many methods have been developed for analyzing metagenomic samples [see 157, 159, 213, 254, for benchmarking of these tools]. By treating reads as independent, these methods can potentially be applied to mixed samples, and are routinely used for analyzing metagenomic samples. However, they often require assembled references and do not work

under the assumption that most reads would belong to one or two species. Also, many come with pre-trained libraries of microbial references. Thus, they are not designed for solving the Eukaryotic mixture problem.

In this paper, we formulate the mixture analysis without exact matches in the reference library as the solution to a "deconvolution" optimization combined with a phylogenetic placement problem. Sample identification for single-species samples without an exact match is possible through phylogenetic placement [11], a methodology that can handle unassembled genome skims for both the reference and the query. To extend phylogenetic placement to mixed samples, we develop a model for decomposing distances between a mixed sample and reference species into distances of its constituent parts to reference species. We present several theoretical results under the model, including results showing that a mixed sample has its minimum possible distance to both of its constituents. Using this model and a non-convex optimization problem, we develop a method for simultaneous deconvolution and phylogenetic placement of samples. Our method, called MISA, is the first to try this kind of analysis and shows promising results on extensive simulation analyses on mixed samples and a real hybridization dataset.

5.2 Approach

5.2.1 Model

Assumptions and definitions

Our model makes several assumptions. (*i*) Each genome (skim) consists of n unique k-mers. We represent a genome A with the set S_A of its fixed-length k-mers. (*ii*) For two constituent genomes A and B, the mixture M includes all k-mers of both genomes: $S_M = S_A \cup S_B$. (*iii*) Evolution is modeled using the time-reversible [105] model, where each position mutates to other positions independently and identically. Evolution occurs along a phylogenetic tree T with genomes as leaves and branch lengths measured in the unit of the expected number of substitutions per position. Let d_{ij}^T be the path length between two nodes i and j in T.

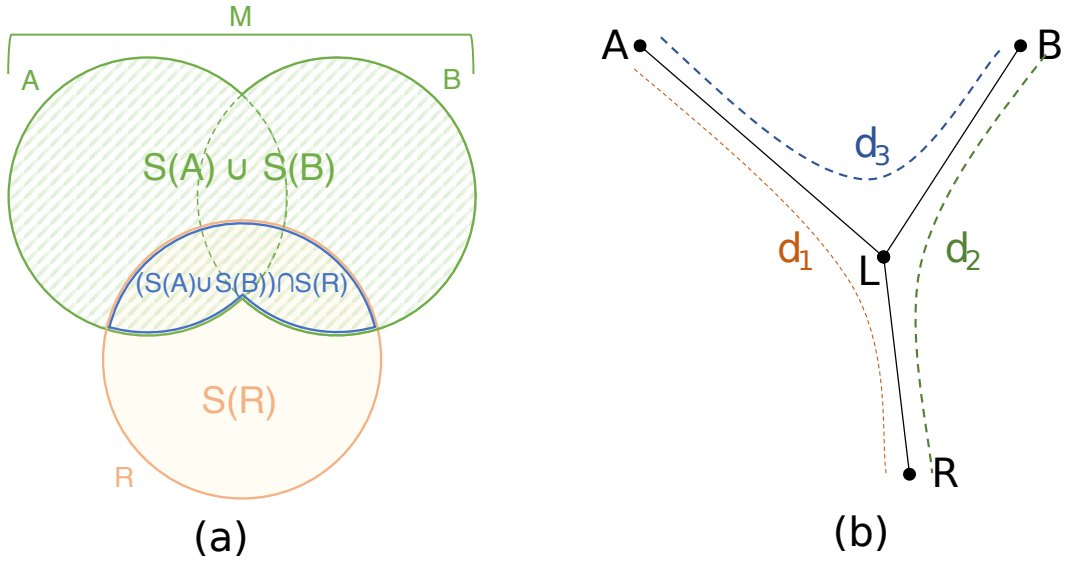


Figure 5.1. (a) Venn diagram showing the intersection of k-mers of mixture M and R. (b) Distances between Mixture constituents A and B and a third genome R. The distance between ancestral genome L and extant genome A is $(d_1 + d_3 - d_2)/2$.

Then, according to the 105 model, the probability of *observing* a change along the branch is $\delta_{ij} = h(d_{ij}^T) = \frac{3}{4}(1 - e^{-4/3 d_{ij}^T})$. We use the h function and its inverse h^{-1} to translate between phylogenetic distance and probability of substitution (i.e., expected hamming distance). (*iv*) We compare the mixture versus a reference genome, referred to by R. For three genomes A, B, and R, let $L(A,B,R)$ (shorthanded to L when clear) be the only node with degree three in T when restricted to these three genomes (Figure 5.1). We define $d_1 = d_{AR}^T$, $d_2 = d_{BR}^T$, and $d_3 = d_{AB}^T$, and let $\delta_i = h(d_i)$. Note that by additivity of tree distances, $d_{LA}^T = (d_1 + d_3 - d_2)/2$ (d_{LB}^T and d_{LR}^T can be written similarly). Also, tree distances (d_1, d_2, d_3) conform to the triangle inequality. (*v*) We assume $d_1 + d_2 + d_3 < 2$, which will enable further approximations. Note that a total branch length of 2 is very high, corresponding to an expectation of two substitutions *per site*. Thus, the assumption is reasonable. (*vi*) $S_A \cap S_B \cap S_R \subset S_L$. Under the assumption (*v*), it is exceedingly unlikely for a k-mer to be present in A, B, and R but not in L (see Figure 5.6).

Recall that the *Jaccard index* J is a similarity measure between two sets defined as the ratio of their intersection to their union. 63 used the Jaccard index J_{AR} of sets S_A and S_R to

estimate

$$\hat{\delta}_{AR} = 1 - \left(\frac{2J_{AR}}{1+J_{AR}}\right)^{\frac{1}{k}} \tag{5.1}$$

and 209 later extended this equation to account for low coverage and sequencing error. Their tool, Skmer, computes the Jaccard index and k-mer frequencies and uses these values to estimate δ directly from reads, accounting for coverage (as low as $^1/_8\times$) and error.

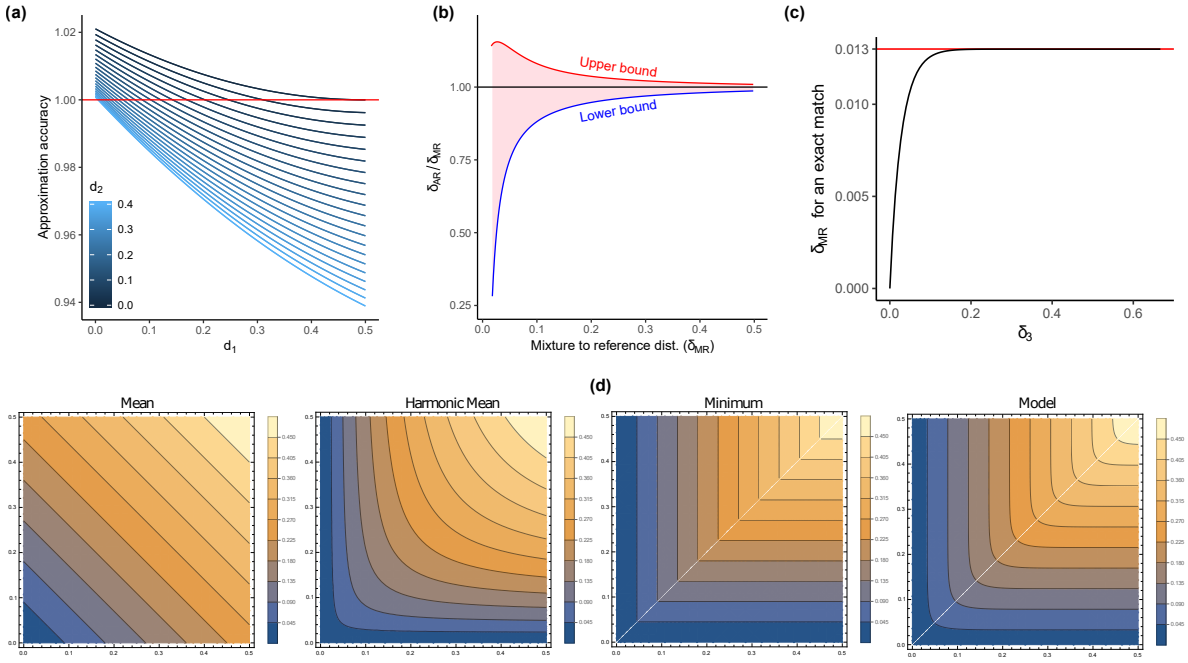


Figure 5.2. Demonstration of model properties. (a) We show $\left(1-h(\frac{d_1+d_2-d_3}{2})\right)\left(1-h(\frac{d_2+d_3-d_1}{2})\right)\left(1-h(\frac{d_3+d_1-d_2}{2})\right)$ divided by $=1-h(\frac{d_1+d_2+d_3}{2})$ and for a set of d_1 and d_2 values (red: no approximation error). We set d_3 to its median value according to triangle inequality ($\max(d_1,d_2)$) but see Figure 5.7 for other choices. (b) y-axis shows the distance between A and R relative to distance between M and R, assuming R is more similar to A than B. We show the bounds of Proposition 2. (c) The fast convergence of $\hat{\delta}_{MR} = 1-(\frac{3-(1-\delta_3)^k}{2})^{-1/k}$ to its upper-bound for $k=31$. (d) The value of $\hat{\delta}_{MR}$ shown as contours versus distance of constituents (x: δ_1 and y: δ_2) with $\delta_3 = \max(\delta_1,\delta_2)$ according to our model (Eq. 5.4) is shown in the rightmost plot. The other three plots compare this model to three simpler models: arithmetic mean, harmonic mean, and minimum. Note that $\hat{\delta}_{MR}$ is close to the minimum value and to a lesser extent to the harmonic mean, but not to the arithmetic mean.

Formulation

The Jaccard-based methodology does not easily translate to mixed samples. Let J_{MR} be the Jaccard index of a mixed sample M and a reference genome R. We can easily compute J_{MR} but cannot translate it to a distance (akin to Equation 5.1) in an obvious way. A more complex formulation is needed. Note that (Figure 5.1a)

$$J_{MR} = \frac{|S_A \cap S_R| + |S_B \cap S_R| - |S_A \cap S_B \cap S_R|}{|S_A \cup S_B \cup S_R|} = \frac{|S_A \cap S_R| + |S_B \cap S_R| - |S_A \cap S_B \cap S_R|}{3n - |S_A \cap S_R| - |S_B \cap S_R| - |S_A \cap S_B| + |S_A \cap S_B \cap S_R|}. \tag{5.2}$$

A k-mer is shared between A and B only if no mutation occurs in any position on the k-mer. In expectation, $|S_A \cap S_R| = n(1-\delta_1)^k$, $|S_B \cap S_R| = n(1-\delta_2)^k$, and $|S_A \cap S_B| = n(1-\delta_3)^k$. Moreover, due to our Assumption (*vi*), a k-mer is shared between all three genomes A, B, and R only if that k-mer is present in L. Therefore, in expectation:

$$|S_A \cap S_B \cap S_R| = n(1-\delta_{LA})^k(1-\delta_{LB})^k(1-\delta_{LR})^k = n\Big((1-h(\frac{d_1+d_3-d_2}{2}))(1-h(\frac{d_2+d_3-d_1}{2}))(1-h(\frac{d_1+d_2-d_3}{2}))\Big)^k \tag{5.3}$$

We define $\hat{\delta}_{MR} = 1 - \left({}^{2J_{MR}}/_{(1+J_{MR})}\right)^{1/k}$ (for skims, instead of plugging J into Equation 5.1, we can use the the more complex coverage-aware equations of [209]). Note that $\hat{\delta}_{MR}$ is just a mathematical construct without a clear biological meaning. By re-writing J_{MR} in terms of d_1, d_2, d_3, and k, plugging it in this definition, and further simplifications, we derive the following model.

$$\hat{\delta}_{MR} = 1 - \Big(\frac{2}{3-(1-h(d_3))^k}\big((1-h(d_1))^k + (1-h(d_2))^k - ((1-h(\frac{d_1+d_3-d_2}{2}))(1-h(\frac{d_2+d_3-d_1}{2}))(1-h(\frac{d_1+d_2-d_3}{2})))^k\big)^{\frac{1}{k}}$$

Furthermore, under assumption (*v*), $n\big(1-(h(\frac{d_1+d_2+d_3}{2}))^k\big)$ approximates Equation 5.3 well, falling within 5% of its value in much of the relevant space (Figure 5.2a). Thus, we further simplify the model to:

$$\hat{\delta}_{MR} = 1 - \Big(2\frac{(1-h(d_1))^k + (1-h(d_2))^k - (1-h(\frac{d_1+d_2+d_3}{2}))^k}{3-(1-h(d_3))^k}\Big)^{\frac{1}{k}} \tag{5.4}$$

All our subsequent results are based on this model.

As expected, in this model, $\hat{\delta}_{MR}$ is a function of d_1, d_2, and d_3. Plotting Equation 5.4 shows (Figure 5.2d) that $\hat{\delta}_{MR}$ resembles the harmonic mean of δ_1 (i.e., $h(d_1)$) and δ_2 much better than their arithmetic mean; moreover, $\hat{\delta}_{MR}$ is quite close to the minimum of δ_1 and δ_2. This observation can be further formalized as a bound.

Proposition 2. *(Proof in Appendix 1) Let $\delta_{AR} \leq \delta_{BR}$. For a fixed value $z = \hat{\delta}_{MR}$, the lower-bound for δ_{AR} is $1 - \left(\frac{3}{2}\right)^{\frac{1}{k}}(1-z)$ and its upper-bound u is given by $\frac{4(1-u)^k - 2(1-2u)^k}{3-(1-2u)^k} = (1-z)^k$.*

Thus, the distance between the reference genome and the closer of constituent samples bounds the distance to the mix (Figure 5.2b). We utilize this lower bound in our algorithm (described later).

Reference-guided deconvolution.

Given the Skmer distance $\hat{\delta}_{MR}$, our aim is to estimate the distance between R and constituents A and B of M; i.e., to estimate d_1 and d_2 (and less crucially, also d_3). Given these estimates, we can place the mix on two branches of the phylogeny using distance-based phylogenetic placement [e.g., 11]. The challenge is that d_1, d_2, and d_3 are not observed directly from the data, and all three impact our single observation $\hat{\delta}_{MR}$. Thus, we have to deconvolute $\hat{\delta}_{MR}$ to constituent parts. However, the problem has infinitely many solutions, including trivial ones like $h(d_1) = h(d_2) = \hat{\delta}_{MR}$, $d_3 = 0$.

Our main insight is that although the problem is underdetermined when only one data point (R) exists, given multiple $\hat{\delta}_{MR_i}$ values and a phylogenetic tree, we can impose constraints on the values of these $\hat{\delta}_{MR_i}$ variables. The simplest example of such implicit constraints is the triangle inequality (e.g., given $\delta_{R_1R_2} = 0.1$, two inequalities must hold: $0.1 \leq |\delta_{AR_1} - \delta_{AR_2}|$ and $\delta_{AR_1} + \delta_{AR_2} \leq 0.1$). Moreover, the correctly deconvoluted values should be close to additive (i.e., should fit a tree). Our approach is to define a set of constraints on deconvoluted distances based on the model and to seek a combined deconvolution and placement solution that minimizes

deviations from additivity. We then define an optimization problem to find the optimal value for all the variables.

5.2.2 Phylogenetic double-placement

We extend the distance-based phylogenetic placement problem of [11] to introduce the distance-based phylogenetic double-placement problem. Let an unrooted tree T be a weighted connected acyclic undirected graph with leaves denoted by $\mathscr{R} = \{R_1 \cdots R_m\}$. Placement of a query sequence can be represented by the placement edge as well as distal and pendant edge lengths of the added taxon (Figure 5.8). Given the phylogenetic tree T, a mixture M of A and B, and Jaccard-driven estimates of $\hat{\delta}_{MR_i}$, we aim to find the optimal position of A and B on T. We represent the solution as two placement trees P and Q, each obtained by adding a new leaf (A or B) to a specific position on a branch in T with a pendant edge length.

Mixture of known species

We start by results concerning cases where the mixture is of two references present in the reference phylogeny. In this case, luckily, the smallest $\hat{\delta}_{MR_i}$ values readily identify the constituents.

Proposition 3. *(Proof in Appendix 1) When $A \in \mathscr{R}$ and $B \notin \mathscr{R}$, $\inf_{R_i} \hat{\delta}_{MR_i} = A$. Also, $\hat{\delta}_{MA} = 1 - \left(\frac{2}{3-(1-\delta_3)^k}\right)^{1/k}$*

Corollary 2. *Without loss of generality, Let $\inf\left\{\hat{\delta}_{MR_i} | R_i \in \mathscr{R}\right\} = R_1$ and $\inf\left\{\hat{\delta}_{MR_i} | R_i \in \mathscr{R} \setminus \{R_1\}\right\} = R_2$. If $A \in \mathscr{R}$ and $B \in \mathscr{R}$, $A, B \in \{R_1, R_2\}$ and $\hat{\delta}_{MR_1} = \hat{\delta}_{MR_2} = 1 - \left(\frac{2}{3-(1-\delta_3)^k}\right)^{\frac{1}{k}}$.*

Thus, $\hat{\delta}_{MA}$ and $\hat{\delta}_{MB}$ values are expected to be identical to a function of k and δ_3 (unknown). Luckily, regardless of δ_3, this value has a constant upper-bound with a small value $1 - (3/2 - 1/2(1 - 3/4)^k)^{-1/k}$. Moreover, as δ_3 increases, $\hat{\delta}_{MR}$ quickly approaches this upper bound (Figure 5.2d). Therefore, when the reference library includes both constituents, one can simply find the smallest two among all $\hat{\delta}_{MR_i}$ and the identification problem is solved. Moreover, in this scenario, $\hat{\delta}_{MR_i}$ should not exceed a small constant (e.g., 0.013 for $k = 31$).

Mixture of unknown species

The most interesting case, which necessitates phylogenetic placement, is when the mixture is of two species absent from the reference set. For distance-based phylogenetic placement, we use the Ordinary Least Squares (OLS) criterion [35]. Following the standard formulation, we seek the solution that minimizes:

$$\sum_{i=1}^{n}(h^{-1}(\delta_{Ai})-d_{Ai}^{P})^2+\sum_{i=1}^{n}(h^{-1}(\delta_{Bi})-d_{Bi}^{Q})^2\ .$$

If δ_{Ai},δ_{Bi} were known, due to the independence of the two placements in this formulation, the problem could be considered as two single-species placement problems. However, for a mixed sample we do not have δ_{Ai},δ_{Bi}. Instead, we consider δ_{Ai},δ_{Bi} as variables and approach the determination of these variables and placement of mixtures constituents as a simultaneous solution of the deconvolution problem and the placement problem.

Least Squares Phylogenetic Double-Placement:

Input: A backbone tree T on $\mathscr{R}$ and a vector with elements $\hat{\delta}_{Mi}$, each giving the Jaccard-based distance between M and a species $i\in\mathscr{R}$;

Output: Vectors x_*^A,x_*^B, variable x_3, and two placement trees P and Q that add A and B on T respectively, such that:

$$\sum_{i=1}^{n}(x_i^A-d_{Ai}^{P})^2+\sum_{i=1}^{n}(x_i^B-d_{Bi}^{Q})^2 \tag{5.5}$$

is minimized, subject to:

$$\hat{\delta}_{Mi}=1-\Big(2\frac{(1-h(x_i^A))^k+(1-h(x_i^B))^k-(1-h(\frac{x_i^A+x_i^B+x_3}{2}))^k}{3-(1-h(x_3))^k}\Big)^{\frac{1}{k}} \tag{5.6}$$

This problem formulation can be extended to multiple query sequences to define a phylogenetic multi-placement problem. In this paper, we only focus on the special case of double placement. We are faced with a non-convex optimization problem with many variables.

5.2.3 Solving the non-convex optimization problem

For each constituent, the number of possible placement edges is $2n-3$, and for each placement edge, one distal and one pendant edge length characterize a placement tree (Figure 5.8). We encode these two lengths as two more variables. To solve the optimization problem, we iterate over all pairs of edges (in parallel), and for each pair, find $2n+1$ distance variables and four distal and pendant edge length variables that minimize the optimization score. At the end, we return the placement with the minimum least square error across all $\binom{2n-2}{2}$ placements. All the d^T_{ij} values are precomputed using a simple dynamic programming.

For a fixed pair of edges, our optimization problem has a quadratic objective function and non-linear constraints. We solve the problem numerically, using the trust region method of [40], as implemented in the SciPy optimize module [236]. For this numerical optimization solutions to converge, several difficulties need to be addressed.

Jacobian and Hessian.

Providing the Jacobian and Hessian of the optimization score (Eq. 5.5) and non-linear constraints (Eq. 5.6) to the numerical solver is crucial in achieving convergence. To be able to compute derivatives of Equation 5.6 analytically and to help achieve convergence, we had to adopt two further approximations. Firstly, having $h()$ on the right-hand side (RHS) is a challenge. To deal with this difficulty, on the left hand side (LHS), we replace $\hat{\delta}_{Mi}$ with $h^{-1}(\hat{\delta}_{Mi})$, and on RHS, we replace all $h(x^A_i)$ terms with $h^{-1}(h(x^A_i)) = x^A_i$ (ditto for x^B_i). This approximation is akin to making an infinite sites assumption and is negligible when distances are relatively small (i.e., $h(x)$ is close to identity for x close to 0). Secondly, having a variable x_3, which represents $h(d_3)$, that is shared between all n constraints makes derivations of Jacobian and Hessian difficult and complicate the optimization since constraints cannot be handled independently. We therefore approximate x_3 with $\max\left\{d^T_{uv}, d^T_{u'v'}, d^T_{uv'}, d^T_{u'v}\right\}$ where (u,u') and (v,v') denote the placement edges being tested. These approximations make it relatively easy to derive the Jacobian and Hessian (Appendix 1).

Inequality constraints.

We further impose lower bounds on values of x_*^A and x_*^B according to the upper and lower bounds obtained in Proposition 2. These constraints dramatically reduce the feasible solution space and help with faster convergence.

Initialization and termination.

Trust region method requires a valid initial point (i.e. one that satisfies the constraints). We always initialize pendant and distal edge lengths to 0. Let $\hat{x}_3$ be the constant approximated value of x_3 described previously. For each reference sequence R_i, we initialize one of x_i^A and x_i^B to a value x^0 and set the other to $x^0 + \hat{x}_3$ such that when we plug x_i^A and x_i^B in Eq. 5.6, the constraint is satisfied. This is achieved by $x^0 = 1 - (1 - h^{-1}(\hat{\delta}_{Mi}))((3 - (1 - \hat{x}_3)^k)/2)^{1/k}$. To decide whether x_i^A or x_i^B are set to x^0, we compare d_{iA}^T and d_{iB}^T and choose the smaller.

We use the default termination conditions for the trust region algorithm but limit the maximum number of iterations to 5000. In our preliminary tests, we observed that, in most cases, low residual errors and convergence are obtained in much fewer iterations (e.g., see Figure 5.9).

MISA

We implement our algorithm in a tool called MISA (MIxed Sample Analysis tool). The input to MISA is the vector $\hat{\delta}_{MR_i}$ of distances between the query sample and target species (e.g., computed using Skmer or Mash), the value of k, and the backbone tree. It uses the [105] model to correct phylogenetic distances, uses Treeswift [169] for tree operations, and generates the output in the jplace format [154].

Automatic choice of k.

MISA can suggest a k for a given backbone dataset. To do so, it computes the LSE of the backbone tree with regards to the reference genomes for a set of k values (here, all odd values of $21 \leq k \leq 31$) and picks the k that leads to the the minimum LSE error.

5.3 Experimental Setup

5.3.1 Datasets

Drosophila dataset (Simulated mixture).

We use a set of 14 Drosophila assemblies published by [161] (Table C.1) to evaluate the accuracy of our approach in an ideal setting where the mixed sample consists of the concatenation of the assemblies. We test 20 simulated mixtures of randomly chosen species in three scenarios where none, one, or both of the constituents are present in the reference library.

Columbicola (Lice) dataset (Simulated mixture).

In order to evaluate the accuracy of our method on real genome skimming data, we use a set of 61 genome-skims by Boyd et al. [27] (PRJNA296666), including 45 known Lice species (some represented multiple times) and seven undescribed species. We use randomly subsampled genome-skims of 4Gb. We use BBTools [31] to filter subsampled reads for adapters and contaminants and remove duplicated reads. Then, we create five replicates each containing 20 organisms sampled from the full dataset at random. For each replicate, we simulate five mixtures with A and B chosen uniformly at random. We simulate mixtures by simply combining preprocessed genome-skims of the two constituents. The exact coverage of the genome-skims is unknown but is estimated to range between 4X and 15X by Skmer.

Yeast dataset (Real Hybridization).

In addition to simulated mixtures, we create a dataset of real hybrid yeast species. We select representative genomes for eight non-hybrid Saccharomyces species with assemblies available on NCBI. We also create a second extended dataset where we include seven more species from Genera Naumovozyma, Nakaseomyces, and Candida (see Table S2 for accession numbers). We curate four assembled and two unassembled strains of hybrid yeast species, some of which were previously analyzed by [119]. Unassembled hybrid strains muri [117] and YMD3265 are subsampled from NCBI SRA to 100Mb and filtered for contaminants in the same

fashion as the previous dataset. We do not include strains such as *Saccharomyces bayanus* that are conjectured to be hybrid of three species [134]. For each hybrid species, the hypothesized ancestors are known from the literature [117, 119, 120] and NCBI Taxonomy annotation, and we use these postulated ancestors as the ground truth.

5.3.2 Distance calculation and backbone trees

On all datasets, we compute reference-to-reference and reference-to-query sequence distances using Skmer. To select k, we use the automatic procedure described earlier (i.e., the k with the minimum LSE on the backbone). This procedure chooses $k = 21$ for the two assembly-based datasets (Drosophila and yeast) and $k = 31$ for the skim-based dataset (Columbicola) (Table 5.1). The backbone tree topologies are set to those of previously published phylogenies for Yeast [216, 225], Drosophila [161], and Columbicola [27]. For all datasets, the backbone tree branch lengths are re-estimated (Figure 5.10) by running FastME2.0 [123] on sequence distances according to the [105] model. This branch re-estimation method can produce negative branch lengths. In the case of the yeast dataset, the tree includes one branch with a negative length. In the Lice data, four branches have negative estimated length. In three out of our 25 total replicates in the Lice dataset, the placement distal edge length is negative. These likely reflect errors (like contamination) in the data, and our approach does not model negative length. To remedy this, we set length of negative branches on backbone tree to zero. In addition, one species (called 931 here) contributes disproportionately to the LSE error of the backbone tree compared to other species (Figure 5.11). While this species is suspect (e.g., may be contaminated), we keep it in the analyses.

To create test cases with the query constituents missing from the reference set, we simply remove the constituents from the full backbone tree, but we do not recompute the backbone or its branch lengths.

Table 5.1. Impact of k on the error of the backbone trees. Error is measured using the unweighted least square error (LSE) and [68] weighted least square error (FME). In FME, each squared error term is weighted by the inverse of observed distance squared, reducing the contribution of longer distances. Error for the backbone tree including all 61 species in the dataset is shown for Columbicola. Error for extended backbone tree is shown for yeast.

	(a) Columbicola		(b) Drosophila		(c) Yeast	
	LSE	FME	LSE	FME	LSE	FME
k=21	0.053	4.768	0.0003	0.014	0.0094	0.106
k=23	0.0385	6.4778	0.0005	0.0158	0.0094	0.1133
k=25	0.0275	4.4182	0.0005	0.0193	0.0097	0.1254
k=27	0.0208	3.1422	0.0006	0.0235	0.01	0.1385
k=29	0.0167	2.272	0.0007	0.0278	0.01	0.1508
k=31	0.013	1.5779	0.0008	0.0348	0.0102	0.1641

5.3.3 Evaluation

Evaluation metric.

We quantify the error in each placement by counting the number of branches between the placement found by each method and the correct placement. We report the error for the two placements separately. We simply define the placement with the lower error to be the *primary* placement and the other placement to be the *secondary* placement. Note that the primary/secondary distinction is not made by the tool and is solely used to facilitate the analysis of error in an interpretable way. When constituents are in the reference tree, we also compute the tree distance between the constituent and the placement tree; ideally, this distance should be zero for simulated mixes.

Methods compared.

No existing method can solve the mixture problem as defined here (input: reads from a mixed sample, a tree, and reads from each leaf of the tree; output: two placements on the tree). Thus, we are forced to compare MISA to two control methods.

The simplest alternative method is TOP2: compute the distance of the mixed sample to all reference species and place the query as sister to the species with smallest two distances. By Corollary 2, this method should work well when the constituents are in the reference. Note that

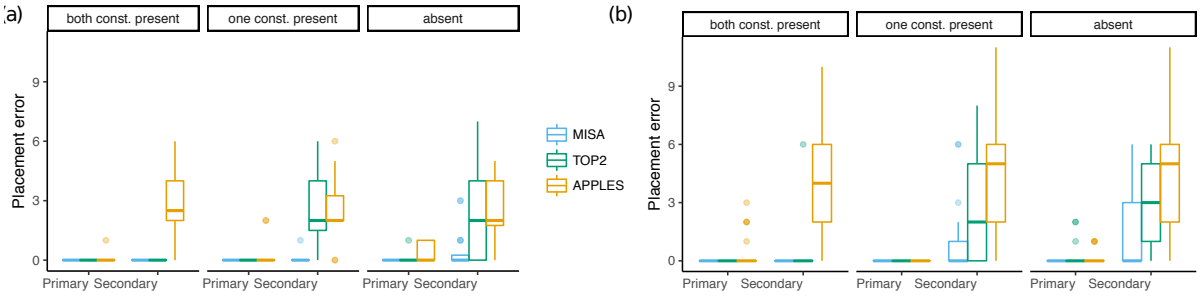


Figure 5.3. Placement error for each method when constituents of simulated mixtures are both present, one present and one absent, or both absent in the reference set. Distributions are over 20 replicates for Drosophila (a), and 25 replicates for Columbicola/Lice (b) datasets. We show the number of branches between each placement and the correct placement (i.e., 0 means perfect accuracy). Because there are two placements, we show the error for both placements, designating the one with lower error as primary and the second one as secondary.

we do not use Corollary 2 in the design of the MISA method. The second alternative to MISA is to pretend the mix is a single-species sample and to perform phylogenetic placement using APPLES [11]; in this scenario, we set both placements to be equal. By definition, APPLES is not trying to get both placements correct; however, we can hope it can place at least one of the two constituents correctly.

5.4 Results

5.4.1 Simulated mixture datasets

Constituents sampled in the reference set

When both constituents are in the reference library, the MISA method perfectly identifies both constituent species both for the assembly-based Drosophila data and assembly-free Lice data (Figure 5.3). In agreement with Corollary 2, TOP2 detects the correct placement in 89 out of 90 placements across the two datasets. In one case on the Columbicola dataset, the secondary placement of TOP2 is wrong by six edges. APPLES is able to correctly place one of the two constituents everywhere except one case (out of 20) for Drosophila and five cases (out of 25) for Lice. APPLES, by design, fails to identify the second species unless the constituents happen to form a cherry.

MISA also produces branch lengths that can be examined. When mixed species are in the library, ideally, the total branch length between output placements and constituents should be zero. For both datasets, the distance between the MISA placement and the correct placement is in all but one case below 0.013 for both primary and secondary placements and is 0.018 in the remaining case (Figure 5.4a).

Examining $\hat{\delta}_{MR}$ values on the Drosophila dataset, as predicted by the theory, $\hat{\delta}_{MA}$ and $\hat{\delta}_{MB}$ are close to the theoretical bound 0.019 for $k = 21$ (always between 0.015 and 0.025; see Figure 5.4b). On the Lice dataset, both $\hat{\delta}_{MA}$ and $\hat{\delta}_{MB}$ are close to 0.013 (theoretical bound for $k = 31$) in all but three cases. In one case, the mixture has close to zero distance to both constituents, one of which is 931; as mentioned before, this species contribute abnormally high levels to the error of the backbone phylogeny (Figure 5.11) and should be treated as suspect. The other outlier is a species, which we call 932, where the distance to mixture is 0.03. This species has the longest terminal branch length in the tree (Figure 5.10). Interestingly, despite not agreeing with the numerical predictions of the model, the 932 species is placed correctly by MISA, while it is not placed correctly using TOP2 (the only case where TOP2 has an error).

Constituents fully or partially missing in the reference set

When one of the constituents is in the reference library, TOP2 finds that species with perfect accuracy (Figure 5.3). However, it cannot accurately find the second constituent; the median error is two edges for both Lice and Drosophila datasets, and it can be as high as eight edges for the Lice dataset. Thus, TOP2 is only partially successful. APPLES similarly performs well for one of the constituents but cannot find the second species. MISA, in contrast, has high accuracy in this scenario. Its primary placements are always correct in both datasets. The secondary placements are correct in 19 of 20 cases in Drosophila dataset. In one replicate, the secondary placement is off by one edge. On Lice dataset, its secondary placements have a median error of zero edges. The error is two edges or less in all but three cases (Fig. 5.3b). One of these outlier cases is the only example in the Lice dataset where the two constituents happen to form a

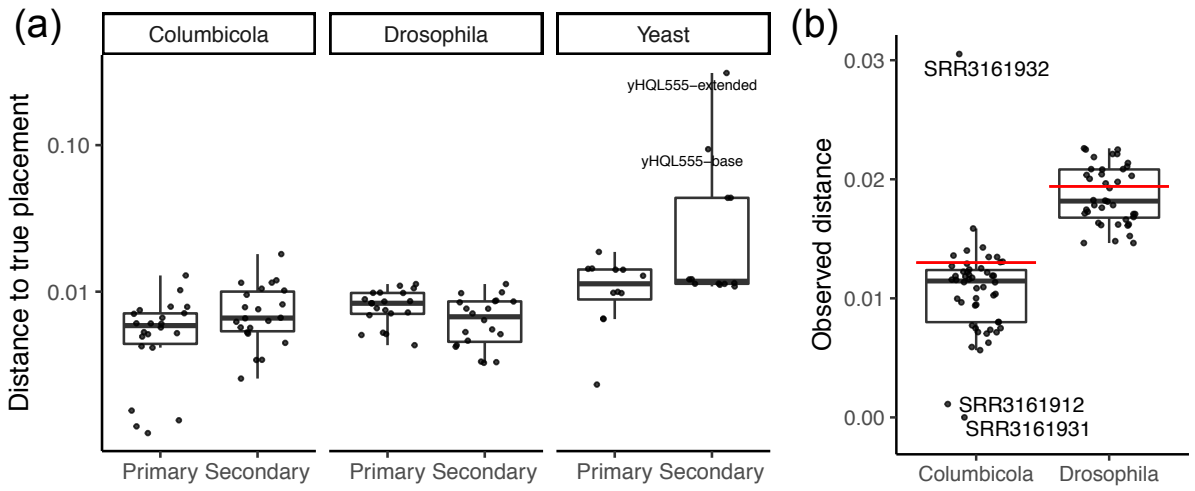


Figure 5.4. (a) For each placement found by MISA, we show the tree distance between the placement and the correct constituent (which, here, is present in the reference set) on Columbicola, Drosophila, and yeast (Saccharomyces) datasets. For yeast, we show results both for small and extended backbone tree. (b) Observed distances between mixtures to constituents (e.g., $\hat{\delta}_{MA}$ and $\hat{\delta}_{MB}$) in Drosophila and Lice datasets. Horizontal red line indicates the value predicted by our model (with k=31 and k=21 respectively). The outlier are labelled.

cherry (i.e., are sister taxa). Thus, they must be both placed on the same branch, but MISA only places one of them on the correct branch. Nevertheless, for partially complete reference sets, MISA greatly outperforms the alternatives.

When both constituents are missing from the set, TOP2 remains surprisingly accurate in finding *one of* the two species (Figure 5.3); its primary placements are correct in all cases except for one replicates of the Drosophila and three replicate of the Lice datasets. However, for the second placement, TOP2 has high error levels (median: two edges for Drosophila and three edges for Lice). APPLES has higher error than TOP2.

MISA has much better accuracy than the alternatives. It remains fully accurate on the primary placement and has a median error of zero for the secondary placement on both datasets. On Drosophila data, MISA finds the correct secondary placement in 75% of cases, and its error does not exceed three branches in any replicate. On more challenging Lice dataset, it is within three edges for the secondary placement in all but four outlier cases where its second placement is five or six branches away from the correct placement. One of the outlier cases is again the

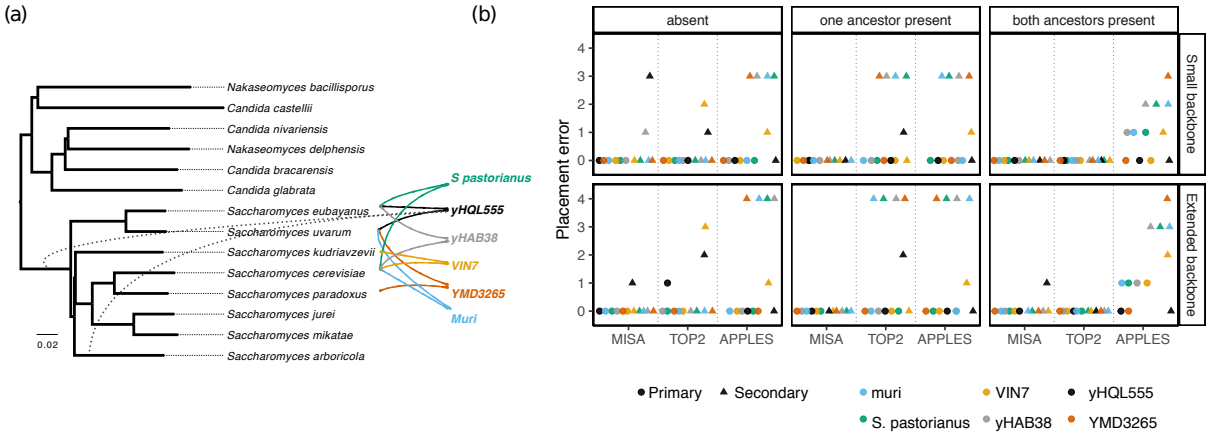


Figure 5.5. Results on the Yeast (Saccharomyces) hybrid dataset. (a) The phylogeny of 14 "pure" species and the origins for each of the six hybrid species and strains as postulated in the literature (solid lines). The dotted lines show the placement of yHQL555 when the ancestors are removed from the database. All other species are correctly placed by MISA when given this extended backbone tree. (b) Placement error for each Yeast hybrid species when hybrid ancestors are both present, one present and one absent, or both absent in the reference set. The error is shown for both a backbone phylogeny of 8 Saccharomyces species and an extended phylogeny of 14 yeast species sampling various genera. Errors are shown separately for the two placements (circle: the placement with a lower error; triangle: the placement with a higher error).

cherry, and another outlier is a replicate where the true placement is on a zero-length branch. A third outlier is a replicate where species 932, which was the outlier in terms of distance to the mixture (Figure 5.4b), is one of the two constituents.

5.4.2 Fungal Hybridization

In the fungal dataset, overall, MISA has the best ability to identify the ancestral species (Figure 5.5). When both ancestors are present in the database, MISA is able to identify both ancestors correctly in all six cases with the small backbone and all but one ancestor with the larger backbone. The exception is yHQL555, which is a mix of two sister species (i.e., a cherry). MISA puts the hybrid at relatively low phylogenetic distances (0.018 or lower in all but three cases) to both constituent (Figure 5.4a). However, distances are generally larger than Drosophila and Lice datasets, which were true mixes. These larger distances on the hybrid yeast data may indicate some amount of evolution after the hybridization event. With ancestors present, TOP2 has perfect accuracy. APPLES is rarely correct and often finds both ancestors incorrectly.

The advantage of MISA compared to TOP2 becomes clear when one of the ancestors are missing. When one ancestor is present, TOP2 finds it correctly in every case. However, except for VIN7, it finds the second ancestor incorrectly, and it can be up to four branches off. In contrast, MISA finds both placements correctly in all cases. Similar patterns are observed when both ancestors are absent. TOP2 finds one of the two placements correctly everywhere except for yHQL555 (with the large backbone) but fails to find the correct placement for the second ancestor for VIN7 and yHQL555. Here, MISA has similar performance but manages to find both ancestors correctly for VIN7 and one of the ancestors for yHQL555.

5.5 Discussion

We introduced MISA, a method for inserting a mixed sample onto two positions in a reference phylogeny. MISA is a traditional distance-based phylogenetic method with a novel twist: it seeks to decompose the measured distances between the mix and reference species into their constituent parts. To enable this "deconvolution", we introduced a simplified model that, despite its various assumptions and approximations, is useful in making sense of mixture distance (Eq. 5.4). Our results showed that not only MISA has high accuracy in identifying constituents of a mixed sample, it can also identify ancestors of a recently hybridized species. Moreover, MISA can accomplish this difficult task using the simplest possible form of input – sets of unassembled reads both for the mixed query sample and the reference set.

Our model also allowed us to prove an interesting result: a mixed sample is expected to have a lower measured distance to its constituents than to any other reference. More surprisingly, this minimum distance is expected to be lower than a small constant value. Thus, in addition to MISA, we were able to describe a simple method called TOP2 that simply picks the two smallest distances as constituents. Our theoretical and empirical results showed that this fast and simple method works well when the constituents are part of the reference set but can have reduced accuracy in other scenarios. Therefore, the power of MISA, driven from its reliance on

phylogenetic placement, is needed when we suspect a mixture may include novel or unsampled species.

Our model predicts that $\hat{\delta}_{MR_i}$ is close to the minimum of $\hat{\delta}_{MA}$ and $\hat{\delta}_{MB}$ (Figure 5.2d) for k=31. The gap between our model and this minimum value, however small, is crucial for the successful deconvolution of distances (with the minimum model, both constituent distances cannot be recovered). Importantly, this gap vanishes as k goes to infinity and becomes larger for smaller k. Our experiments show that a reasonable way to choose k is to examine the additivity of distances obtained from different values. On Drosophila and Yeast data, increasing k from 21 to 31 results in an increase in error in double-placement (Figure 5.12) as well as the lack of additivity as measured by the LSE error (Table 5.1). We therefore choose k that yields the most additive distances for the backbone tree.

MISA was relatively fast on our datasets. On the largest dataset (Lice) with 20 backbone species, the average execution time of an analysis on a mixture was approximately 30 seconds using 36 cores Intel(R) Xeon(R) Gold 6240 CPU 2.60GHz with 384GB of DDR4. Solving the optimization problem for a pair of branches took $\approx$1 second on average, and MISA runs in an embarrassingly parallel fashion across all branch pairs. To be able to extend this approach to much larger backbone trees, we will need to design heuristic methods that avoid examining all pairs of placement branches.

5.5.1 Relevant literature

Our approach to mixture analysis is quite different from the literature. Methods developed for metagenomics focus on matching reads to marker genes [e.g., 137, 214, 226] or reference genomes [e.g., 248], placing reads on a phylogenetic tree [e.g., 18, 174], and finding signatures of composition [e.g., 28, 203]. Here, we take a distance-based approach and seek to decompose observed distances into their individual parts. A somewhat similar philosophy was used by 112, 113, who used spectral methods to factorize a matrix of k-mer frequencies into an abundance vector and observed k-mer frequency matrices. This approach is different from ours because it *i*)

assumes constituents are in the library, *ii*) operates on k-mer frequencies (for small k) rather than k-mer presence or absence (for large k), and *iii*) formulates the problem as matrix factorization. While these metagenomics methods are not presently available in forms that avail themselves to application in our setting (placement of Eukaryote mixtures), future work should explore whether they can be adapted to our setting.

Another relevant literature is phylogenetic network reconstruction in the face of hybridization [see 171]. The problem addressed in network phylogeny is more challenging than our problem because networks could have hybridization at ancestral nodes. In our case, hybridization (or mixture) happens only between leaves of the tree, and little or no evolution occurs after hybridization (as time passes, the hybrid eventually ceases to resemble the combination of its constituents). Because of their more ambitious goal, explicit network methods do not operate on distances and are not based on alignment-free methods. Instead, they operate on aligned homologous loci and seek to find a network that best explains the distribution of observed gene trees. In contract, some of the popular *implicit* network methods, such as SplitsTree [96], use distances. However, these methods do not seek to find the correct placement under any model; they simply provide means of visualizing discordance (i.e., lack of additivity) among observed distances. Because the problem we address is simpler than explicit network reconstruction, we can approach it using assembly-free distance-based methods. At the same time, unlike implicit methods, we use a model that generates interpretable output (as opposed to simple visualizations of discordance). Finally, we note that our model, as presently constructed, can handle alloploidy but not homoploidy (where the hybrid is not the combination of both ancestors).

5.5.2 Shortcomings and future work

While generally accurate, MISA had a clear loss of accuracy under a special case. When the constituents of the hybrid form a cherry (e.g., are sister species), both placements should be on the same branch. The only incorrect identification by MISA on the real fungal dataset (yHQL555) was a cherry, and some of the four outlier cases with high error on the Lice dataset

were cherries. While the current formulation of MISA as two separate placements precludes finding a cherry as the output, one can still hope that it puts the two placements on the same branch. However, MISA is often able to place one but not both of the constituents on the correct branch. This limitation is not a fundamental shortcoming of our model or methodology and can be ameliorated in the future by allowing cherries as the solution to the optimization problem. Achieving this improvement requires a second round of cherry placements on the phylogeny and a change in the approximations used for x_3.

Incorrect placements of MISA can be due to either an imperfect distance deconvolution or inaccurate of distance-based placement using ordinary least squares. In other words, even if distances are deconvoluted perfectly, we can still observe erroneous placements. In fact, on both simulated datasets, where we know the true deconvolution, we observe this pattern (Figure 5.13). In particular, on the Drosophila dataset, distance deconvolution seems to contribute very little to the final error.

On the optimization side, we can enforce more constraints such as triangle inequality, but these extra constraints may challenge convergence. Also, we did not fully explore optimizer settings (e.g., the number of iterations and multiple initial points), leaving such exploration to future work. Finally, some of the approximations (for example, the use of h^{-1} for the derivation of Jacobian or the approximation of x_3) could perhaps be improved in the future.

The optimization formulation can also be further improved. Here, we enforce the model (Eq. 5.4) as hard constraints and optimize the OLS error between phylogenetic distances and sequence distances. Thus, MISA tries to find the double-placement that is closest to additivity while enforcing expectations under our model. However, our model involves stochastic uncertainty (which we ignored; see below), and thus, the constraints may be too rigid. Future work can explore alternative formulations where the model of $\hat{\hat{\delta}}_{MR_i}$ is treated as uncertain. For example, we can incorporate the difference between LHS and RHS of Eq. 5.6 as part of the optimization score; such a formulation would require a principled way to combine this penalty with the penalty for deviations from additivity (i.e., the current objective function). Furthermore,

we assigned equal weight to every term in Eq. 5.5. Previous results on distance-based placement of single-species samples show that employing a weighting scheme [e.g. 68] improves accuracy by downscaling the error contribution of long distances. At the cost of increasing the complexity of the objective function, using FM weighting may improve the accuracy of MISA.

Our model leaves several questions unanswered. We do not know whether under our model constituent distances are unique given observed distances to a set of references (we know they are not for one or two references). Moreover, in deriving the model, we freely replaced random quantities with their expectations without care for careful statistical modelling. These derivations, therefore, have another level of approximation built into them. Our mode also assumes a limit on the evolutionary divergence among reference and query species. We have no reason to believe that MISA has high accuracy on mixtures of highly divergent species (e.g., from different phyla). Nevertheless, we find it remarkable that despite all the simplifying assumptions and approximation, the method still works with high accuracy on data that violate many of those assumptions. Note that our simulated datasets were far from fully complying with our assumptions. For example, the genomic contribution of the constituents to the mixture varies from 40-60% in Drosophila, 36-62% Lice dataset (Figure 5.14).

Our analyses of hybrid yeast dataset only focused on mixes of two species. There are known cases of mixes of three species, such as S. bayanus, which is hybrid of S. uvarum, S. cerevisiae, and S. eubayanus [134]. Applied on S. bayanus, MISA identifies one of the ancestors, S. uvarum and places the second ancestor on the root of the (extended) backbone tree. Performing a second deconvolution of the uncertain placement may help identify the remaining two ancestors. In general, how this model can be extended to mixtures of three or more species remains a topic of future research.

5.6 Acknowledgement

Chapter 5, in full, is a reprint of the material as it appears in "Balaban, M., & Mirarab, S. (07 2020). Phylogenetic double placement of mixed samples. Bioinformatics, 36(Supplement_1), i335–i343". The dissertation author was the primary investigator and first author of this paper.

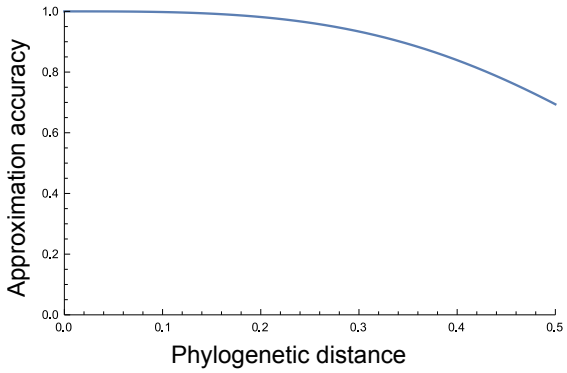
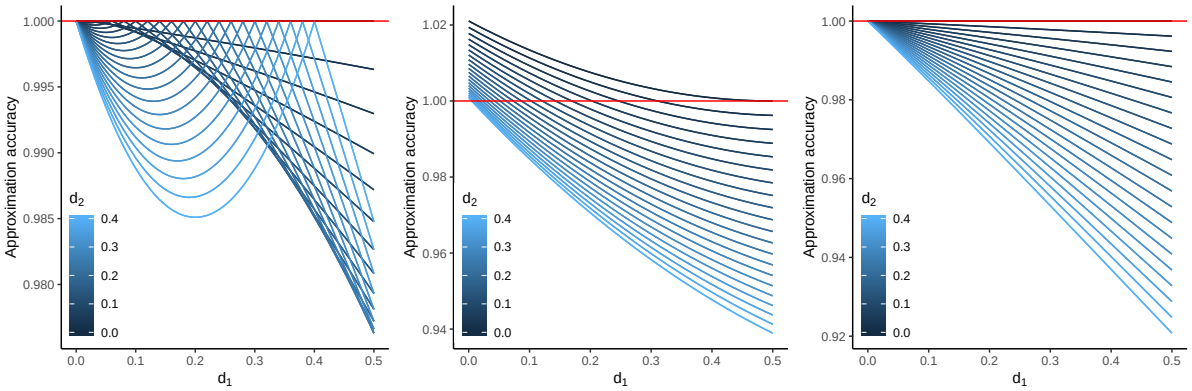


Figure 5.6. Expected proportion of $\frac{S_A \cap S_B \cap S_R \cap S_L}{S_A \cap S_B \cap S_R}$ when $\delta_{AB} = \delta_{AR} = \delta_{RB} = \delta$. x-axis shown in phylogenetic distance $h^{-1}(\delta)$. The probability that a k-mer in L remains without mutations in three extant genomes is $((1-\delta)^3)^k$. The probability that a k-mer is shared in all three extant genomes is $(\frac{\delta^3}{16} + (1-\delta)^3)^k$. The approximation accuracy is the ratio of these two probabilities.



$$f(d_1,d_2,d_3) = (1-h\left(\frac{d_1+d_2-d_3}{2}\right))(1-h\left(\frac{d_2+d_3-d_1}{2}\right))(1-h\left(\frac{d_3+d_1-d_2}{2}\right))$$

$$\tilde{f}(d_1,d_2,d_3) = (1-h\left(\frac{d_1+d_2+d_3}{2}\right)$$

$$d_3 = max(d_1,d_2)$$

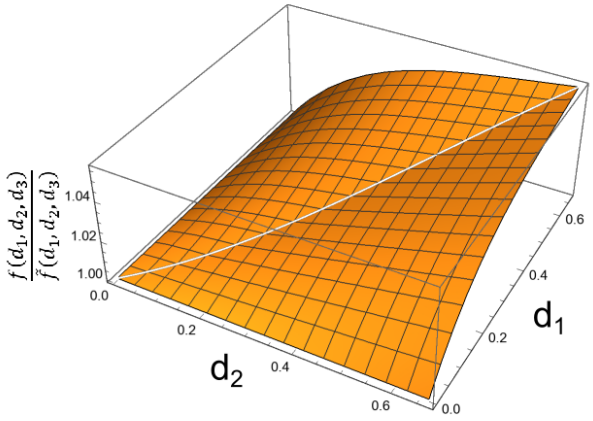


Figure 5.7. We approximate $f(d_1,d_2,d_3) = \big(1-h(\frac{d_1+d_2-d_3}{2})\big)\big(1-h(\frac{d_2+d_3-d_1}{2})\big)\big(1-h(\frac{d_3+d_1-d_2}{2})\big)$ with $\tilde{f}(d_1,d_2,d_3) = \big(1-h(\frac{d_1+d_2+d_3}{2})\big)$. We show the ratio of the two functions for a set of d_1 and d_2 values (thus, 1 shown in red means a perfect approximation). Top: from left to right, we set d_3 to its minimum, median, and maximum possible values according to the triangle inequality, which are respectively, $|d_1-d_2|$, $\max(d_1,d_2)$, and d_1+d_2. Bottom: We set $d_3 = \max\{d_1,d_2\}$. Note that in real applications, d_i values are in most cases less than 0.3.

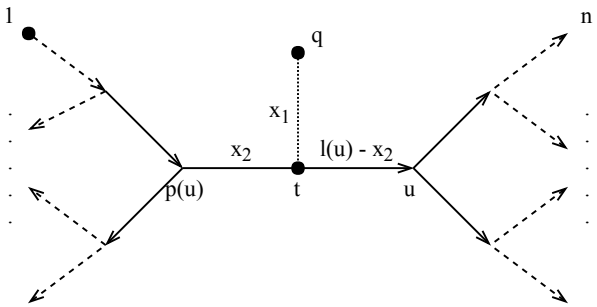


Figure 5.8. Each placement tree includes the addition of a new species onto an existing tree. Rooting the tree arbitrarily at leaf 1, the placement edge here is between $p(u)$ and u. Each placement has a distal edge length, shown as x_2 and a pendant edge length, shown as $x(1)$. Here, $p(u)$ is the parent of u and $l(u)$ is the length of u.

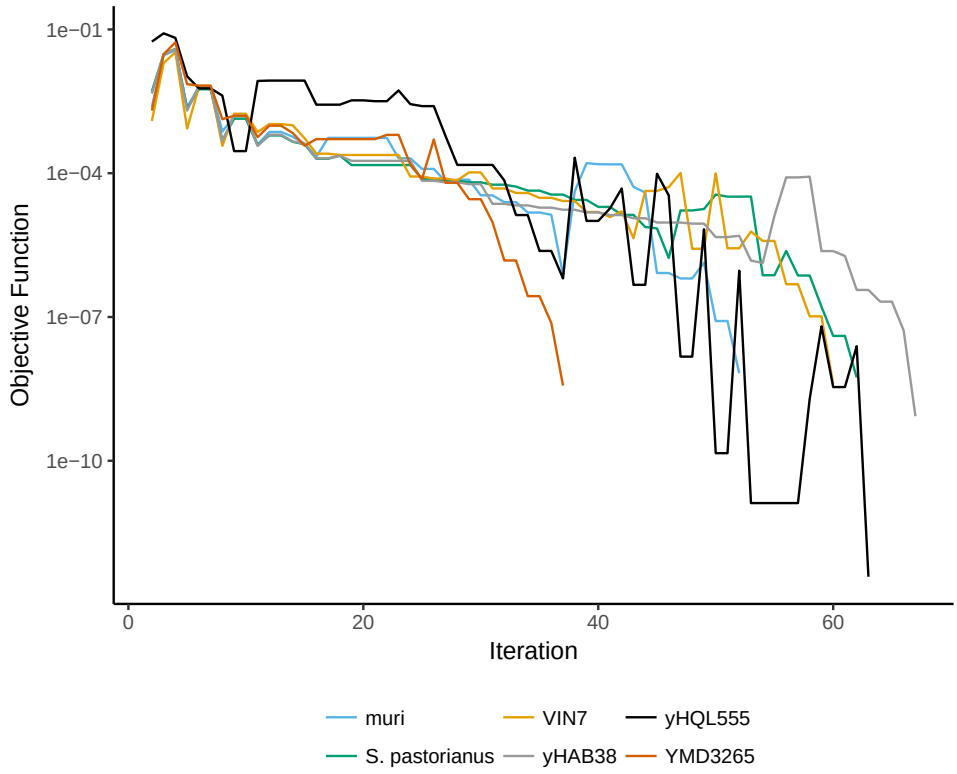


Figure 5.9. Trajectories of optimization function (i.e. OLS error) for yeast species on small backbone. All queries satisfy convergence criteria by the norm of the Lagrangian gradient (gtol=10^{-8}) and by the change of the independent variable (xtol=10^{-8}) [40] before reaching the limit of maximum number of iterations (5000).

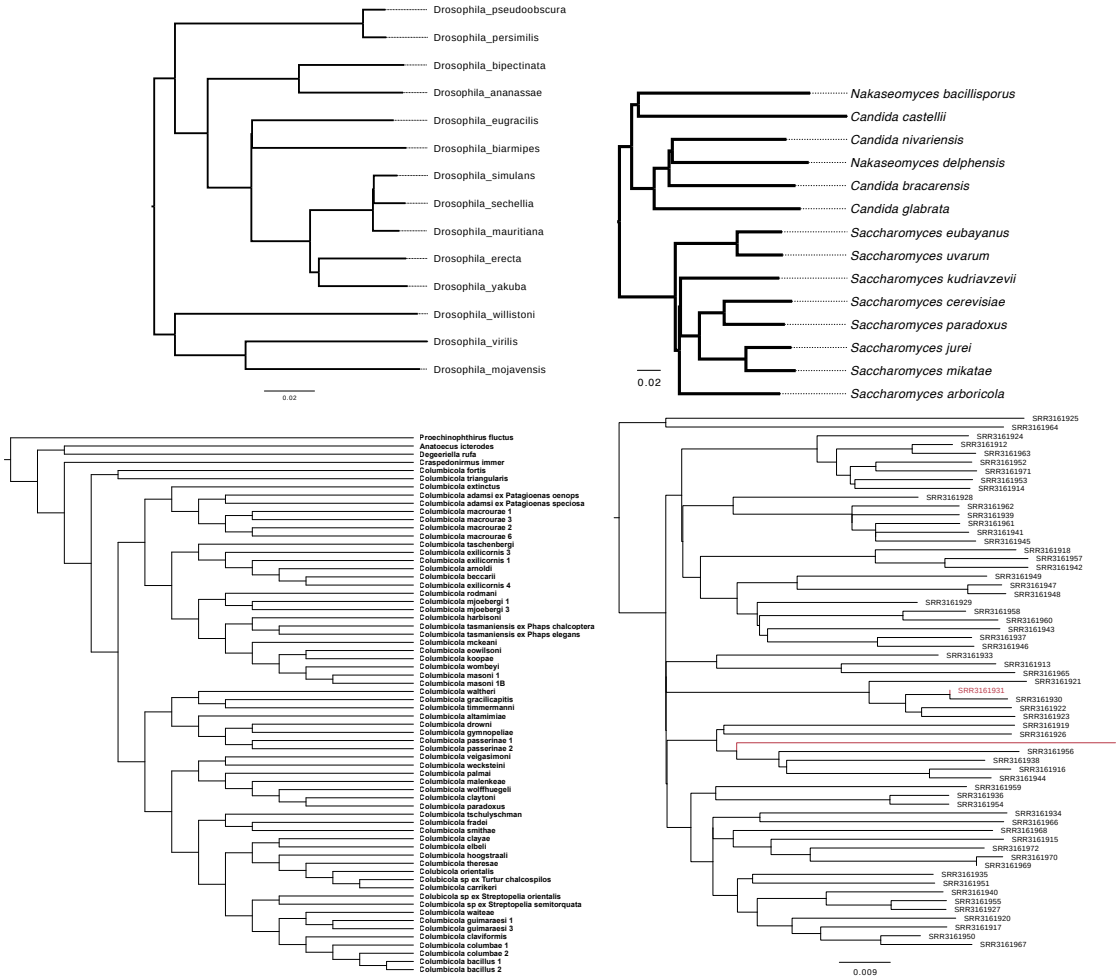


Figure 5.10. The reference biological trees obtained from [161] (Drosophila), [216] (Yeast) and Boyd et al. [27] (Columbicola).

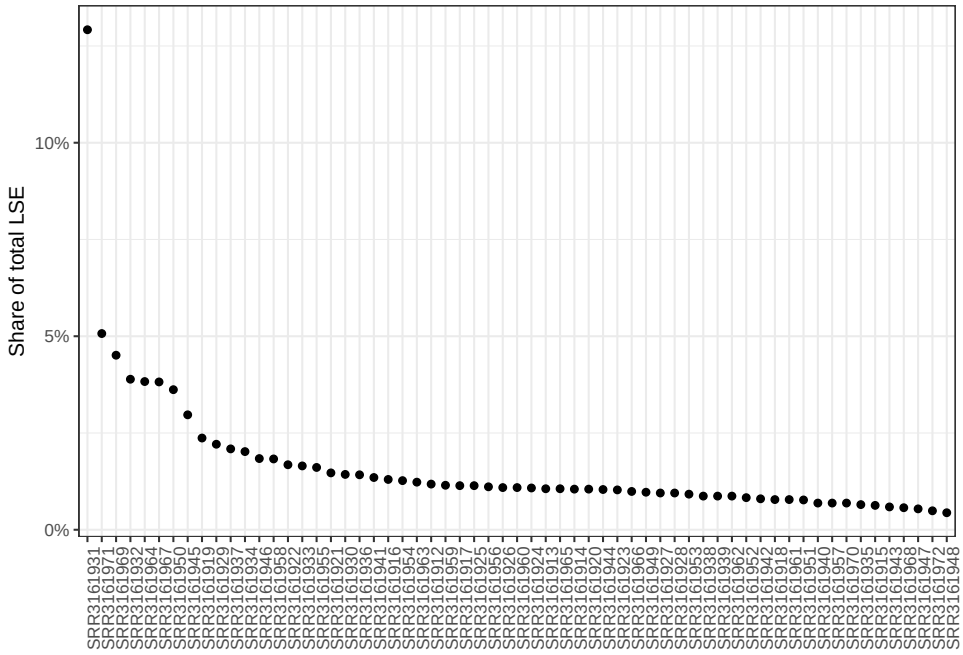


Figure 5.11. Contribution of each species in Columbicola dataset to the total LSE. Recall that LSE error is sum of squared differences between corrected sequence and phylogenetic distances for every pair of leaves in the backbone tree. Here, we show the percentage of the total LSE that is contributed by terms that include each species.

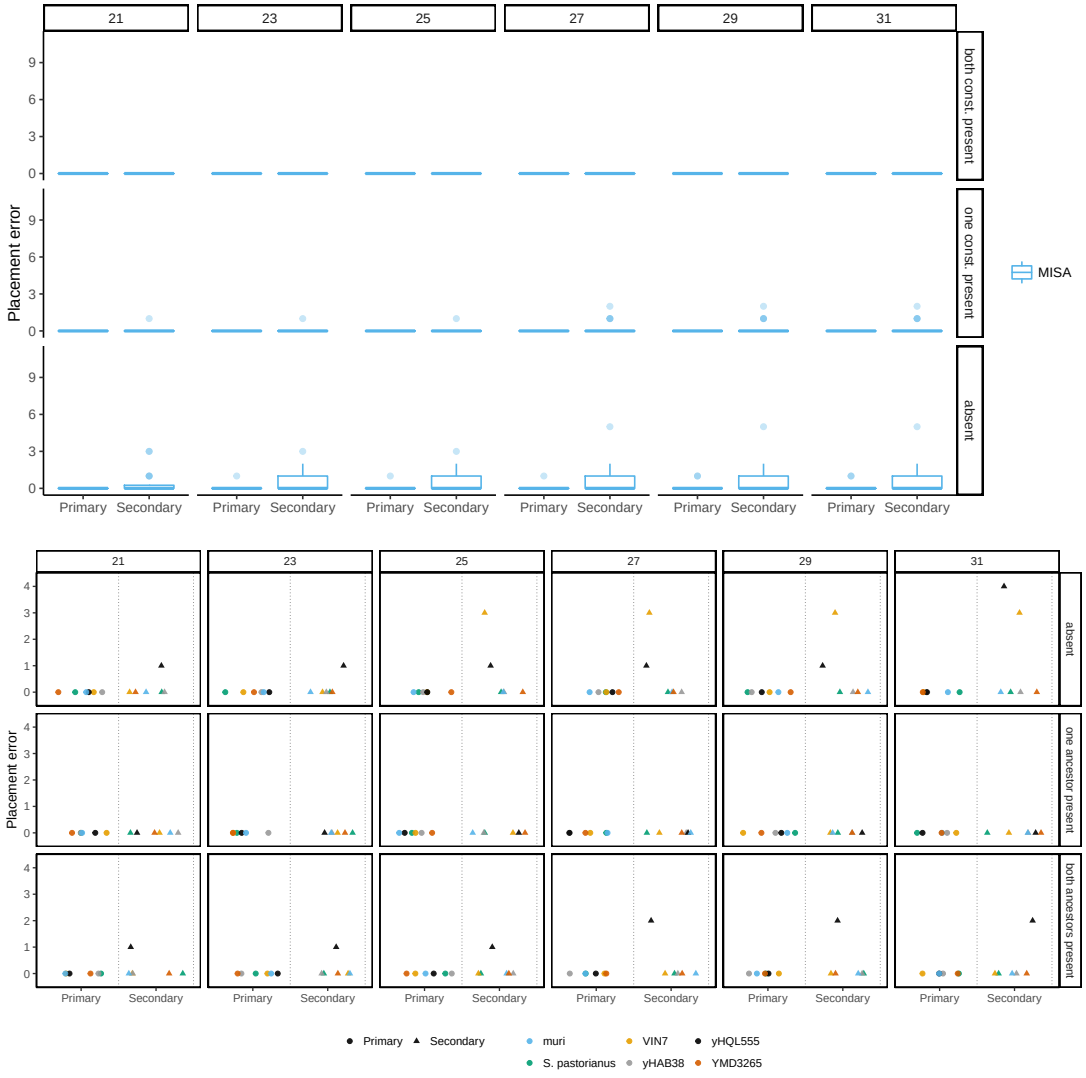


Figure 5.12. Impact of kmer size to accuracy of double-placement on Drosophila (a) and Yeast (extended) (b) datasets.

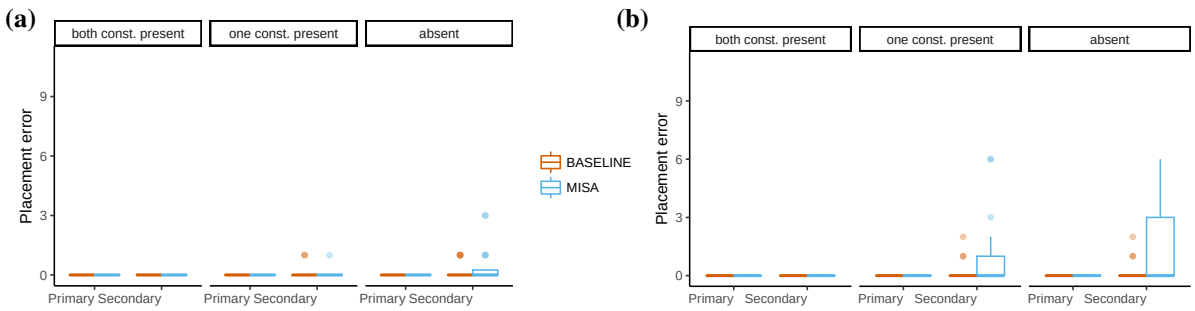


Figure 5.13. Placement error when constituents of simulated mixtures are both present, one present and one absent, or both absent in the reference set. Baseline shows ordinary least squares APPLES placement of primary and secondary constituent prior to artificial mixing (i.e., with correct deconvlution of distances). Distributions are over 20 replicates for Drosophila (a), and 22 replicates for Columbicola/Lice (b) datasets. We show the number of branches between each placement and the correct placement (i.e., 0 means perfect accuracy).

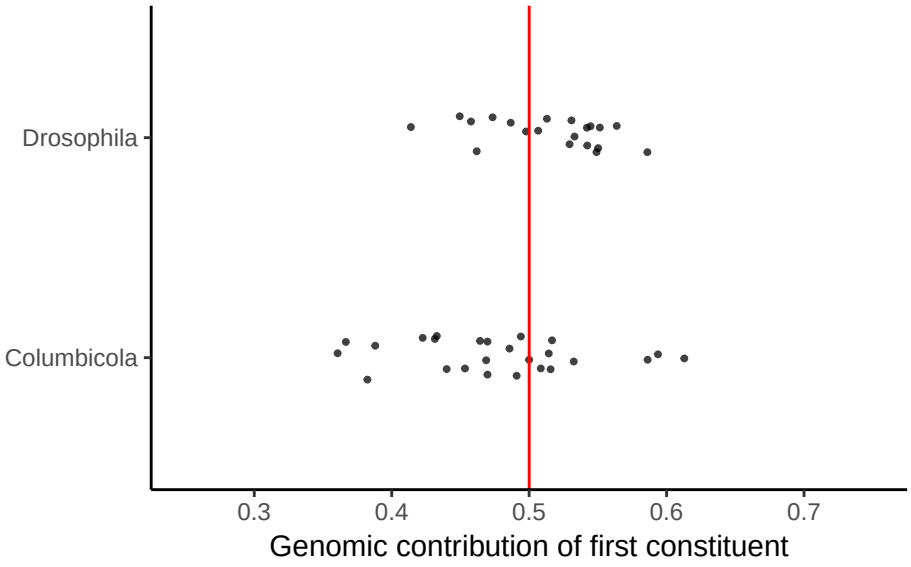


Figure 5.14. Genomic contribution of the first constituent in the entire mixture. When each constituent contributes equally, the contribution value becomes 0.5, indicated with the vertical line.

Chapter 6

Genome-wide alignment-free phylogenetic distance estimation under a no strand-bias model

Motivation: While alignment has been the dominant approach for determining homology prior to phylogenetic inference, alignment-free methods can simplify the analysis, especially when analyzing genome-wide data. Furthermore, alignment-free methods present the only option for emerging forms of data, such as genome skims, which do not permit assembly. Despite the appeal, alignment-free methods have not been competitive with alignment-based methods in terms of accuracy. One limitation of alignment-free methods is their reliance on simplified models of sequence evolution such as Jukes-Cantor. If we can estimate frequencies of base substitutions in an alignment-free setting, we can compute pairwise distances under more complex models. However, since the strand of DNA sequences is unknown for many forms of genome-wide data, which arguably present the best use case for alignment-free methods, the most complex models that one can use are the so-called no strand-bias models.

Results: We show how to calculate distances under a four-parameter no strand-bias model called TK4 without relying on alignments or assemblies. The main idea is to replace letters in the input sequences and recompute Jaccard indices between k-mer sets. However, on larger genomes, we also need to compute the number of k-mer mismatches after replacement due to random chance

as opposed to homology. We show in simulation that alignment-free distances can be highly accurate when genomes evolve under the assumed models and study accuracy on assembled and unassembled biological data.

Availability: Our software is available open-source at https://github.com/nishatbristy007/NSB

6.1 Introduction

The dominant methodology used in phylogenetic inference is assembling and aligning sequences and using the alignments as input to phylogenetic inference. However, a large body of work also exists on alignment-free [25, 46, 84, 91, 106, 126, 249] and even assembly-free methods for inferring phylogenies [3, 63, 135, 209, 255]. While, for the most part, the alignment-free methods have not been as accurate as alignment-based methods [25, 91], they do provide several benefits and also enjoy emerging applications. The most obvious advantage is that inferring alignments is difficult, and forgoing them would simplify the tree inference. The challenges are further exacerbated when working with genome-wide data where long sequences and large-scale events such as rearrangements make alignment even more challenging [264]. There is, therefore, a hope that by skipping the alignment step, we can eliminate the errors [264] that are known to occur in the alignment step and impact phylogenetic accuracy [132, 143, 177]. In particular, at the whole-genome level, homology detection and alignment are both difficult and error-prone [55, 128, 219]. Therefore, it seems possible (though by no means certain) that alignment-free methods could provide a better trade-off between accuracy, running time, and complexity of analyses, especially for analyzing genomes [72].

The main advantage of alignment-free methods may come from situations where alignment is not possible. In particular, genome skimming has recently emerged as a promising method of acquiring genome-wide data inexpensively [26] by generating short reads from across the genome at low coverage (e.g., 1X). While such data cannot be assembled, mapping them against a reference genome, when available [242], or analyzing them in an assembly-free fashion,

when references are not available, are now possible [10, 13, 121, 209, 230]. Multiple sequence alignment is not an option given reads with low coverage, leaving us with alignment-free methods as the only possibility. Among assembly-free methods, many use k-mers to compute distances between all pairs of species and then use distance-based methods to infer the phylogeny or to classify and/or cluster them. A long history [198, 199, 255] of methods using k-mer counts (with small k) exist. Some recent k-mer-based methods that work with both assembled and unassembled data and model low coverage instead use presence/absence with large k [63, 209, 230]. We refer the reader to a recent benchmarking analysis for a complete survey [265].

Despite their practical benefits in terms of simplicity and scalability, alignment-free methods have limitations of their own. One of these limitations is the reduced complexity of sequence evolution models employed. Most alignment-free methods rely on the simplest possible model of sequence evolution, Jukes-Cantor (JC) [105], which assumes equiprobable bases and base substitutions. Criscuolo [41] recently showed how to compute alignment-free distances under a slightly more complex Felsenstein [65] (F81) model where the base frequencies can be different. In contrast, alignment-based methods use more complex models, such as the general time-reversible (GTR) [231] model paired with models of rate variation across sites and further partitioning data to allow changing model parameters. The reliance on models like JC and F81 is not an oversight by the research community. In the absence of alignments, it is more challenging to design methods for more complex sequence evolution models that need to estimate parameters related to relative rates of substitutions among bases. The difficulties are exacerbated by the fact that sequences can come from either of the two strands for unassembled and unaligned data, making it difficult to calculate some parameters of complex models and impossible to compute others [257]. Despite that, Sarmashghi et al. [209] proposed a trick that they conjectured could be used in conjunction with the well-known LogDet technique [222] to compute distances under the GTR model from unassembled reads. The claim that distances under more complex time-reversible models like GTR can be computed from unassembled data has never been carefully examined.

In this paper, we argue that when input data is unassembled, where reads can be of either strand (mixed-strand data), no strand-bias models are the most complex time-reversible models one can employ. We go on to describe an algorithm that can estimate all the parameters needed to compute distances for a time-reversible no strand-bias model called TK4 [227]. Our proposed algorithm, in summary, replaces the nucleotide characters in input sequences in four different ways (e.g., $C \to G$), and computes the Jaccard index between these letter-substituted sequences. However, we observed that a fundamental assumption of many k-mer-based methods, including Skmer, that matching k-mers can only appear by homology for a large enough k, can easily be violated after letter substitutions, especially for genomes with unbalanced base frequencies, since the number of characters in the base genomes decreases from four to three. Luckily, the expected number of random matches between two k-mers from two random genomes can be derived [202], and we go one step further and compute the expected (containment) Jaccard between two unrelated genomes (Lemma 5). Using these calculations, we can correct for the effect of non-homologous k-mer matches. Using analytical calculations and simulations, we show that using this technique to compute distances under the TK4 model can improve the accuracy of estimated distances compared to JC, especially when the distances are high and deviations from the assumptions of the JC model are sufficiently high. We then use real biological data to demonstrate that using the TK4 model improves the concordance of phylogenetic trees inferred from alignment-free methods and those inferred from alignment-based methods, indicating improved accuracy. We end by discussing the limitations of the method.

6.2 Approach

6.2.1 Background information

Evolutionary model.

Suppose that we have two homologous DNA sequences $\mathscr{G}$ and $\mathscr{H}$ on character alphabet $\Sigma = \{A, C, G, T\}$ taken from two species $\mathscr{F}_1$ and $\mathscr{F}_2$ that share a common ancestor. For a given base $i \in \Sigma$, let $\bar{i}$ denote its complementary base (e.g., $\bar{A} = T$). We assume that each homologous

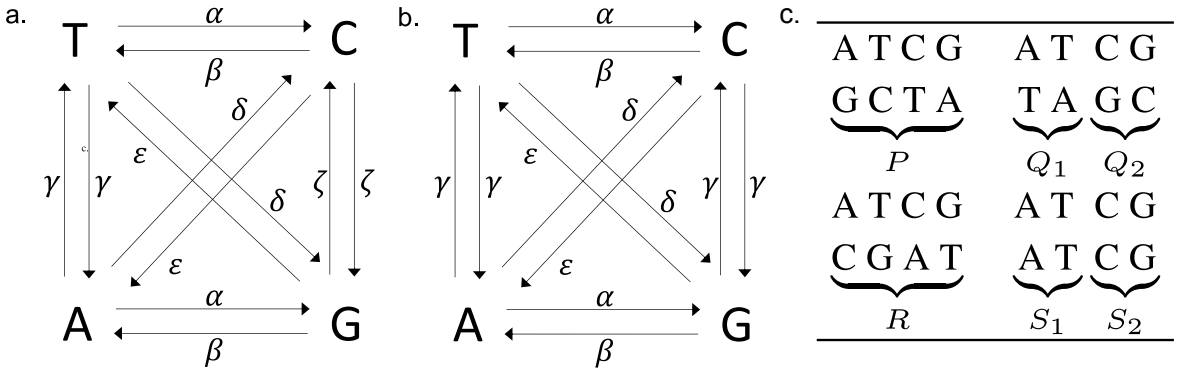


Figure 6.1. (a) Sueoka's no strand-bias model of evolution with 6 rate parameters. TK5 model (b) is a special case of the 6-parameter model with the constraint $r_{AT} = r_{GC}$. TK4 is the time-reversible version of TK5 model with the condition $\omega = \frac{\beta}{\alpha+\beta} = \frac{\varepsilon}{\varepsilon+\delta}$ where ω is the total equilibrium frequency of bases A and T. (c) Nucleotide base pairs in homologous sites and their observed relative frequencies.

site in $\mathscr{G}$ and $\mathscr{H}$ is evolved independently and according to a stationary continuous-time Markov-chain process on state set Σ that is defined by a 4×4 instantaneous rate matrix $\mathbf{R} = (r_{ij})$. Letting $\pi = \begin{bmatrix} \pi_A & \pi_C & \pi_G & \pi_T \end{bmatrix}$ denote the stationary base frequencies in $\mathscr{G}$ and $\mathscr{H}$, (thus, $\pi\mathbf{R} = 0$) the most general time-reversible stationary model, GTR [231], adds local balance constraints (i.e., $\forall i, j : \pi_i r_{ij} = \pi_j r_{ji}$), which lead to nine free parameters. Another constraint is added by requiring the time to be in the unit of one expected mutation, leaving us with eight free parameters. The transition matrix $\mathbf{P} = e^{\mathbf{R}t}$ governs probabilities of base substitutions after time t.

Our goal is to estimate the time of divergence t between the two given genomes. Such estimates, if statistically unbiased, would converge to additivity and can be used with any distance-based phylogenetic inference method. In the last 50 years, numerous models with reduced complexity (i.e., fewer parameters) compared to the general Markov model have been proposed [83, 105, 222, 228], and some of these models have analytical equations for distance calculations [83, 228]. For example, let *genomic distance* d be the probability of observing a change in a homologous position. Under the simplest model, JC, the maximum likelihood estimator is

$$\hat{t} = -\frac{3}{4}\ln\left(1 - \frac{4}{3}d\right) \tag{6.1}$$

No strand-bias models.

A restriction of GTR, relevant to the study of Next-gen sequencing (NGS) reads, is the model proposed by Sueoka [224]. Chargaff [36] had earlier noted that in double-stranded DNA, the frequency of A should equal T, and that of G should equal C (parity rule 1). Thus, an $i \to j$ substitution occurring on the forward DNA strand must have an identical rate to an $i \to j$ substitution occurring on the reverse strand, which is the basis of Sueoka's no strand-bias model (Fig. 6.1). Since an $i \to j$ entails an $\bar{i} \to \bar{j}$ substitution on its opposite strand, the model constrains $r_{ij} = r_{\bar{i}\bar{j}}$ and therefore reduces the number of independent parameters in the model to six. Surprisingly, the parity of A with T and C with G has been extensively documented on single-strand DNA as well (parity rule 2) [168]. The reason behind parity on a single strand has been debated from the start [71, 74] and continue to be debated [73, 160], with the two (not mutually exclusive) hypotheses based on a) Sueoka's model of mutational bias in the replication of polymerase in neutrally evolving genomes [140, 224], and b) Forsdyke's structural model that invokes selective pressure. Regardless of the cause of parity rule 2, a no strand-bias model can be appropriate even for single-strand data, as Sueoka intended the model to be used.

In this paper, we deal with conditions where the no strand-bias model is the *best* we can do due to parity rule 1. Assume that $\mathscr{G}$ is not a single stranded sequence but a set of n homologous sequences $\mathscr{G}_1, \mathscr{G}_2, \cdots, \mathscr{G}_n$ (similarly for $\mathscr{H}$) where each sequence $\mathscr{G}_i$ or $\mathscr{H}_i$ comes from an arbitrary strand. Inputs made of k-mers, reads, or (unaligned) contigs can be viewed this way. With this data, r_{ij} is unidentifiable from $r_{\bar{i}\bar{j}}$. The main limitation of the no strand-bias model is that it does not allow analytical calculation of distances [257].

TK4.

Predating Suoeka's paper by 14 years, Takahata and Kimura introduced a 5-parameter non-time reversible model called TK5 [227] (Fig. 6.1b) that imposes on the general 6-parameter model the constraint $r_{AT} = r_{TA} = r_{GC} = r_{CG} = \gamma$ and assumes that $\pi_A = \pi_T = \omega/2$ and $\pi_C = \pi_G = (1-\omega)/2$. By imposing $\omega = \frac{\beta}{\alpha+\beta} = \frac{\varepsilon}{\varepsilon+\delta}$, Takahata and Kimura introduce a time-reversible

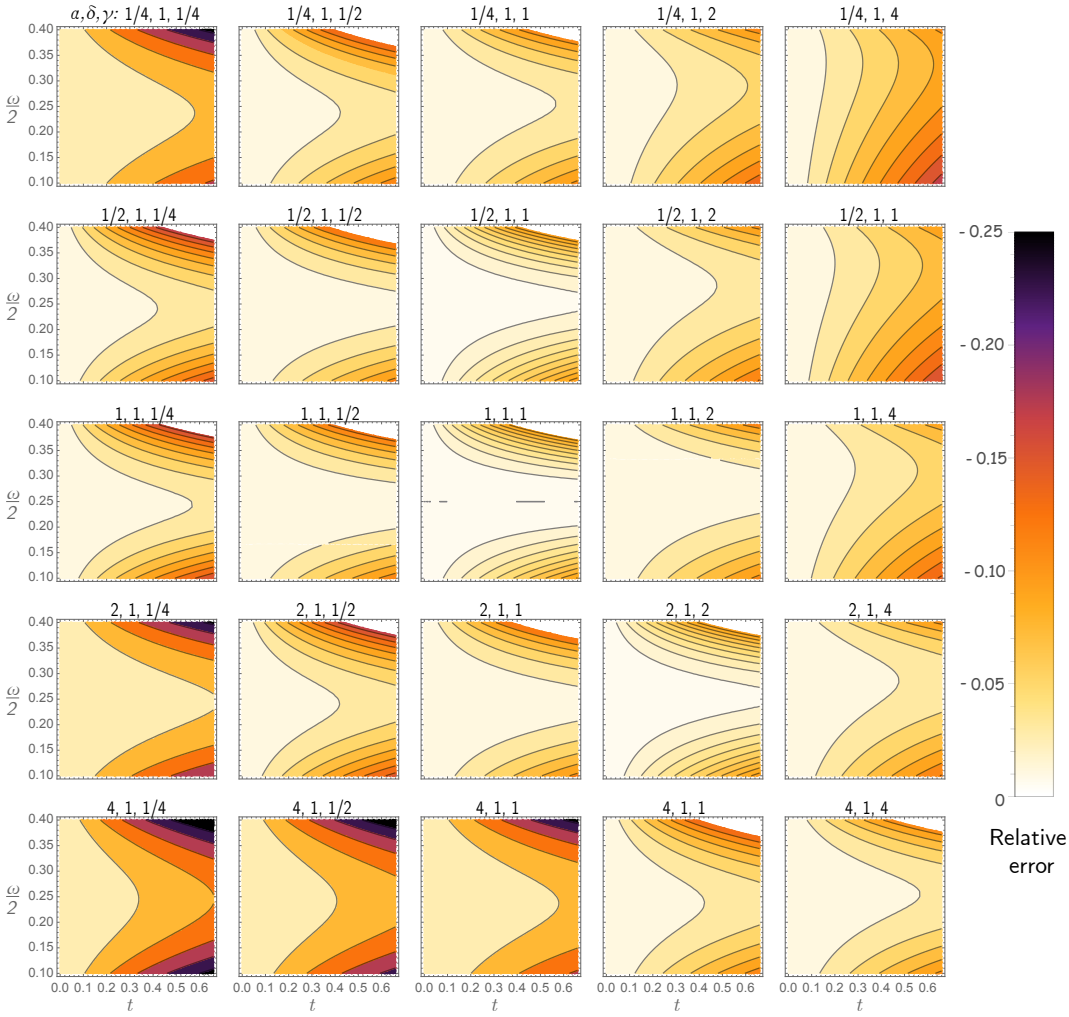


Figure 6.2. JC under-estimates TK4. The colors show relative bias of JC defined as $\frac{\hat{t}-t}{t}$ with $\hat{t}$ computed using equation (6.1) where d is set to the expected hamming distance under TK4, which can be computed as $d = 1 - \pi.\text{diag}(e^{\mathbf{R}t})$. Each subplot corresponds to a choice of α, δ, γ, changing α across rows and γ across columns, fixing $\delta = 1$. The x-axis changes the true evolutionary distance t, and the y-axis changes the base frequency parameter ω. Note that the JC model corresponds to $\alpha = \delta = \gamma = 4\omega = 1$.

version of TK5 model with 4 parameters, called TK4 [227], and derive an analytical formula for distance estimation under TK4. This equation uses 16 combinations of bases possible at each site, as summarized in Figure 6.1c. Let f_{ij} for $i, j \in \Sigma$ denote the relative frequency of sites where the first and second genome has character i and j, respectively. We define $P = f_{AG} + f_{GA} + f_{TC} + f_{CT}$, $Q = f_{AC} + f_{CA} + f_{TG} + f_{GT}$, $R = f_{AT} + f_{TA}$, $S = f_{CG} + f_{GC}$, $S_1 = f_{AA} + f_{TT}$, and $S_2 = f_{CC} + f_{GG}$.

Note that $P+Q_1+Q_2+R+S_1+S_2=1$. An unbiased estimated phylogenetic distance $\hat{t}$ between $\mathscr{G}$ and $\mathscr{H}$ is given in terms of relative frequencies as follows:

$$\hat{t} = -\frac{1}{4}\ln\left[\left\{\frac{(S_1-Q_1)(S_2-Q_2)-(\frac{P-R}{2})^2}{\omega(1-\omega)}\right\} \cdot\left\{1-\frac{P+R}{2\omega(1-\omega)}\right\}^{8\omega(1-\omega)-1}\right] \tag{6.2}$$

where ω can also be written as:

$$\omega = S_1+Q_1+\frac{1}{2}(P+R) \tag{6.3}$$

We note that the Takahata and Kimura [227] article has a mistake and has the term (S_1+Q_1) in (6.2) instead of (S_1-Q_1). Substituting the values of $X_-(T)$ and $Y_-(T)$, as defined in Eqn. 2 of the original paper, to Eqn. 18 in the original paper results in (S_1-Q_1) instead of (S_1+Q_1).

Comparing (6.1) and (6.2), it is not immediately obvious if the differences are consequential. By plotting the relative difference between (6.1) given the expected hamming distance under TK4 and the true time t, we can see that when parameters diverge from JC in biologically plausible ways, the often-used equation (6.1) can underestimate the true distance by more than 25% (Fig. 6.2). For example, with an AT-rich genome with $\omega=0.75$, setting $\alpha=4$ but keeping all other parameters equal to JC leads to 8% and 16% bias for true distances $t=0.25$ and 0.5, respectively. As expected, bias is reduced when TK4 parameters are all close to 1 (i.e., JC assumption). Overall, it seems that high levels of bias correspond to cases where some of the relative rates diverge from others while base frequencies also diverge substantially from 25% (both of which are biologically plausible).

Assembly-free distance estimation.

Although it is trivial to compute observed frequencies of substitutions between two aligned sequences, such calculations are challenging in the absence of alignment, for instance,

when inputs are sets of unassembled reads. In the assembly-free setting, most methods assume the simple JC model, which only requires genomic distance. Luckily, various alignment-free methods can estimate d [100, 179, 209, 255]. Many of these algorithms [179, 209] break down the genome skims into k-mers.

We assume that a genome $\mathscr{X}$ is a finite i.i.d. stochastic process $X_1X_2\cdots X_L$ where each random variable (site) X_m is drawn from categorical distribution with probability distribution $P[X_m = A] = P[X_m = T] = \pi_A = \pi_T = \omega/2$ and $P[X_m = C] = P[X_m = G] = \pi_C = \pi_G = (1-\omega)/2$. A k-mer at position m is $X_mX_{m+1}\cdots X_{m+k-1}$ and denoted with x_m in short. We make the standard simplifying assumption of k-mer independence (x_m is independent from all $k-1$ k-mers on each side). We denote the set of all k-mers in $\mathscr{X}$ with $s(\mathscr{X})$. When k is sufficiently large with respect to L and ω, we can assume that $|s(\mathscr{X})| \approx L$. A second genome $\mathscr{Y}$ is originated from $\mathscr{X}$ through a substitution process described earlier. The probability of a match between two homologous k-mers is $(1-d)^k$. Therefore, the expected total number of homologous k-mer matches between $s(\mathscr{X})$ and $s(\mathscr{Y})$ is approximately $S = L\cdot(1-d)^k$ [63, 179, 209]. Denoting by $C = S/L$ the containment Jaccard index, note that

$$\hat{d} = 1 - \mathrm{C}^{\frac{1}{k}} . \tag{6.4}$$

The *Jaccard index* J, defined as the intersection divided by the union of two sets, is easy to compute using techniques such as min-hash [179]. Thus, instead of C, most methods have relied on J, which is intimately connected to C because $J = \frac{S}{2L-S}$ and thus, $\mathrm{C} = \frac{S}{L} = \frac{2J}{1+J}$ Finally, following the TK4 notations, $\hat{d} = P + Q_1 + Q_2 + R$ holds.

6.2.2 Containment Jaccard correction

In addition to homologous ones, k-mers in non-homologous positions in the two genomes can also match, albeit with lower probability. Distance estimation using Jaccard index requires computing the number of shared k-mers through homology. The number of non-homologous

k-mer matches contributing to $|s(\mathscr{X}) \cap s(\mathscr{Y})|$ is negligible in most settings when k is large enough for the size of the alphabet; e.g., $k = 31$ with $|\Sigma| = 4$, leading to $4^{31} \approx 4 \times 10^{18}$ possible k-mers. However, as we will see, our algorithm for estimating TK4 distances requires reducing the alphabet set to three letters, and based on the value of ω, may lead to biased probabilities. Under such conditions, the non-homologous k-mer matches cannot be ignored.

Röhling et al. [202] have derived an expression for the expected number of k-mers x_m and y_n, $n \neq m$ that match between the two genomes by chance (i.e., not through homology). However, to compute the contribution of non-homologous k-mer matches to $|s(\mathscr{X}) \cap s(\mathscr{Y})|$, not only we need to know the expected number of k-mers matching by chance, we also need to account for a k-mer x_m matching multiple k-mers in the other genome. Consequently, we propose a more precise estimate for the cardinality of intersection between two random genomes.

Lemma 5. *The expected value of* $\tilde{\mathrm{C}} = \frac{|s(\mathscr{X}) \cap s(\mathscr{Z})|}{L}$*, containment Jaccard for k-mers between two genomes* $\mathscr{X}$ *and* $\mathscr{Z}$ *generated by two i.i.d processes with stationary distribution* $\pi_A = \pi_T = \omega/2$ *and* $\pi_C = \pi_G = \frac{1-\omega}{2}$*, is:*

$$\mathbb{E}[\tilde{\mathrm{C}}] = \frac{2^k}{L} \sum_{a=0}^{k} \left(1 - \left(1 - \left(\frac{\omega}{2}\right)^a \left(\frac{1-\omega}{2}\right)^{k-a}\right)^L\right)^2 \binom{k}{a} \tag{6.5}$$

Proof. For $0 \leq a \leq k$, let $r \in \Sigma^k$ be a k-mer with a A's and T's.

$$P(x_m = r) = P(z_m = r) = \left(\frac{w}{2}\right)^a \left(\frac{1-w}{2}\right)^{k-a}.$$

Thus, due to independence assumption of x_m from its overlapping neighbors, the probability of r

being in set $s(\mathscr{X})$ is the following

$$\begin{aligned} P(r \in s(\mathscr{X})) &= 1 - P(r \notin s(\mathscr{X})) = 1 - \prod_{m=1}^{L} P(r \neq x_m) \\ &= 1 - (1 - P(r = x_m))^L \\ &= 1 - \left(1 - \left(\frac{w}{2}\right)^a \left(\frac{1-w}{2}\right)^{k-a}\right)^L \end{aligned}$$

Results follows by noting that there are $2^k \binom{k}{a}$ many selections for each r and $P(r \in s(\mathscr{X})) = P(r \in s(\mathscr{Z}))$. □

By stationarity of the substitution process, $\mathscr{Y}$ has the same base frequencies as $\mathscr{X}$. Thus, $|s(\mathscr{X}) \cap s(\mathscr{Z})|$ can be used to estimate the non-homologous portion of $|s(\mathscr{X}) \cap s(\mathscr{Y})|$. In other words, $|s(\mathscr{X}) \cap s(\mathscr{Y})| - |s(\mathscr{X}) \cap s(\mathscr{Z})|$ is the number of homologous k-mers. Combining (6.4) and (6.5), d can be estimated from the containment Jaccard C of $\mathscr{X}$ and $\mathscr{Y}$:

$$\hat{d} = 1 - (\mathrm{C} - \mathbb{E}[\tilde{\mathrm{C}}])^{\frac{1}{k}} \tag{6.6}$$

On unassembled data, we account for lack of coverage and sequencing errors when computing $\hat{d}$ using the approach described by Sarmashghi et al. [209] as detailed in D.1.

6.2.3 Calculation of TK4 terms via replacement

Given the possibility of high error with the JC model (Fig. 6.2), we would like to develop alignment-free methods of computing distances according to the TK4 model using (6.2). Therefore, our goal is to *estimate* the terms P, Q_1, Q_2, R, S_1, S_2, and ω. Consider the replacement technique where every occurrence of a character $i \in \Sigma$ in $\mathscr{X}$ and $\mathscr{Y}$ is replaced with character $j \in \Sigma$, $i \neq j$. Let d_{ij} be the genomic distance between two genomes after such replacement. The reduction in genomic distance after i to j substitution is exactly $f_{ij} + f_{ji}$. Using the Eqn. (6.6), d_{ij} can be estimated from empirical containment Jaccard C_{ij} and expected number of background

k-mer matches $\mathbb{E}[\tilde{C}_{ij}]$. Using this replacement scheme, the P, Q_1, Q_2 and R terms in (6.2) are estimated as follows:

$$\begin{aligned} P &= 2\hat{d} - \hat{d}_{AG} - \hat{d}_{CT} \qquad & Q_1 &= \hat{d} - \hat{d}_{AT} \\ R &= 2\hat{d} - \hat{d}_{AC} - \hat{d}_{GT} \qquad & Q_2 &= \hat{d} - \hat{d}_{CG} \end{aligned} \tag{6.7}$$

As base frequencies $\omega = {}^{(\pi_A + \pi_T)}/_2$ can be trivially computed from $\mathscr{X}$ and $\mathscr{Y}$, we can compute the remaining terms S_1 and S_2 using (6.3):

$$S_1 = \omega - Q_1 - \frac{P+R}{2} \qquad S_2 = 1 - \omega - Q_2 - \frac{P+R}{2} \tag{6.8}$$

As mentioned previously, estimating d_{ij} requires computation of $\mathbb{E}[\tilde{C}_{ij}]$. Calculation of this term depends on the type of replacement. Lemma 5 can be easily updated to account for replacements. For instance,

$$\begin{aligned} \mathbb{E}[\tilde{C}_{AT}] &= \frac{1}{L}\sum_{a=0}^{k}\left(1-\left(1-\omega^a\left(\frac{1-\omega}{2}\right)^{k-a}\right)^L\right)^2\binom{k}{a}2^{k-a} \\ \mathbb{E}[\tilde{C}_{CG}] &= \frac{1}{L}\sum_{a=0}^{k}\left(1-\left(1-\left(\frac{\omega}{2}\right)^a(1-\omega)^{k-a}\right)^L\right)^2\binom{k}{a}2^{a} \\ \mathbb{E}[\tilde{C}_{AC}] &= \mathbb{E}[\tilde{C}_{AG}] = \\ &\frac{1}{L}\sum_{a=0}^{k}\sum_{b=0}^{k-a}\left(1-\left(1-(\frac{1}{2})^a(\frac{\omega}{2})^b(\frac{1-\omega}{2})^{k-a-b}\right)^L\right)^2\binom{k}{a}\binom{k-1}{a} \end{aligned} \tag{6.9}$$

Since letter replacements, especially A to T for $\omega > 0.5$ and G to C for $\omega < 0.5$, can lead to a very high expected number of shared k-mers by chance, the use of these equations to correct for their effect is essential. For example, with a pair genomes of length 10^8 and $\omega = 0.6$, the expected number of background matches between 2-way genomes after A-to-T replacement is 289,000, which is $5\times$ larger than the number of homologous k-mer matches when $t = 0.5$. Figure 6.7 shows the accuracy of equations (6.9) and their improvement over simply using the expected number of k-mer matches, as derived by Röhling et al. [202].

6.2.4 Handling mixed-strand conditions

We now consider the case in which each k-mer in $\mathscr{X}$ and $\mathscr{Y}$ may come from the forward or reverse DNA strand arbitrarily. In practice, chromosomes or contigs in an assembly or reads in a sequencing run may come arbitrarily from either forward or reverse strands. For simpler exposition, assume each genome consists of a single contig from an unknown strand (no such assumption needed by method). Let $\mathscr{X}'$ be another finite i.i.d. stochastic process $X'_1X'_2X'_3\ldots X'_L$ such that is $X'_i = X_i$ with some unknown but fixed probability $p_x > 0$ and $X'_i = \bar{X}_{L-i}$ with probability $1 - p_x$ where $\bar{X}_i$ is reverse complement (RC) of X_i. $\mathscr{Y}'$ is defined similarly. Genomic distance between $\mathscr{X}$ and $\mathscr{Y}$ can still be computed using (6.4) by using canonical k-mers, a concept utilized by several tools [150, 179]. We utilize the same concept and construct a 2-way genome $\dot{\mathscr{Z}} = Z'_1Z'_2Z'_3\ldots Z'_L\bar{Z}'_L\bar{Z}'_{L-1}\bar{Z}'_{L-2}\ldots\bar{Z}'_1$ with $\mathscr{Z} \in \{\mathscr{X}, \mathscr{Y}\}$ by adding the RC of each genome to itself. By design, both forward and reverse copies of each k-mer in $\mathscr{Z}$ are present in $\dot{\mathscr{Z}}$. If $x_m = y_m$, either $(\dot{x}_m = \dot{y}_m) \wedge (\dot{x}_{2L-m} = \dot{y}_{2L-m})$ or $(\dot{x}_m = \dot{y}_{2L-m}) \wedge (\dot{x}_{2L-m} = \dot{y}_m)$. Either way, the number of homologous k-mer matches and genome length both double compared to the case where all sequences are of the same strand, leaving containment Jaccard due to homologous k-mers unchanged; thus, Eqn. (6.6) is applicable to 2-way genomes as long as $\mathbb{E}[\tilde{\mathrm{C}}]$ is computed with $2L$.

Similarly to the replacement technique shown previously, we introduce i to j replacements on a 2-way genome. For each homologous site (X_m, Y_m) in the base genomes $\mathscr{X}$ and $\mathscr{Y}$, we have two pairs of homologous sites in $\dot{\mathscr{X}}$ and $\dot{\mathscr{Y}}$. Although there are 4 alternative choices for assignment of forward and reverse strand to $\{\dot{X}_m, \dot{Y}_m, \dot{X}_{2L-m}, \dot{Y}_{2L-m}\}$, without loss of generality, let $(\dot{X}_m, \dot{Y}_m) = (X_m, Y_m)$ and $(\dot{X}_{2L-m}, \dot{Y}_{2L-m}) = (\bar{X}_m, \bar{Y}_m)$. After replacing every occurrence of

$i \in \Sigma$ with $j \in \Sigma$ in $\dot{\mathscr{X}}$ and $\dot{\mathscr{Y}}$,

$$\begin{aligned} P(\dot{X}_m = \dot{Y}_m) &= 1 - \big(d - P(X_m = i, Y_m = j) \\ &\quad - P(X_m = j, Y_m = i)\big) \\ P(\dot{X}_{2L-m} = \dot{Y}_{2L-m}) &= 1 - \big(d - P(X_m = \bar{i}, Y_m = \bar{j}) \\ &\quad - P(X_m = \bar{j}, Y_m = \bar{i})\big) \end{aligned}$$

The reduction in genomic distance between $\dot{\mathscr{X}}$ and $\dot{\mathscr{Y}}$ after the replacement, $\hat{d} - \hat{d}_{ij}$, is $f_{ij} + f_{ji}$ in the forward strand (i.e. $(\dot{X}_1, \dot{Y}_1) \dots (\dot{X}_L, \dot{Y}_L)$) and $f_{\bar{i}\bar{j}} + f_{\bar{j}\bar{i}}$ in the reverse strand (i.e. $(\dot{X}_{L+1}, \dot{Y}_{L+1}) \dots (\dot{X}_{2L}, \dot{Y}_{2L})$). The overall reduction is the average of the reduction in the forward and reverse strands, which is $\frac{1}{2}(f_{ij} + f_{ji} + f_{\bar{i}\bar{j}} + f_{\bar{j}\bar{i}})$. As a result, $\hat{d}_{AG} = \hat{d}_{CT}$ and $\hat{d}_{AC} = \hat{d}_{GT}$. The P, Q_1, Q_2 and R terms in (6.2) are estimated from 2-way genome using:

$$\begin{aligned} P &= 2\hat{d} - 2\hat{d}_{AG} \qquad & Q_1 &= \hat{d} - \hat{d}_{AT} \\ R &= 2\hat{d} - 2\hat{d}_{AC} \qquad & Q_2 &= \hat{d} - \hat{d}_{CG} \end{aligned}$$

Thus, we need to compute only five distance values from the data, $\hat{d}$, $\hat{d}_{AC}$, $\hat{d}_{AG}$, $\hat{d}_{AT}$, and $\hat{d}_{CG}$ in addition to an estimation of ω.

Although four TK4 terms P, Q_1, Q_2, and R can be determined independently given the estimate $\hat{d}$, they must satisfy the constraint $P + Q_1 + Q_2 + R = \hat{d}$. Thus, $\hat{d} = 2\hat{d} - 2\hat{d}_{AG} + \hat{d} - \hat{d}_{AT} + \hat{d} - \hat{d}_{CG} + 2\hat{d} - 2\hat{d}_{AC}$. Since all five estimated values $\hat{d}$, $\hat{d}_{AC}$, $\hat{d}_{AG}$, $\hat{d}_{AT}$, and $\hat{d}_{CG}$ are empirical, it cannot be ensured that this equation will be satisfied. In other words, the system of equations has one excess observation. Among the five, the distance with no replacements $\hat{d}$ is always the largest, i.e. has the lowest containment Jaccard index. For large distances, the containment Jaccard can be zero, which prohibits computing any evolutionary distance (JC or TK4) from the data. In order to increase the distance upper-bound of $TK4$ model, we opt to reduce the number of free variables in the system by computing $\hat{d}$ from $\hat{d}_{ij}$, not directly from

data. More precisely,

$$\begin{aligned}\hat{d} &= 2\hat{d} - 2\hat{d}_{AG} + \hat{d} - \hat{d}_{AT} + \hat{d} - \hat{d}_{CG} + 2\hat{d} - 2\hat{d}_{AC} \\ &= (2\hat{d}_{AG} + 2\hat{d}_{AC} + \hat{d}_{AT} + \hat{d}_{CG})/5\end{aligned} \tag{6.10}$$

This equation can be used to compute JC model distances using (6.1). We also use it to calculate P, Q_1, Q_1, and R as a linear combination of four $\hat{d}_{ij}$ distances calculated using (6.6) after replacement:

$$\begin{aligned}P &= (-6\hat{d}_{AG} + 4\hat{d}_{AC} + 2\hat{d}_{AT} + 2\hat{d}_{CG})/5 \\ R &= (4\hat{d}_{AG} - 6\hat{d}_{AC} + 2\hat{d}_{AT} + 2\hat{d}_{CG})/5 \\ Q_1 &= (2\hat{d}_{AG} + 2\hat{d}_{AC} - 4\hat{d}_{AT} + \hat{d}_{CG})/5 \\ Q_2 &= (2\hat{d}_{AG} + 2\hat{d}_{AC} + \hat{d}_{AT} - 4\hat{d}_{CG})/5\end{aligned} \tag{6.11}$$

6.2.5 NSB: TK4 distance estimation using k-mers

Algorithm 6.1 combines results in the previous sections into a three-step process (Fig. 6.6) for estimating phylogenetic distances under the TK4 model. We implemented the algorithm using Python in a method called the NSB (No Strand-Bias) distance estimator. In its first step, NSB adds the reverse complement of all input sequences. It then builds separate k-mer libraries for each of the inputs using a left/right encoding scheme where nucleotide bases A, C, G and T are represented as 2-bit numbers, thus requiring 64-bit integer for $k <= 32$. NSB then builds base substituted encoded k-mer libraries from the initial encoded library by replacing the encoded bits of base i with the encoded bits of base j, for $(i, j) \in \{(A,C),(A,G),(A,T),(C,G)\}$. Thanks to a Left/Right encoding scheme, a replacement operation on an array of k-mers can be computed rapidly using fast and vectorized bitwise operations such as XOR, AND, and Shift (e.g., see A_to_C function in Algorithm 6.1). Finally, NSB computes the Jaccard indices for 4 pairs of base-substituted encoded libraries by computing the cardinality of the intersection succeeded by containment Jaccard correction. In practice, input genomes are almost never the same size and

never follow the same base frequencies. When computing $\mathbb{E}[\tilde{\mathrm{C}}_{ij}]$ using Lemma 5, $P(r \in s(\mathscr{X}))$ and $P(r \in s(\mathscr{Y}))$ are computed using L and ω of the respective genome for a given k-mer r. In the final stage, we estimate the phylogenetic distance of each pair of genomes using Equation (6.2). Various components in this equation are calculated using the equations (6.8)–(6.11). L and ω are set to the average of the two input genomes. When input data is unassembled (reads), we run Skmer prior to NSB to obtain L, coverage, and sequencing error rate. Computing the cardinality of the intersection between two encodings of size N takes $O(N\log(N))$ time and $O(N)$ memory. Therefore, time and memory complexity of Algorithm 6.1 are $O(n^2N\log(N))$ and $O(N)$ since no more than two encodings are loaded into the memory simultaneously.

6.3 Validation Results

We validate NSB in simulations and on real data. and compare it to three methods on assembled data. NSB-JC is JC distance computed using (6.10) and (6.1) with our tool. We also test using Jellyfish (2.3.0) and Skmer (3.1.0) to estimate containment Jaccard index and subsequently JC distance using (6.1) and (6.6). Jellyfish computes Jaccard exactly and Skmer approximates it using 10^5 sketches. On genome skims, we compare NSB-TK4 to Skmer.

6.3.1 Simulation study

Simulating genome sequences under the TK4 model

We use our own procedure to simulate pairs of genomes evolved under the TK4 model with controlled levels of distance and model parameters (https://github.com/balabanmetin/tk4-evol-sim). First, we either use a real genome as the ancestral genome or simulate one by drawing each site randomly from π with user-defined ω. We simulate two separate genomes from the ancestral genome by introducing substitutions at random positions. The frequency of each substitution type is determined by the TK4 model transition probability matrix $\mathbf{P}$ and half of the targeted distance $t/2$, producing two genomes with the evolutionary distance t. We create two simulated datasets. The first dataset uses a randomly generated $100Mbp$ base genome with

Table 6.1. NSB: TK4 Distance estimation. Notations: We denote the set of all reference sequences by S. NSB first runs PREPROCESS procedure, which itself uses ADD_RC to add the RC of genomes. It then calculates pairwise distances of the sequences according to the PAIRWISE-DIST procedure. BG_INTERSECT computes expected number of background matches after replacement the using equation (6.9).

PREPROCESS(S)
for $G \in S$ **do**
  $E \leftarrow$ENCODE(ADD_RC(G))
  for $(i,j) \in \{(A,C),(A,G),(A,T),(C,G)\}$ **do**
    $E_{ij} \leftarrow \emptyset$
    for $e \in E$ **do**
      $E_{ij} \leftarrow E_{ij} \cup \{$ i_TO_j(e) $\}$
    end for
  end for
  Save $\{E_{AC}, E_{AG}, E_{AT}, E_{CG}\}$ to disk
end for

ENCODE(G_{2way})
$E \leftarrow \emptyset$
for k-mer $a \in G_{2way}$ **do**
  $e \leftarrow 2k$ bit zeros
  for letter $l_i \in a$ **do**
    $e_i \leftarrow 1$ if $l_i \in \{C,G\}$
    $e_{i+k} \leftarrow 1$ if $l_i \in \{A,G\}$
  end for
  $E \leftarrow E \cup \{e\}$
end for
return E

A_TO_C(e) # an example of i_TO_j function
$mask \leftarrow 2^k - 1$
$e_1 \leftarrow$ first k bits of e
$e_2 \leftarrow$ last k bits of e
$e_3 \leftarrow e_2$ & $(e1 \oplus mask)$
$e_1 \leftarrow e_1 \oplus e_3$
$e_2 \leftarrow e_2 \oplus e_3$
return $2k$ bits $((e_1 << k) + e_2)$

PAIRWISE-DIST(G_1, G_2)
for $(i,j) \in \{(A,C),(A,G),(A,T),(C,G)\}$ **do**
  $(E_{ij,1}, E_{ij,2}) \leftarrow$ Read (G_1, G_2) from disk
  $D_{ij} \leftarrow$ G_DIST($E_{ij,1}, E_{ij,2}, L_1, L_2, \omega_1, \omega_2$)
end for
return CLC-TK4-DIST($D_{AC}, D_{AG}, D_{AT}, D_{CG}$)

CLC-TK4-DIST($D_{AC}, D_{AG}, D_{AT}, D_{CG}$)
$D \leftarrow (2D_{AG} + 2D_{AC} + D_{AT} + D_{CG})/5$
$P \leftarrow D - D_{AG}$
$Q_1 \leftarrow D - D_{AT}$
$Q_2 \leftarrow D - D_{CG}$
$R \leftarrow D - D_{AC}$
$\omega \leftarrow {}^{(\omega_1+\omega_2)}/_2$
$S_1 \leftarrow \omega - {}^{(P+R)}/_2 - Q1$
$S_2 \leftarrow 1 - \omega - {}^{(P+R)}/_2 - Q2$
return TK4 distance using Equation (6.2)

G_DIST($E_{ij,1}, E_{ij,2}, L_1, L_2, \omega_1, \omega_2$)
$I \leftarrow |E_{ij,1} \cap E_{ij,2}|$
$I_c \leftarrow$ BG_INTERSECT($i, j, L_1, L_2, \omega_1, \omega_2$)
$C \leftarrow {}^{2(I-I_c)}/_{(L_1+L_2)}$ # Containment Jaccard
return $1 - (C)^{\frac{1}{k}}$

$\omega = 0.6$. The second dataset uses a real assembled genome of *Saccharomyces arboricola* (11 Mbp) as the base sequence. The base frequencies of the available *S. arboricola* genome are $\pi_A \approx \pi_T \approx 0.307$ and $\pi_C \approx \pi_G \approx 0.193$, which follow the assumptions of TK4 with $\omega = 0.614$. We set the parameters of the TK4 model according to Fig. 6.1, exploring eight values of α, δ, and γ. Recall that ${}^{\delta}/_{\alpha} = {}^{\varepsilon}/_{\beta}$ and $\omega = \frac{\beta}{\beta+\alpha} = \frac{\varepsilon}{\varepsilon+\delta}$, leaving us with only three free parameters for a fixed w. We generated eight model conditions with different TK4 parameters (Table D.1) chosen

to include cases with both minimal and substantial deviations from the JC model based on the earlier calculations (Fig. 6.2). For each model condition, we simulated genome sequences with true distances $t \in \{0.01, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5\}$, each with 10 replicates, covering a range of both short and long distances.

Results on simulations under the TK4 model

Random base genomes.

When input genomes are generated in the i.i.d. fashion assumed by both evolutionary models, across all model conditions and regardless of the true phylogenetic distances t, the distances estimated by NSB-TK4 are highly accurate (Fig. 6.3). In contrast, JC distances are accurate when the true distance t is low but are under-estimated when t increases. In the most challenging case, $t = 0.5$, NSB-TK4 deviates only 0.3% from the true value on average compared to 7.8% for Jellyfish-JC. The error of Jellyfish-JC is as high as 18% when $\gamma = 32$, which causes extreme deviations from JC. The best performance of JC is when all parameters except ω follow JC. As models become successively more deviant from JC assumptions, the accuracy of JC diminishes.

Finally, comparing the two ways of obtaining JC distances, for $t \leq 0.3$, the approximate Skmer distances are slightly *more* accurate than Jellyfish. However, when $t > 0.3$, Skmer distances become less accurate. When true distance $t \geq 0.4$, Skmer fails to estimate distances in some cases (most cases for $t = 0.5$) because the true Jaccard index becomes too small (e.g., $< 10^{-5}$) to compute reliably with sketches of size 10^5.

Yeast-base simulations.

The TK4-based calculations show improvements over JC computed using NSB or Skmer across some model conditions except for $\delta = \gamma = 1$ that resembles JC (Fig. 6.3). However, the comparison to JC computed exactly (using JellyFish) is more complex. When deviations from JC are relatively low, JellyFish-JC can be as accurate or even more accurate than NSB-TK4. It is only with higher levels of deviation from JC that improvements of NSB-TK4 over JC are clear.

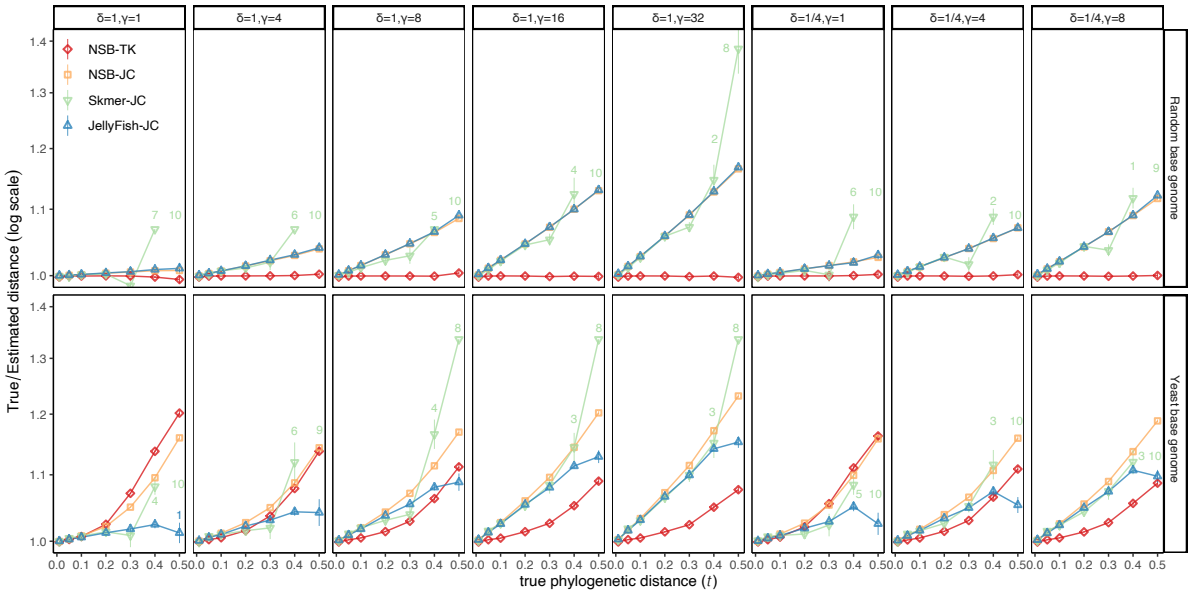


Figure 6.3. Comparing the accuracy of distances estimated by different approaches on random and Yeast-based simulated genomes. Genome sequences were simulated by randomly substituting the genome skims of *Saccharomyces arboricola* (11 Mbp) and a random 100 Mbp sequence with eight different sets of TK4 parameters and with seven controlled true distances. Here, ω is fixed, and since these rates do not have a scale, $\alpha = 1$ in all cases. We show the average true distance divided by estimated distances (y-axis) with standard errors (over replicates, requiring at least two) against the true distances. Annotated numbers show the number of replicates out of 10 where Skmer or JellyFish return infinity. See Fig. 6.9 for linear scale.

Regardless of simulation parameters, phylogenetic distances $t \leq 0.1$ are estimated with high accuracy under both TK4 and JC models. However, the JC model starts to underestimate the distance as we increase the distance t, and the underestimations are substantial when $t \geq 0.3$. Moreover, the JC error is not linear or even monotonically increasing with respect to t, meaning that the distance matrices obtained from the JC model may not be additive. When t is increased to 0.5, TK4-based distances tend to have reasonable accuracy with a few exceptions (e.g., for $\gamma = 8$). In some cases, TK4 distances have more than 10% error with increased t and in three conditions are consistently less accurate than JC. Comparing the results to random base genomes, the reduced accuracy of TK4 on these conditions has to be due to violations of the model in the base genome, a point that we will return to in the discussion section.

Finally, We explore the impact of the choice of k-mer size in accuracy of tested methods.

We select the simulated yeast genomes with a fixed model-condition $\delta = 1$ and $\gamma = 4$ and test $k \in \{21, 23, 25, 27, 29, 31\}$ over ten replicates. We do not explore $k > 31$ because Jellyfish and NSB do not support it. There is no single k value that performs universally better than others (Fig. 6.8); the choice depends on the distance and the method. For $k = 21$, NSB-TK4 overestimates or underestimates the true distance when $d \leq 0.1$ or $0.1 < d \leq 0.4$ respectively. On the other hand, for d values larger than 0.4, NSB-TK4 does not return a valid distance due to overestimation of number of background matches. As k increases, distance estimation using NSB-TK4 becomes more accurate, reaching peak accuracy in $k = 31$ setting. More generally, NSB-TK4 and NSB-JC are more sensitive to the selection of k than Jellyfish and Skmer. For example, when $d = 0.4$, the estimation error difference between the most and the least accurate estimates are 13.7% ($k = 31$ and $k = 23$), 12.5% ($k = 31$ and $k = 23$), 6.3% ($k = 27$ and $k = 31$), and 1.2% ($k = 31$ and $k = 21$) for NSB-TK4, NSB-JC, Skmer-JC, and Jellyfish-JC respectively. Given the totally of results, we recommend setting $k = 31$ for NSB-TK4.

Simulation of genomes and phylogenies under the GTR model

In order to compare the performance of TK4 and JC models under the presence of model misspesification, we simulate an 8-taxa dataset with genomes that are evolved under the GTR [232], which can substantially violate assumptions of both JC and TK4 models. Of the 120 fully-balanced and caterpillar tree topologies simulated by Rachtman et al. 194 using Simphy [149], we first proceed with taking the first 20 for each category. In these 8-taxa trees, branch lengths are randomly selected from the log-uniform distribution ranging between 0.00001 and 0.12. Next, we simulate 10Mbp genome sequences using INDELible [70] with following model parameters. Base frequences of the GTR model follow $\{\pi_A = \omega/2, \pi_C = (1-\omega)/2, \pi_G = (1-\omega)/2, \pi_T = \omega/2\}$ where ω is a random variable from $Beta(30, 21)$ distribution. Other entries of the GTR matrices are drawn from Dirichlet distribution with parameters $(50, 7, 12, 12, 14, 50)$ corresponding to $C \leftrightarrow T$, $A \leftrightarrow T$, $G \leftrightarrow T$, $A \leftrightarrow C$, $G \leftrightarrow C$, $G \leftrightarrow A$. Each method produces an 8×8 distance matrix, which is then given to FastME 2.0 [123] to estimate the phylogeny.

Since we have a tree, we compare the methods by measuring Robinson and Foulds 201 (RF) distance between the true tree and the inferred tree. Beyond topological accuracy, we quantify the divergence of the TK4 and JC distances from the additivity using the Fitch and Margoliash 68 (FME) weighted least squares error Since FME metric weights distances by $\hat{t}^{-2}$, it is insensitive to the unit and scale of branch lengths. When measuring the FM metric, we use the combination of true tree topology and estimated distances, which ensures measurements across different methods are based on the same (true) tree.

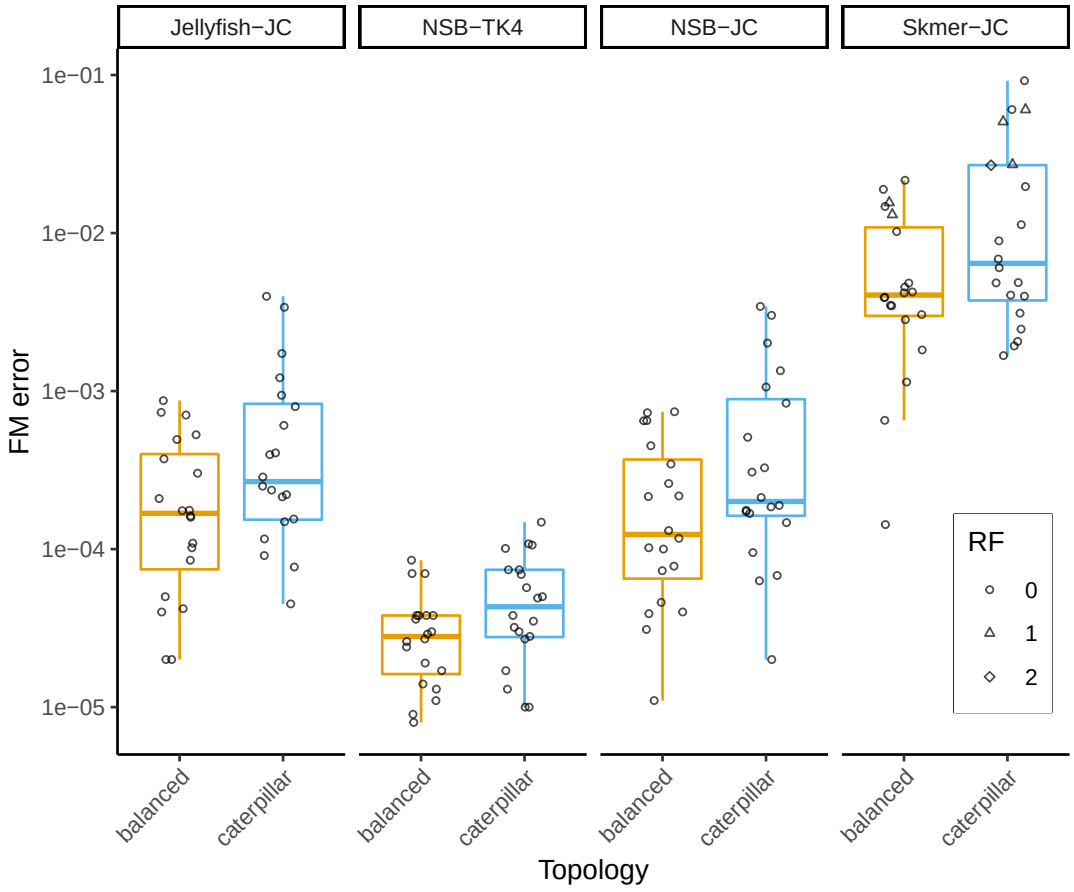


Figure 6.4. Deviation from additivity measured for TK4 and JC models of evolution on the dataset of 40 8-taxa phylogenies simulated under GTR model. The dataset consists of 20 balanced and 20 caterpillar tree topologies. Whiskers in the boxplot demonstrate the range between the first and third quartiles. Point shape represents the RF distance between the constraint-free tree inferred by the method and the true tree. Y-axis is in log-scale.

Results on phylogenies evolved under the GTR model

Topological accuracy remains high despite model misspecification. NSB-TK4, NSB-JC, and JellyFish-JC infers the correct topology in all 40 cases whereas Skmer-JC is erroneous in 6/40 trees tested. The mean FME error of NSB-TK4 (4e-05) is an order of magnitude lower than those of NSB-JC and Jellyfish-JC (5e-04), which have near identical levels of error (Fig. 6.4). Therefore, in simulations, TK4 model produces distances that are closer to additivity than JC when model misspesification is present. However, Skmer-JC has 27 times higher error than the other two JC-based methods, indicating that the sketching process affects accurate distance estimation in a higher degree than model misspesification. Finally, we observe that regardless of the method used, the 20 replicates with balanced topologies tend to have lower deviations from additivity than those based on unbalanced topologies.

6.3.2 Evaluation on biological bacterial data

Dataset.

We created a dataset consisting of 10 clades of microbial species subsampled from the Web of Life (WoL) [263] ASTRAL tree of 10575 Bacteria and Archaea taxa. We started with finding all the clades with 30 to 50 leaves and with 0.2 to 0.7 diameter (the largest pairwise tree distance between any pair). We then selected the top 25 clades with the highest local posterior probabilities and for each clade, computed an all-pairwise distance matrix using Skmer (sketch size 10 million), inferred a phylogenetic tree using FastME 2.0, and computed the RF distance between the WoL ASTRAL reference tree and the inferred tree. We then selected nine clades with the lowest RF distance, and these clades had 32 to 46 species and RF distance between 0.16 and 0.42. As none of the nine selected clades had any missing data in their distance matrix, we also curated a challenging subtree with 86 taxa from the *Erysipelotrichaceae* family from the WoL reference tree that contained 114 missing data entries in its distance matrix (RF distance: 0.43) computed using Skmer.

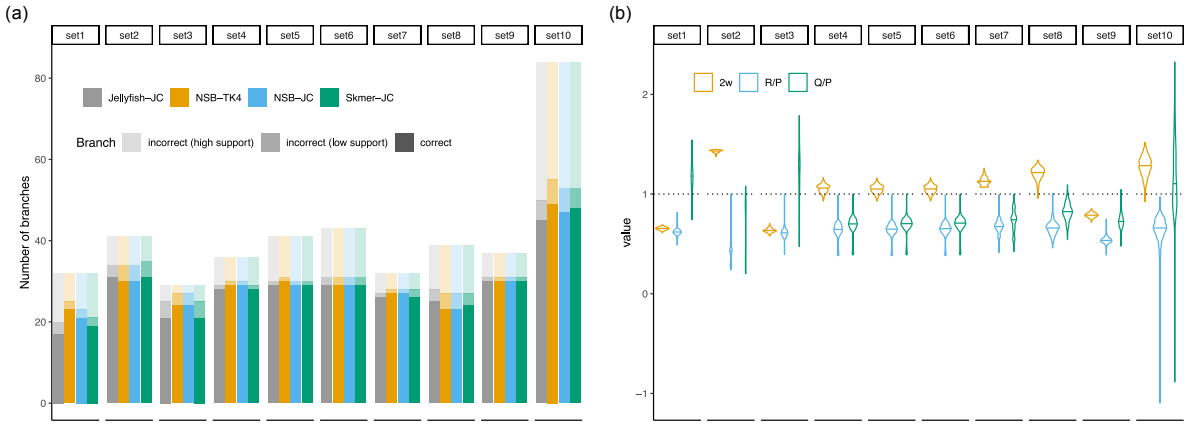


Figure 6.5. (a) Comparison of different methods to the ASTRAL tree on 10 subsets of the bacterial dataset. We show the number of branches in the reference tree that are correctly estimated or are incorrectly estimated and have low (less than 0.95) or high support in the reference tree. (b) TK4 model parameters inferred using NSB-TK4 for each set. Deviations from $y = 1$ indicate violation of the JC model. $Q = Q_1 + Q_2$.

Results on Bacterial Dataset

On the 10 bacterial data sets, while methods are generally competitive (Fig. 6.5a), overall, NSB-TK4 is better than others as it produces the best result in 8 datasets out of 10. The total number of missing branches for NSB is 120 (out of 403) (Table D.2), which is lower than Jellyfish, with 133 missing branches. Results are similar when focusing on highly supported branches: NSB-TK4 misses 95 out of 374 branches with at least 0.95 support while Jellyfish misses 109. Among the three methods that compute JC distances, NSB-JC is the most accurate, matching or improving on Jellyfish and Skmer in seven out of 10 cases, and with eight and four fewer wrong branches, respectively. On the most challenging case (Set 10), the distance matrix produced by NSB-TK4 contains 20 fewer missing entries (infinity) than both Jellyfish-JC and Skmer-JC. As a result of its replacement technique, NSB can compute distances where other tools cannot. To perform a tree inference on distance matrices with missing data, we impute the missing distances using a machine-learning-based algorithm [24]. Here, NSB-TK4 distances produce the tree with the fewest differences to the reference phylogeny compared to JC-based tools.

Jellyfish-JC had between 7% and 57% (mean: 22%) higher FM error than NSB-TK4 across datasets (Table. D.3). NSB-TK4 distances are not only more additive but also on average 13% and 32% larger than those of Jellyfish and Skmer, which may under-estimate the distances.

TK4 model parameters inferred by NSB-TK4 demonstrate that JC model assumptions are significantly violated in the real data (Fig. 6.5b). For instance, 2ω, assumed to be 1 in the JC model, is as low as 0.65 on average across all pairs in a set. In addition, transversion to transition ratios R/P and $(Q_1 + Q_2)/P$ are less than 1 in almost every case, in clear violation of the JC model; thus, NSB captures the long understood [253] divergence of transversion and transition rates.

6.3.3 Evaluation on biological Yeast dataset

We also study the yeast dataset used by Balaban and Mirarab 10, consisting of eight genomes (Table C.2) with sizes in the 10.9 – 12.4Mbp range and number of scaffolds in varying between 16 and 2808. We use ART v2.5.8 [92] to create in-silico genome skims of 150bp reads with Illumina HiSeq 2500 error profile. We test for 1, 2, 4, and 8$\times$ sequencing coverage levels. We use the yeast phylogeny published by Shen et al. [216] as the reference and compare it to alignment-free trees inferred under TK4 and JC models using FastME 2.0.

When analyzing Yeast assemblies, NSB-TK4 and Jellyfish-JC produce a phylogenetic tree that is identical to the reference phylogeny (Fig. 6.10). However, Skmer-JC distances produce a tree with one branch mismatch. Although the trees inferred using NSB-TK4 and Jellyfish-JC distances are topologically identical, their branch lengths differ: NSB-TK4 trees have 16% increased tree height (Fig. 6.11), indicating that JC model likely under-estimates distances. In terms of additivity, Jellyfish-JC distances have an FME of 0.0034, which is 70% higher than that of NSB-TK4 (Table D.5).

When analyzing the genome skims, the tree inferred by NSB-TK4 and Jellyfish-JC is identical to the reference phylogeny regardless of the sequencing coverage (Fig. 6.10). Similar to assemblies, NSB-TK4 and Jellyfish-JC are able to recover the reference phylogeny on Sac-

charomyces genome skims for all levels of coverage (Table D.5). While Skmer-JC can match the reference phylogeny on the genome skim of $2\times$ coverage, the Skmer tree has one branch mismatch in other coverage levels. On yeast genome skims, NSB-TK4 consistently achieves the lowest FM error among the three methods tested. Furthermore, even on the shallowest genome skim data ($1\times$) tested, the NSB tree achieves a lower FM error than JC based method on assembled data. In contrast to NSB and Jellyfish, Skmer-JC trees have higher FM error with increasing coverage. Nevertheless, at $8\times$ coverage where most k-mers in the genomes are covered by at least one read, all three methods seem to approximate their level of an error on the assembled data.

6.4 Discussion

We introduced a method for computing phylogenetic distances on alignment-free data based on the time-reversible, no strand-bias, four-parameter evolutionary model, TK4. Through theoretical and empirical analyses, we explored the model conditions where the more general model TK4 offers more accurate distances than the Jukes-Cantor model, which is the simpler yet most widely used model. As expected, the improvements are most pronounced for larger distances and more substantial deviations from the JC model assumptions.

Despite overall improvements, in the simulations based on the yeast genome, we observed conditions where the TK4 model was less accurate than the JC model it contains. Deviations from the TK4 model can explain this surprising result. Even if used as the base genome for subsequent simulations, the real genomes can violate the assumptions of our algorithm in several ways. *i)* Presence of non-randomly generated repeats (e.g., recent gene duplications) causes overestimating the Jaccard index. The probability of a k-mer being present in both input genomes is higher when it repeats multiple times across the genome. Our calculations only correct for these repeats when they occur randomly but not by homology. *ii)* Systematic variations of ω across the genomes, violating i.i.d. assumptions, can create loci with increased numbers of

homologous and non-homologous matches after replacement. *iii)* Presence of *k*-mer motifs can invalidate assumptions of Lemma 5. While some of these issues also violate JC assumptions, NSB-TK4 may be less robust to these violations than JellyFish-JC due to the more complex equations or the more complex estimation procedure (e.g., letter replacement) used by NSB.

More broadly, while the TK4 model is more complex than JC, relevant processes are also missed by TK4. An important aspect of molecular evolution we did not model is the rate heterogeneity among sites. Leading alignment-based phylogenetic estimation tools model the heterogeneity using a discrete or continuous gamma distribution. JC model can be extended to support Gamma-distributed rates [104] if the parameters of the Gamma model are known. With GTR-based simulations, we showed that TK4 is robust to model misspecification. One question is whether TK4 distances are accurate in data simulated under GTR+Γ model of evolution. Furthermore, it may be possible to incorporate a measure of rate variation in the TK4 formula (6.2) as well. We leave these questions to future work.

NSB is based on computing the containment Jaccard index between *k*-mer sets of the input genomes and their perturbations. This approach, previously utilized by tools such as Mash [179] and Skmer [209] for computing JC distances, has the potential to be applied to both assembled genomes and NGS reads in an assembly-free fashion, as we demonstrated. Interestingly, our results showed high levels of accuracy with very low coverage (e.g., 1X) in computing distances, as demonstrated by the low FME values obtained on the yeast dataset. Thus, beyond phylogenetic inference, other applications such as species identification using genome skims can benefit from NSB.

The use of *k*-mers is not the only option for distance calculations. For example, tools like pyANI [189] and Co-phylog [255] estimate the distance between two genomic sequences by efficiently finding local alignments. It is possible to infer substitution probabilities from these local alignments and calculate evolutionary distance according to the TK4 model. While such approaches will not be fully alignment-free, future work should compare these methods to our proposed approach. However, even if accurate, such methods cannot be incorporated into the

analyses of low-coverage short-read NGS data mentioned above when assembly is not possible.

In the scenario where assembly and alignment are available, NSB can be compared to the standard alignment-based methods for distance and phylogeny estimation. A careful comparison would require far more complex simulation pipelines—as our existing simulations do not handle indels and rearrangements. As stated earlier, alignment-free methods have the potential for improving accuracy when rearrangements make it hard to create reliable alignments; phylogenomic analyses often remove large chunks of the genome and focus on parts that are easier to align. If alignment-free methods can incorporate more complex models than currently possible, perhaps they can surpass alignment-based methods by using all of the data. We believe reaching that goal will require further increases in the model complexity of alignment-free methods.

Due to the exact computation of k-mer counts, NSB and JellyFish can both have substantial running times. Running time for NSB scales linearly with the input genome size (Fig. 6.12). On two random genomes of length 100Mbp, NSB completes within 11 minutes where 7 minutes is spent preprocessing the samples and computing the encodings and less than 4 minutes for computing all 4 Jaccard values and the pairwise TK4 distance. Running time for Jellyfish is about a quarter of NSB since it requires computation of a single Jaccard value. Jaccard indices can be estimated accurately without looking at all k-mers using the MinHash sketching technique [179] that dramatically improves the running time, disk space, and memory usage. For instance, for the fixed sketch size, Skmer completes under 15 seconds on the same two random genomes of length 100Mbp (Fig. 6.12). However, we saw that for large distances where Jaccard is small, MinHash sketching fails. This limitation may be alleviated with newer methods such as Dashing [9]. Nevertheless, for smaller distances where it is accurate, we could incorporate sketching into NSB. In preliminary tests, we saw that while the main Jaccard index is often computed accurately using sketching, the replaced Jaccard indices can have consequential error levels. This reduced accuracy is likely because hash functions used in existing tools assume four letters and need to be updated for genomes with replaced letters. It may even be possible to compute

all four Jaccard indices without actually replacing letters by defining hash functions that do not distinguish letters. Finally, NSB may be able to use compressed k-mer sets [197] to reduce its storage requirements while keeping the same accuracy. We leave the exploration of these avenues to further work.

6.5 Availability.

Software is available at https://github.com/nishatbristy007/NSB. Data are available at https://github.com/balabanmetin/yeast-genomes and https://github.com/balabanmetin/bac10.

6.6 Acknowledgement

Chapter 6, in full, has been submitted for publication of the material as it may appear in "Balaban, M., Bristy, N. A., Faisal, A., Bayzid, M. S., & Mirarab, S. (2022). Genome-wide alignment-free phylogenetic distance estimation under a no strand-bias model. Bioinformatics Advances". The dissertation author was the co-primary investigator and co-first author of this paper.

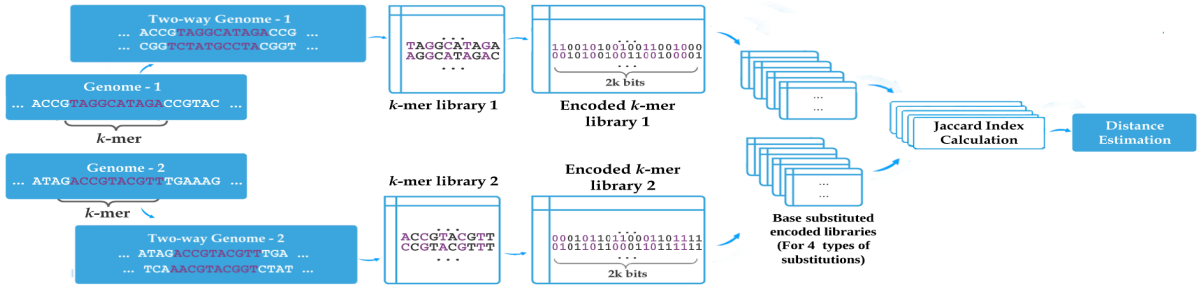


Figure 6.6. Schematic diagram of our proposed pipeline. We start with adding RC of sequences followed by decomposing them into fixed-length k-mers. Next, the bases in k-mers are encoded as bitmasks and thus an encoded k-mer library is constructed. For each $(i,j) \in \{(A,C),(A,G),(A,T),(C,G)\}$, we replace the encoded bits of base i with the encoded bits of j, producing 4 base-substituted encoded libraries. Finally, using these encoded libraries, Jaccard indices and distances are estimated under the assumptions of the TK4 model and using the equations derived and presented in Sec. 6.2.

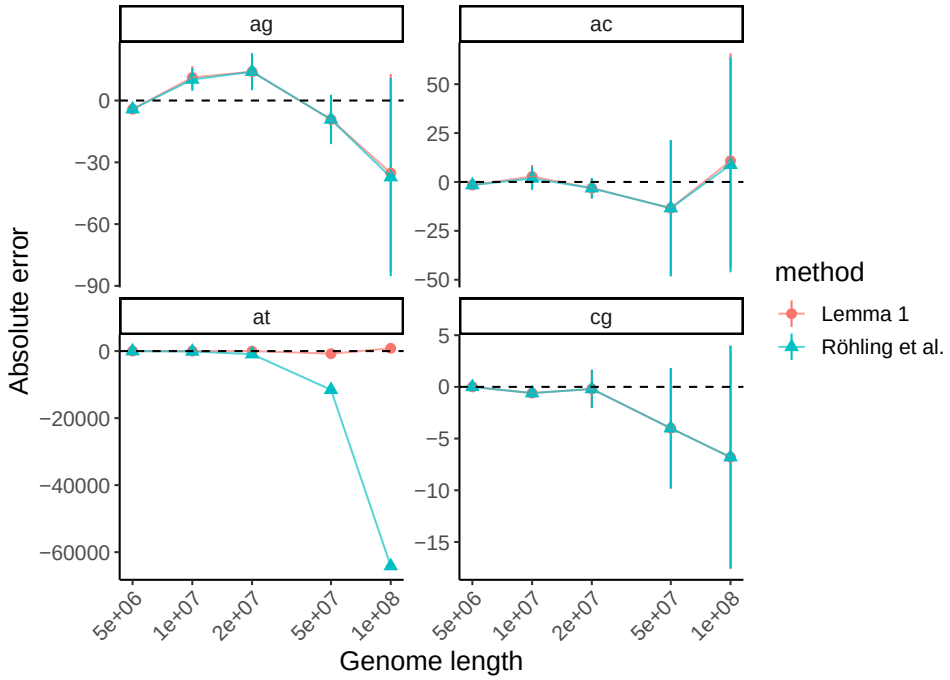


Figure 6.7. Estimating number of non-homologous (by random chance) matches between two genomes after replacements. Two input genomes are created according to an i.i.d. process with base frequencies $\pi = \left[\pi_A = 0.3 \quad \pi_C = 0.2 \quad \pi_G = 0.2 \quad \pi_T = 0.3\right]$. Replacements are performed after 2-way genomes are constructed.

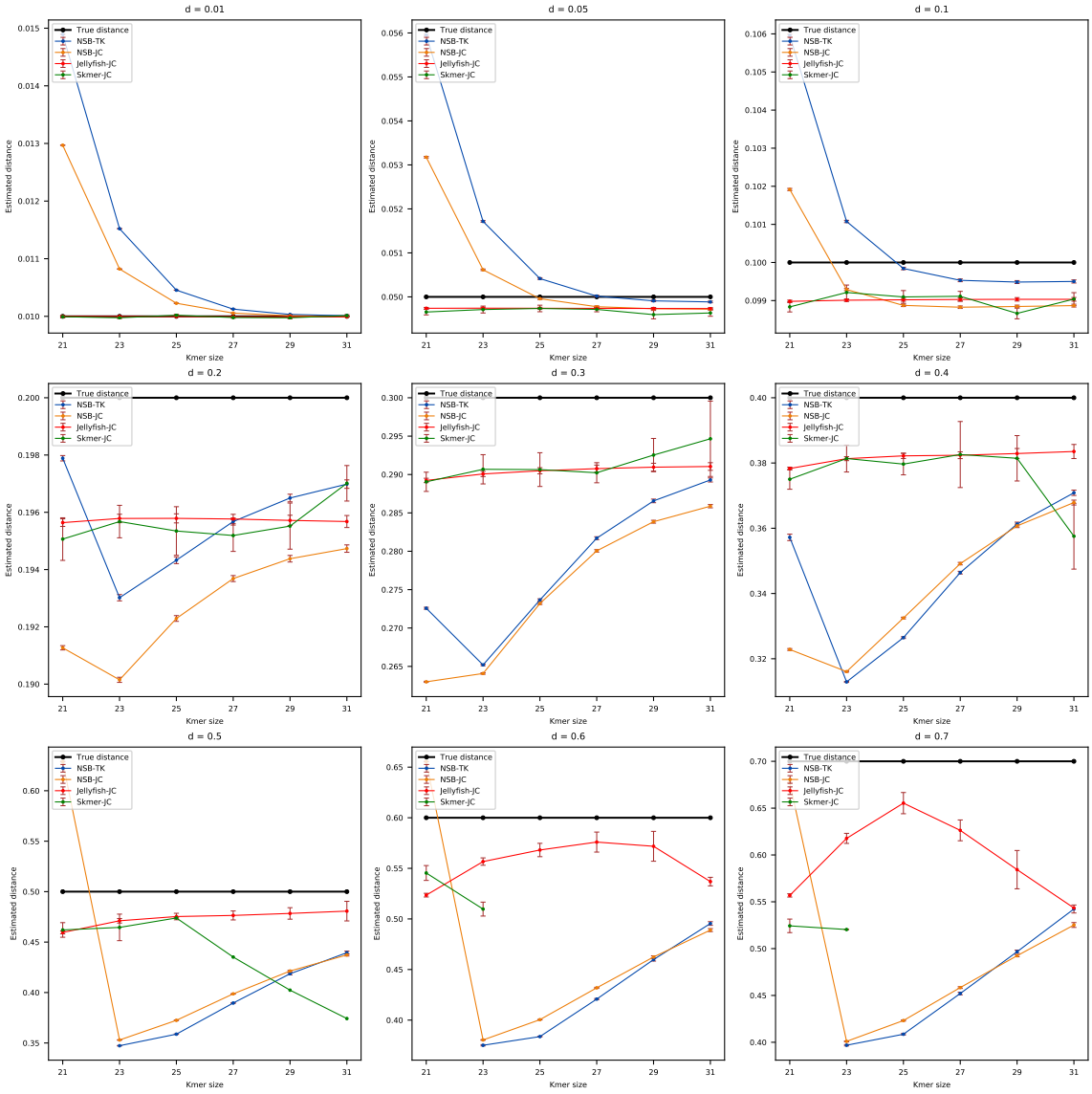


Figure 6.8. The effect of parameter k on accuracy in simulated yeast genomes. Genomes are simulated using TK4 model with parameters $\alpha = 1$, $\delta = 1$, and $\gamma = 4$. Skmer does not return valid distances in $d \geq 0.6$ and $k \geq 25$ setting. NSB-TK4 does not return valid distances in $d \geq 0.5$ and $k = 21$ setting. Error bars show confidence interval of the estimate over 10 replicates.

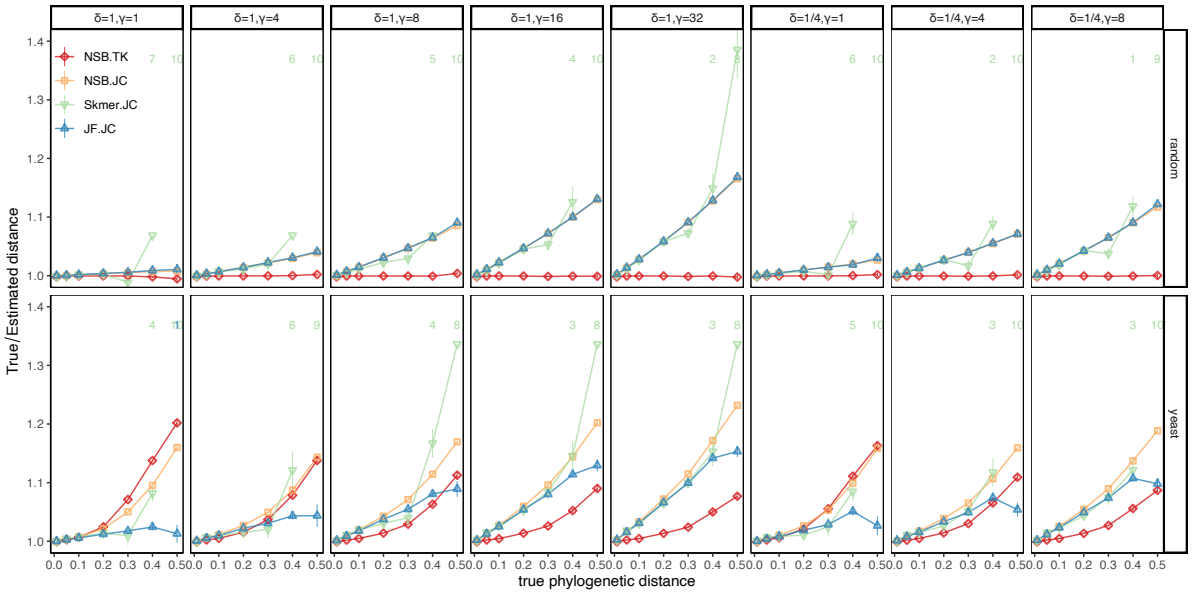


Figure 6.9. Comparing the accuracy of distances estimated by different approaches on random and Yeast-based simulated genomes. Genome sequences were simulated by randomly substituting the genome skims of *Saccharomyces arboricola* (11 Mbp) and a random 100 Mbp sequence with eight different sets of TK4 parameters and with seven controlled true distances. Here, ω is fixed, and since these rates do not have a scale, $\alpha = 1$ in all cases. We show the average true distance divided by estimated distances (y-axis) with standard errors (over replicates, requiring at least two) against the true distances. Annotated numbers show the number of replicates out of 10 where Skmer or JellyFish return infinity.

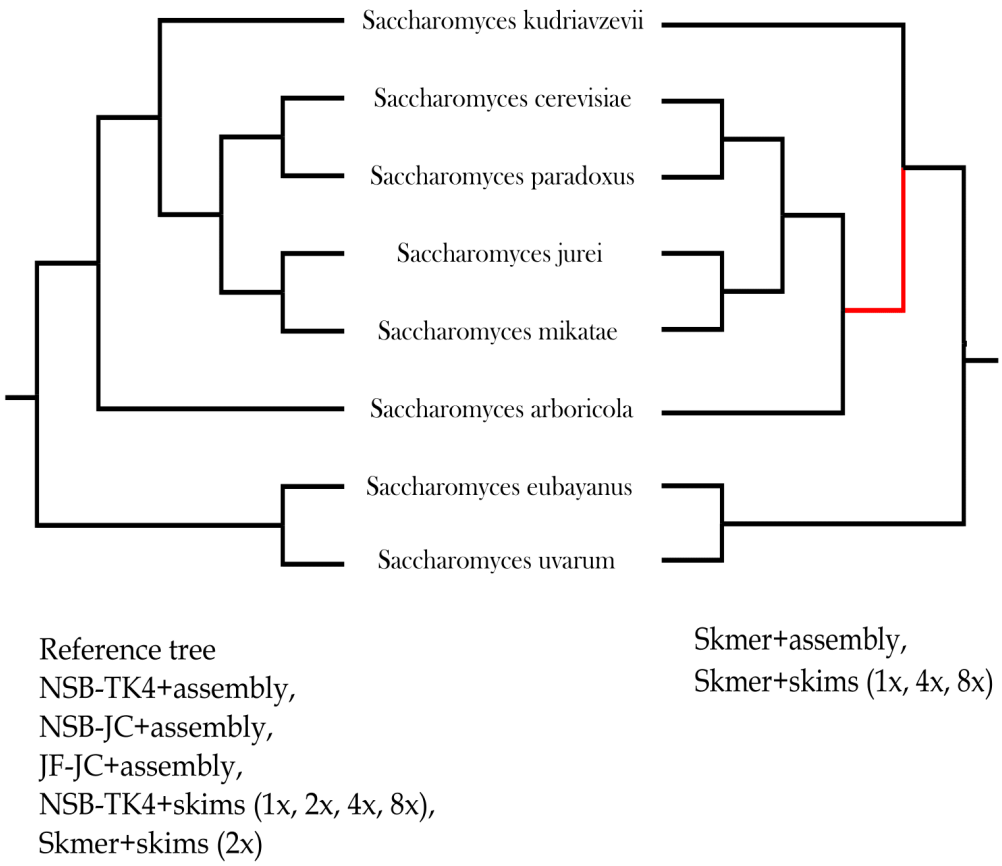


Figure 6.10. Analysis of the yeast genome assemblies and skims using distances estimated under JC and TK4 models. The branch in the estimated tree which is not found in the reference tree is shown in red.

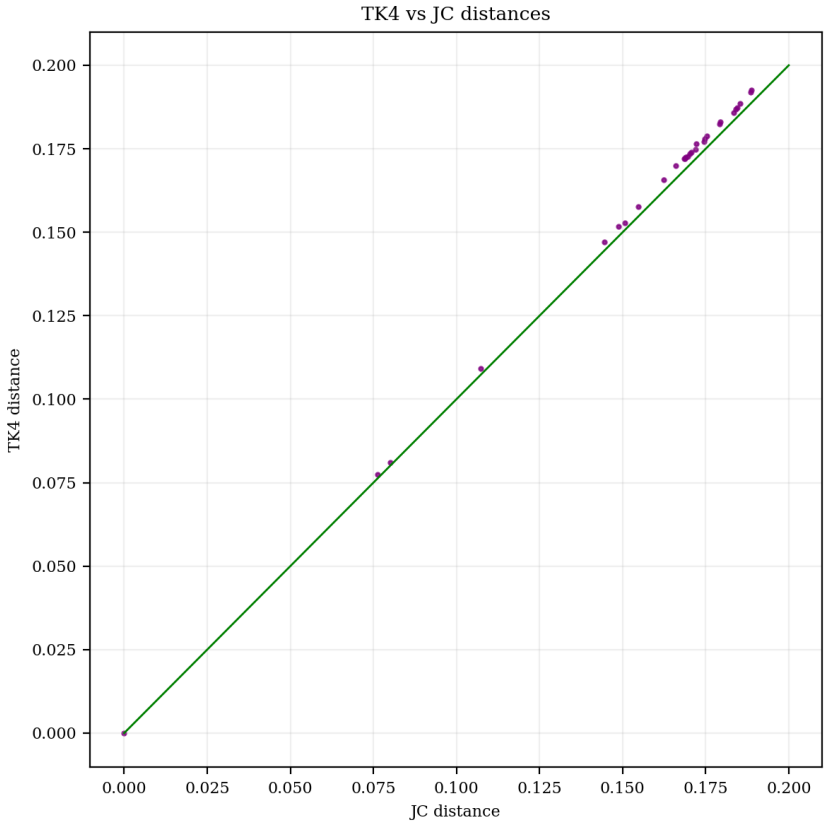


Figure 6.11. Comparison of pairwise distances estimated using TK4 and JC models on 8 real Yeast genomes. TK4 distances are always slightly higher than JC.

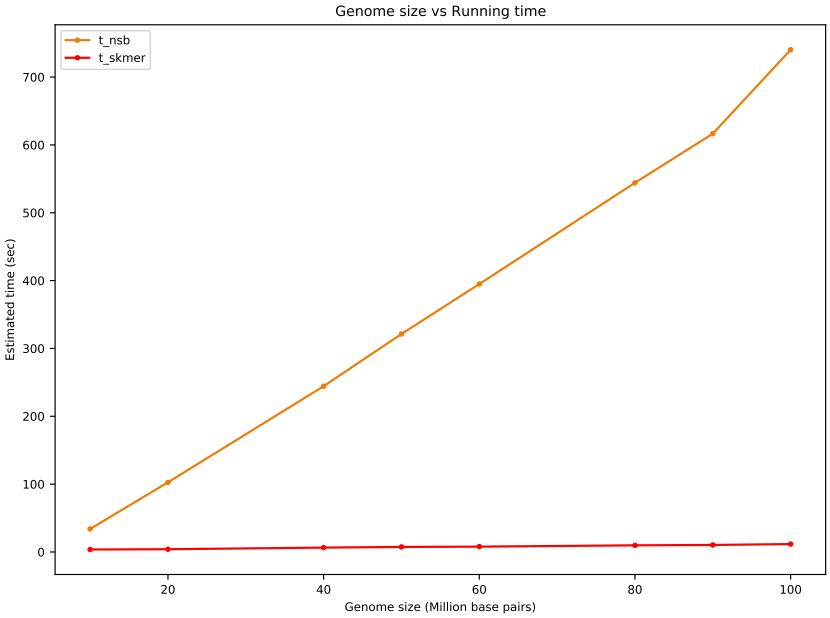


Figure 6.12. Runtime comparion between NSB and Skmer in seconds. Sketch size for Skmer is fixed.

Chapter 7

Growing phylogenomic trees at the ultra-large scale using divide and conquer

7.1 Introduction

Today, the number of Bacteria and Archaea species in the RefSeq database have already exceeded 60 thousand [81]. In addition, RefSeq contains more than a million prokaryotic whole genome sequences in the database. Cataloging the constantly growing number of microbial species is a challenging task. Reference phylogenies are increasingly used in microbiome profiling [76] and to answer the question: what is in my sample? Several databases based on single marker genes such as 16S [48, 190] consist of millions of sequences. On the other hand, when the number of genomes in the database is approximately equal, whole-genome based such as The Web of Life [263] and GTDB [183] reflect the evolutionary history of microbes more accurately because they are based on hundreds of marker genes. The tradeoff between the two paradigms is a large number of genes on one side and the number of taxa on the other. The main reason that the number of genomes in Web of life cannot match the number of sequences in the 16S database is the computational cost: it took 100000 CPU hours to infer this tree of life. Furthermore, ASTRAL, the coalescent-based species tree inference method employed by Zhu et al. [263] fails to scale on datasets with more than 16,000 genomes (according to our

benchmarks).

Several techniques can improve the scalability of phylogenomic inference. Divide-and-conquer schemes have been utilized to improve the efficiency and accuracy of phylogenetic inference (see the review [240]) in the past. Lately, phylogenetic placement has emerged as a scalable alternative to de-novo phylogenetic reconstruction. Balaban et al. [15] introduced improvements in time complexity and accuracy of distance-based phylogenetic placement and studied the feasibility and accuracy of whole-genome placement of microbial genomes and metagenome-assembled genomes (MAG). One of the strong results the authors presented was that using 381 marker genes sampled globally from the whole microbial genome, one can place a query genome onto the Web of Life phylogeny in under 16 seconds. It is therefore feasible to phylogenetically place all assembled microbial genomes uploaded to RefSeq onto the reference tree of microbes. There are two shortcomings of phylogenetic placement in comparison to de-novo phylogenetic reconstruction. First, the relationship between query sequences is unresolved. Secondly, the query sequences cannot refine and update the backbone. In this paper, we introduce a divide-and-conquer workflow for incrementally growing and updating ultra-large phylogenies. This workflow, called uDance, can build a 200,000-genome microbial tree of life from 388 marker genes, resulting in one order of magnitude improvement over the tree from the Web of Life in terms of the number of genomes. Using simulations, we show that uDance can infer de-novo and incremental whole-genome phylogenies with 10,000 taxa more efficiently and accurately than the alternative methods. uDance is publicly available on https://github.com/balabanmetin/uDance.

7.2 Results

7.2.1 uDance overview

We developed uDance, a highly distributed computation framework for incrementally growing and updating existing phylogenies using whole-genome data. uDance employs a divide-and-conquer strategy that enables processing different parts of the tree independently (Fig 7.1)

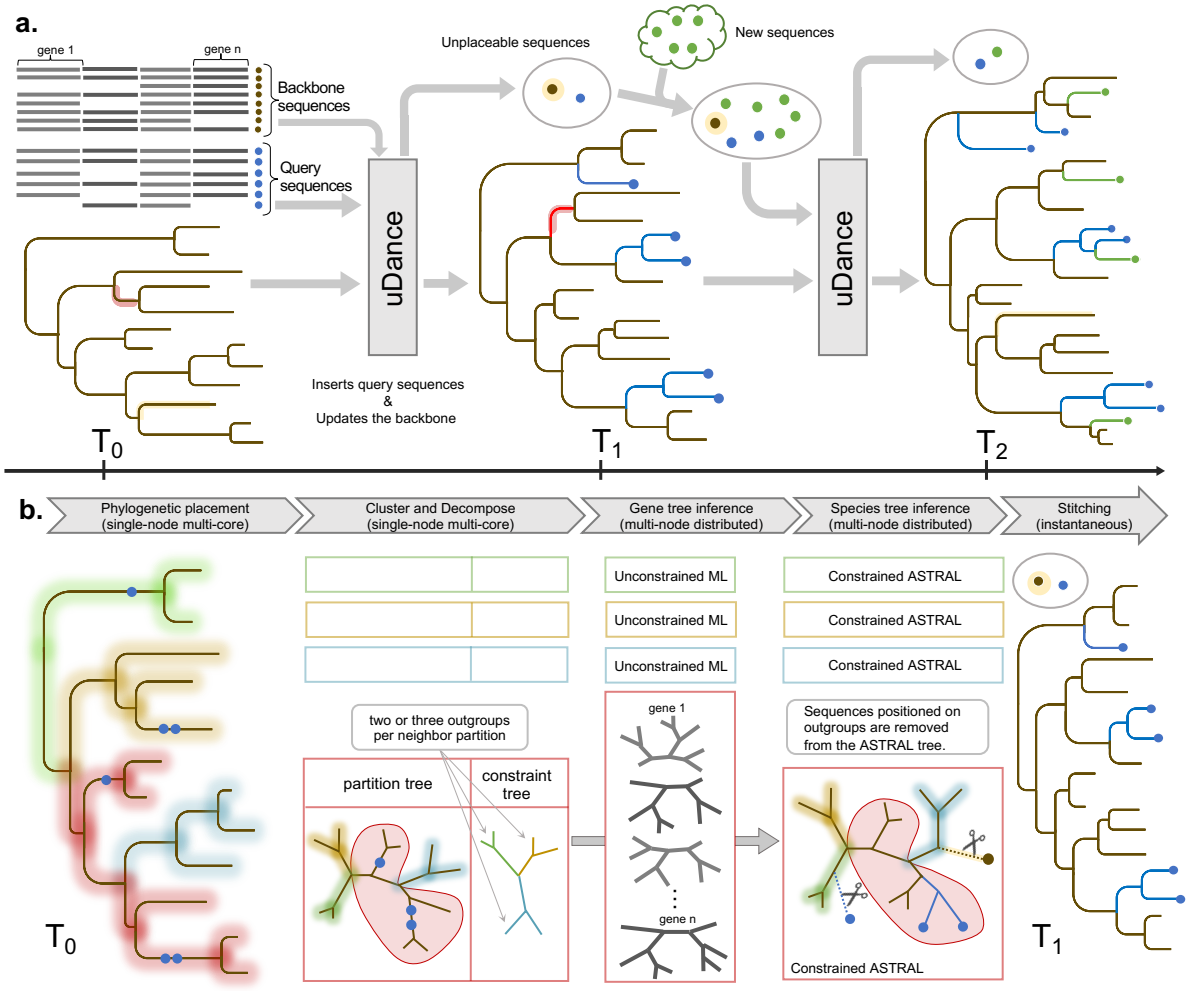


Figure 7.1. uDance overview.

and leverages the independence to achieve phylogenomic reconstruction at ultra-large scale (e.g., tends to hundreds of thousands of genomes) in reasonable time. uDance requires a backbone (reference) tree and a set of multiple sequence alignments (MSAs) of backbone sequences and new (query) sequences that do not appear in the backbone (Fig. 7.1a). If a backbone tree is not available, uDance can compute a set of backbone species with high diversity and reconstruct a backbone tree using whole-genome data (see Materials and Methods for details). uDance outputs a tree with branch lengths on the full set including backbone and query sequences. When yet newer sequences become available, the output phylogeny in the previous iteration can be used as the input in the next iteration to incrementally grow the phylogenetic tree.

We designed uDance to handle hundreds of aligned regions (called genes hereafter for brevity though these do not need to be functional genes) from input genomes. uDance first inserts the query sequences on the backbone tree independently and then refines the tree locally for each region with new sequences, allowing updates to the backbone based on the new information provided by the query sequences. It can alternatively fully preserve the backbone tree topology in the output (See Materials and Methods section), which may the desired when query data is noisy in comparison to the backbone. A main feature of uDance is its various quality control checks; at the end, it may decide that a (typically) small set of backbone and query sequences cannot be confidently in the output phylogeny.

The first step in uDance framework is phylogenetic placement of query sequences onto the backbone tree (Fig. 7.1b) based on the entire data set of gene MSAs. We accomplish this step using massively scalable APPLES-2 algorithm [15]. uDance then decomposes the resulting placement tree into multiple partitions (subtrees) using a novel clustering algorithm that we developed partially based on the TreeCluster [12] algoirthm. For every two adjacent partitions, the algorithm selects one or two representative sequences (See Materials and Methods for details) from each partition and adds them to the other one as outgroup sequences. The next step in the workflow is the inference of a maximum likelihood (ML) gene tree (i.e., RAxML-NG [115]) per gene on the set of backbone, query, and outgroup sequences in every partition.

The workflow proceeds with inferring a species tree per partition by summarizing all the gene trees in the partition using ASTRAL [165], constrained [192] to be compatible with a constraint tree designed for that partition: this constraint is built based on the backbone topology of all outgroup species and allows uDance to successfully stitch back the species trees into a single tree at the end (see Materials and Methods for details). During the stitching step, any backbone or query sequence that are positioned on and unexpected branches (e.g., within outgroup sequences) are removed from the final tree. The most time consuming steps of the workflow, gene tree and species tree inferences, can be performed in fully distributed fashion over multiple machines such as a high performance computing (HPC) cluster. Thus, uDance offers efficient phylogenomic reconstruction of large collection genomes.

7.2.2 Accuracy of uDance in Simulations

Table 7.1. Properties of simulated dataset.

Condition	k	ILS	HGT+ILS	GTEE	EGTD
mc1-full	500	0.03	0.35	0.33	0.50
mc-1	100				
mc2-full	500	0.03	0.37	0.44	0.58
mc-2	100				
mc3-100			0.27	0.43	0.52
mc3-200			0.32	0.44	0.55
mc3-300	100	0.03	0.37	0.43	0.57
mc3-400			0.41	0.43	0.60
mc3-500			0.54	0.48	0.71

Note: k, number of genes; ILS and ILS+HGT, average normalized RF distance between true genes trees and the true species tree due to ILS only (no HGT) and ILS with HGT, respectively; GTEE and EGTD, average normalized RF distance between gene trees estimated with FastTree-2 and the corresponding true gene trees and true species tree, respectively.

In order to validate accuracy and performance of our workflow, we simulated a dataset inspired by a published dataset [263] of 10,575 microbial genomes and ASTRAL their species tree used to study the evolutionary history across bacteria and archaea. We simulated a 10,000-leaves species tree using the Birth-death model and generated 500 gene trees with various degrees

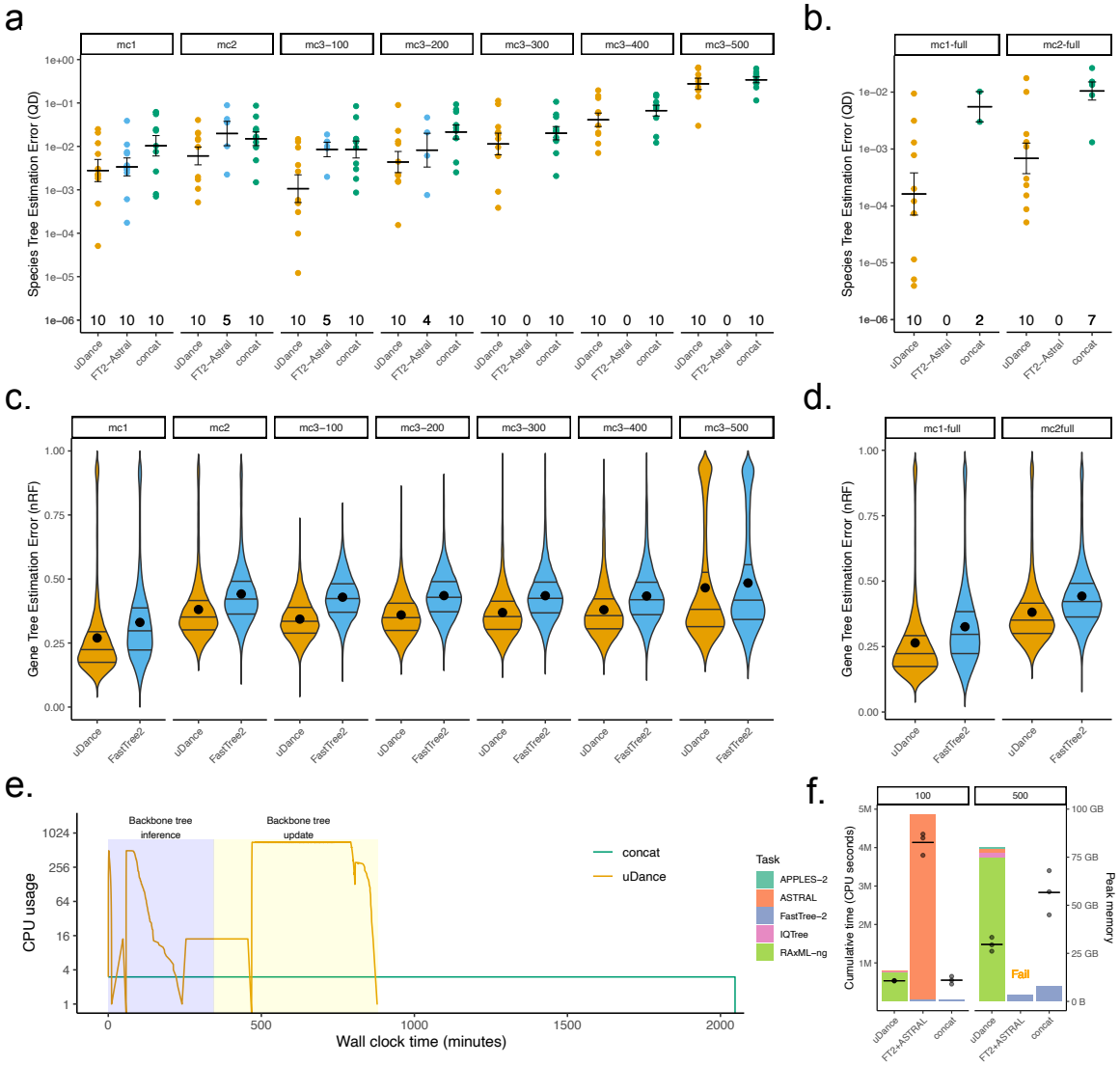


Figure 7.2. Results on simulation data set. (ab) Quartet distance between inferred and true species tree in 100 (a) and 500-gene (b) model conditions. The number above x-axis indicates the number replicates in which each method successfully returned a tree in 2 days given 125GB of memory. Whiskers indicate the first and third quartile range. (cd) RF distance between estimated and true gene trees for all partition-gene pairs in all model conditions. (e) The timeline of CPU usage on replicate 7 in mc2-full data set. Maximum number of cores made available for uDance is 672. (f) Cumulative CPU time (bars) and peak memory (dots and the horizontal bar) used by each method in the only three replicates where concatenation method completed on mc2-full (500 gene) data set and FT2+ASTRAL method completed on mc-2 (100 genes) data set. FT2+ASTRAL failed at ASTRAL step in mc2-full data set in all replicates. Note that we report total CPU time used by all cores across all nodes, and not the wall-clock time. Experiments are performed in machines with Intel(R) Xeon(R) 14-cores E5-2670 2.60GHz CPU and 125 GB RAM. For uDance, running time calculations include the time spent on backbone estimation.

of discordance with the species tree. The main contributor of the discordance in our simulations is horizontal gene transfer (HGT), whereas incomplete lineage sorting (ILS) contributed a small amount to the total discordance (Table 7.1). Next, we evolved DNA sequences on the simulated gene trees. We derived a total of nine model conditions by adjusting various model parameters (see Materials and Methods and Table 7.1). We performed full de-novo phylogenetic reconstruction using uDance where it first reconstructed a 1000-species backbone (that it chose) and then updated the backbone with the remaining 9000 species. We compared uDance to two alternative de-novo whole-genome inference approaches which can manage to run on our large-scale data set: (1) gene tree inference with FastTree-2 [188] followed by species tree inference using ASTRAL [258] (FT2+ASTRAL in short); (2) concatenation approach (concat) where we combine all input MSAs into a single MSA (supermatrix) and infer the species tree using FastTree-2. Note that more advanced ML methods such as RAxML-NG could not be run on the 10,000-taxon gene trees (of which we have 10,000 across this data set).

On simulated data, uDance can accurately infer the species tree in most conditions (Fig. 7.2a). As expected, increased gene tree discordance (mc1 to mc2 and mc3-100 to mc3-500) results in higher species tree error. However, uDance remains mostly accurate in five out of seven model conditions with only 100 gene trees, achieving mean quartet distances (QD) between true and estimated species trees that are less than two percent. On the mc3-400 condition with high (but below 50%) true discordance, uDance still has reasonable accuracy (0.044 QD) whereas in the mc-500 condition, where *estimated* gene trees are on average 71% different from the species tree, the error blows up to 34%. In all model conditions, uDance attains lower mean species tree estimation error than the other two methods, although in the easiest condition, mc1, FT2+ASTRAL comes close. uDance and concat successfully complete in all ten replicates in all seven model conditions with 100 genes but FT2+ASTRAL fails to return a tree within the limit of 48 hours computation time and 125 GB memory for all or some replicates in all model conditions except mc1. When measured using the normalized Robinson-Foulds (nRF) metric that is more sensitive to rogue taxa, uDance still remains to be the most accurate method in all

model conditions except mc3-500 where the HGT levels are the highest (Fig. 7.5). When we increase the number of gens to 500, uDance achieves average species tree estimation error less than 0.1% in both model conditions whereas the concatenation error is approximately an order of magnitude higher than uDance on the nine out of 20 replicates it completes. FT2+ASTRAL did not successfully analyze 500-gene datasets with the allotted time and memory due to the high computational demand of ASTRAL. The better accuracy of uDance is partially because its divide-and-conquer approach that allows computing ML gene trees using RAxML-NG have better accuracy (Fig. 7.2cd). On mc1 and mc2 data sets, uDance has 0.3 and 0.4 nRF distance to the true gene tree on average, and its gene trees are consistently more accurate than FastTree-2 ran on the full gene MSAs in all model conditions, both in 100 and 500 gene scenarios.

The distributed computing framework allows uDance to take advantage of HPC clusters and accelerate phylogenomics significantly. For example, uDance can infer a de-novo whole-genome phylogeny on one replicate of mc2-full dataset, which contains 10,000 sequences and 500 genes, in 14 wall-clock hours using 4 million CPU-seconds on a 48-node HPC with 672 cores in total (Fig. 7.2ef). Approximately 40% of the wall-clock time is spent on the inference of the backbone tree, which would be saved when a backbone tree is available from a previous round. uDance uses less than 1M CPU-seconds for de-novo inference on mc-2 dataset, whereas FT2+ASTRAL uses 5 times more CPU time than uDance despite being less accurate (Fig. 7.2f). For uDance, more than 90% of the CPU time is used for distributed RAxML-NG gene tree inference jobs, whereas for FT2+ASTRAL, the single-node multi-core ASTRAL-MP job consumes more than 99% of total compute. While concatenation uses the least CPU time, since FastTree-2 task is bound to a single-node and poorly parallelized, it takes more wall-clock time than uDance (Fig. 7.2e). Meanwhile, in terms of peak memory usage, uDance is the most scalable method among the three methods tested thanks to the divide-and-conquer strategy. On 100-gene dataset, uDance and concat uses approximately 10GB of memory whereas FT2+ASTRAL uses 80GB in average. However, on 500-gene dataset, uDance uses 30GB memory while concat uses twice as much.

Table 7.2. Exploring uDance parameter configuration.

Condition	Configuration	nRF	ΔnRF	QD	ΔQD
Default setting	Estimated,1000,500	0.061	0	0.012	0
Varying Backbone Size	Estimated,**250**,500	0.073	0.012	0.018	0.006
	Estimated,**500**,500	0.064	0.003	0.013	0.001
	Estimated,**2000**,500	0.059	-0.002	0.01	-0.002
Varying Partition Size	Estimated,1000,**100**	0.064	0.003	0.012	0
	Estimated,1000,**250**	0.062	0.001	0.012	0
	Estimated,1000,**1000**	0.061	0.0005	0.013	0.001
Varying Backbone Tree	**True**,1000,500	0.057	-0.004	0.0002	-0.012

These results makes the case for the utility of divide-and-conquer approach; by diving the full data set into manageable size partitions, it is possible to achieve better accuracy and scalability simultaneously. To assess the robustness of uDance to choices of the configuration parameters, we measures the impact of varying partition and backbone size parameters on the accuracy in mc2 model condition. We found that uDance is very robust to selection of partition size (Fig. 7.2). However, we saw that dramatically decreasing backbone size from (the default) 1000 down to 250 slightly increased QD between estimated and true species tree by 0.006. To further quantify the impact the backbone tree accuracy, we used the true species tree on the same set of backbone species and extended it with the remaining 9000 query species. Using the error-free backbone tree dramatically decreased the mean QD from 0.012 to 2×10^{-4}, signifying the importance of accurate backbone trees.

7.2.3 Phylogenomic Reconstruction of 200,000 Microbial Genomes.

We obtained 656,700 Bacterial and Archaeal genomes assemblies from NCBI. After several postprocessing steps such as duplicate and contamination removal (see Materials and Methods), we curated an approximately 280,792-strain data set with MSAs for the 381 genes used in the Web of Life (WoL) tree [263] and seven additional core ribosomal genes. Next, using

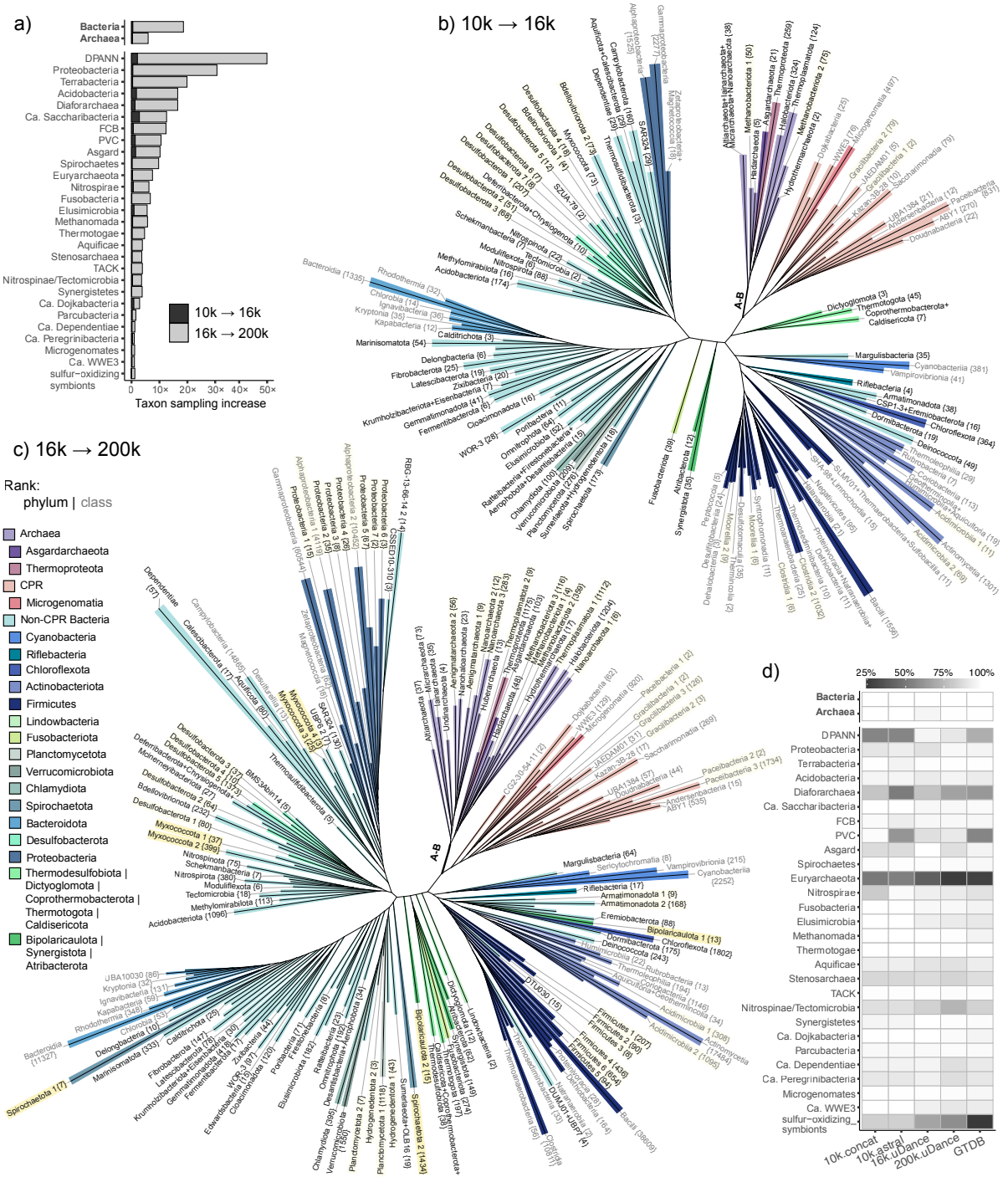


Figure 7.3. New trees of microbial life. (a) Taxon sampling increase in the 16k and 200k tree with respect to the 10k (Web of Life) tree. (b) 16k tree built by updating the 10k tree with uDance. (c) 200k tree built by updating the 16k tree with uDance. (d) Consistency of the large phyla and super-phyla in NCBI taxonomy database with various microbial phylogenies.

an automated procedure (see Supplementary Material), we selected 6,056 genomes from key groups with low taxon sampling in the WoL tree. We updated the WoL tree with the selected sequences using uDance and obtained a tree with 15,953 genomes (called the 16k hereafter). We ran a consecutive iteration of uDance with 280,792 remaining query genomes and reconstructed a tree with 199,330 genomes (called the 200k here after). The number of phyla, classes, and orders covered by the 200k tree are 140, 360, and 1060, when decorated with GTDB [181] taxonomy, which is 72%, 107%, and 172% higher than the WoL tree (also called the 10k tree) respectively. The number of Bacteria and Archaea taxa defined by NCBI increased 18 and 6-fold in the 200k in comparison to the 10k tree (Fig. 7.3a). In addition, the taxon sampling for the DPANN group, which was underrepresented in the sampling of the 10k tree, increased by 50-fold.

Both phylogenetic trees built by uDance demonstrated clear separation between Archaea and Bacteria domains (Fig. 7.3bc) and recapitulated some findings of Zhu et al. [263].First, Candidate Phyla Radiation (CPR, also Patescibacteria in GTDB taxonomy) forms a monophyletic group at the base of Bacteria. The length of the branch that separates Archaea and Bacteria (A-B branch) is still short (0.14, 0.27, and 0.18 in the 10k 16k, and 200k tree respectively, in substitution unit computed by ASTRAL).

We observed some topological changes close to the base of Bacteria as we grow the tree. Dictyglomota, Thermotogota, Caldisericota, Coprothermobacterota, and Synergistota groups with Fusobacteria in the 200k tree. Consequently, the clade defined by Cyanobacteria & Margulisbacteria, Chloroflexota, Actinobacterioda, Deinoccoccota, and Firmicutes is the most basal among non-CPR Bacteria. In the 200k tree, Bipolaricaulota is paraphyletic: 13 genomes from the phylum Bipolaricaulota, which is sister to Synergistota in the 10k (Fig. 7.6) tree, is located near Chloroflexota, whereas 15 genomes from Bipolaricaulota is located close to Synergistota. While Firmicutes is monophyletic in the 10k and 16k trees, it cannot be called monophyletic in the 200k tree unless the total of 19 genomes from phyla DTU030, DUMJ01, and UBP7 are re-assigned to Firmicutes. Like Firmicutes, reassignment of a total of 17 genomes from CSSED10-310 and RBG-13-66-14 could make Proteobacteria, the largest phyla in the

200k tree with more than 75,000 genomes, monophyletic. Desulfobacterota is monophyletic in the 10k tree and paraphyletic in the 16k (7 clades), 200k tree (5 clades), and the GTDB tree (7 clades). The topological changes between the backbone and the output trees occur at branches with various depth (Fig. 7.4c). The shallowest (the most basal) seven branches in the 16k tree can also be found in the backbone tree. However, nearly 10% of the conflicting branches have a depth between 0.35 and 1 while only slightly more than 1% of all branches in the 16k tree are in this group. Most conflicting branches between the 10k and 16k tree are short. 94% and 69% of disagreeing and agreeing branches are shorter than 0.09 respectively (Fig. 7.4d). Similar conclusions can be made when the agreeing and disagreeing branches between the 16k and 200k tree are compared (Fig. 7.7).

The uDance trees shows high consistency for several taxonomic super-phylum groups defined by NCBI taxonomy, including Terrabacteria, FCB, TACK, Microgenomates, and Parcubacteria (Fig. 7.3d). While the 16k tree shows high consistency for Archaeal groups DPANN and Asgard, their consistency is reduced in the 200k tree. Euryarchaeota group is strongly paraphyletic in the 10k (ASTRAL and concatenation), uDance, and GTDB trees. The consistency of super-phylum groups Euryarchaeota, sulfur-oxidizing symbionts, Nitrosplinae/Tectomicrobia, Nitrospirae, and Aquificate monotonically increases with every uDance iteration. On the other hand, PVC group becomes more consistent with every iteration. Therefore, the consistency of various groups in the microbial tree can change after increasing taxon sampling. Next, we investigated the topological similarity between the four phylogenies when they are induced to the members of the largest phyla and super-phyla defined by NCBI (Fig. 7.4e). While generally all phylogenies are largely compatible with each other, the 200k tree was highly similar to the 16k tree. For 24 of the 28 taxonomic groups, the quartet distance between the two tree was less than 2%. DPANN, Asgard, Saccharibacteria, and Synergistetes, are the top four groups in terms of the discordance between the 10k and 200k tree. Finally, we observed similar levels of the topological dissimilarity between the GTDB and the other three ASTRAL based trees (Fig. 7.8). This dichotomy be the consequence of several differences between two methodologies such as

species tree inference strategy (concatenation vs coalescent) or the selection marker gene set.

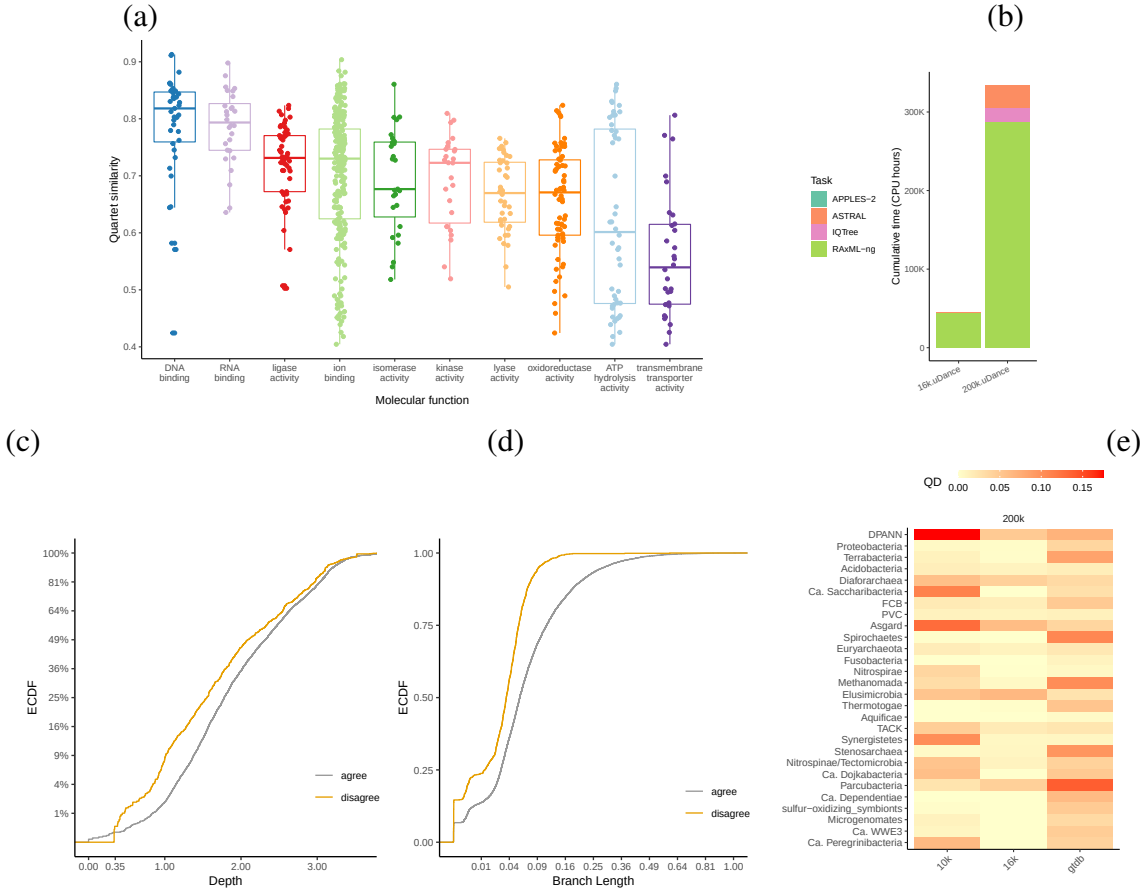


Figure 7.4. (a) Quartet score between the 16k species tree and gene trees categorized by the gene function. (b) Cumulative running time. ECDF of (c) depth (root-to-tip distance) (d) the branch length of agreeing and disagreeing branches between the backbone (the 10k) and the output (the 16k) phylogeny. (e) Quartet distance between the 200k tree and other trees on NCBI defined phyla and super-phyla.

HGT events are pervasive in microbial genomes; however, some genes may more susceptible to HGT than the others. For the 16k uDance tree, we measured the similarity between gene trees and the species tree for the ten largest functional categories of genes [7]. DNA and RNA binding genes, which are involved in the core machinery of genetic information processing, are the ones with the highest similarity to the inferred species tree in average, with 82 and 80 percent quartet similarity (one minus quartet distance) (Fig. 7.4a). On the other hand, phylogenies of genes involved in peripheral functions such as transmembrane transport are, in average, less

topologically to the uDance tree. These findings are in line with the WoL study [263].

7.3 Discussions

We presented uDance: a divide-and-conquer workflow for incrementally growing and updating ultra-large whole-genome phylogenies. We showed that this workflow is more accurate than de-novo species tree reconstruction via concatenation and coalescent-based phylogenomic approach using ASTRAL. We also demonstrated that uDance uses less memory than both alternatives and uses less compute than de-novo ASTRAL-based inference. Since it is massively distributed, uDance enables faster inference than concatenation method. uDance is applicable in real biological data sets: We built a 200,000-genome microbial tree-of-life based on 388 marker genes.

Despite the its scalability and accuracy, the divide and conquer strategy we introduced in this paper has several limitations. We recommend the user to limit the partition to 6,000 as ASTRAL becomes too slow for larger trees. As a consequence, uDance requires the number of partitions to be large when the backbone tree is large. Since uDance fixed the relative positions (the hierarchy) of partitions, the scope of topological updates can be restrained to the edges that are farther from the root. We leave it to future work to reevaluate the the hierarchy of the partitions after partition updates.

Inserting a small set of genomes on a ultra-large backbone tree can be inefficient if APPLES-2 places the genomes uniformly in all partitions of the backbone tree, requiring recomputation in each partition. There is two solutions to this problem. First solution does not require any algorithmic novelty; the user can stall updates on each partition until the number of new genomes reach a significant amount. In uDance, gene tree search is unconstrained. The second solution is to constrain RAxML-NG search with the gene trees from the previous iteration, which can accelerate the gene tree inference in the expense of decrease to find a large likelihood tree. This feature can be added to uDance and is left to future work.

7.4 Materials and Methods

7.4.1 Workflow of uDance

Each step of the pipeline (Fig. 7.1) is described below in detail. Note that most steps include quality control and filtering steps. To create a better flow, we have collected most of these steps under Quality control.

Placement

When a backbone tree is available, the sequences in the input MSAs that are not present in the set of backbone sequences are assumed to be query sequences. uDance concatenates all input MSAs, removes duplicate query sequences, and places them on the backbone tree using the scalable phylogenetic placement tool APPLES-2 [15]. APPLES-2 uses Jukes-Cantor model [105] for DNA and Scoredist [218] for amino acid (AA) characters in the input sequences. Since APPLES-2 places each query independently, the relationship between the queries is not resolved in the resulting placement tree.

Decomposition

uDance refines the placement tree using a divide-and-conquer approach. In this approach, the placement tree is divided into several independent partitions which are processed independently and in parallel. Our decomposition algorithm is parameterized by a (soft) minimum threshold α of diversity (measured in phylogenetic total branch length in substitution units) and a (soft) limit S on the size of each cluster. All other constant values mentioned below are adjustable by the user.

We denote the placement tree with a rooted binary tree $T = (V, E)$ represented by an undirected acyclic graph with vertices V (each with degree one or three, except the root), edges E with a length and weight, and leaf set $\mathscr{L} \subset V$. We denote the path length between nodes u and v on T with $d(u, v)$. We define $p(u)$ to be the parent node of u. The branch length of an edge $e = (u, v)$ is denoted by b_e. We define $P(e)$ to be he set of query sequences placed on an edge.

The weight w_e of the edge e is $|P(e)|$ if e is an internal node and $|P(e)|+1$ if the edge is terminal (leaf). A clustering of the edges of the tree T can be defined by coloring edges with the condition that the set of edges with the same color are always a connected (unrooted) subtree.

We use $C(e)$ to denote the color of and edge e. For the sake brevity, we write $C(u)$ as a shorthand for $C((u,p(u)))$. For a color set $C = \{C(e)|e \in E\}$ of the tree, we define $B(u)$ to be the length of the path from u to the most distant *connected* descendant node in v such that $C(v) = C(u)$. We use $Q(u)$ to denote the total number of query placements and leaves that are descendant of u and share the same coloring. The algorithm (Algorithm 3) uses a bottom-up traversal of the tree and for each node u that we visit, we may decide whether the node is a junction of neighboring colors. The node is a junction if $Q(l)+w_l+Q(r)+w_r+w_u$ exceeds the desired partition size limit S but we make two exceptions to this rule: (1) either b_u, b_l, or b_r is shorter than 10^{-5}; or (2) declaring the node as a junction creates a partition with diameter (maximum pairwise distance between any nodes in the partition) less than the threshold α. Each set of edges with the same color defines a partition that consists of leaves incident to the member edges and all placements on them.

Due to the exceptions described above, a partition may have more sequences than the soft limit S. If the number of all sequences in the query exceeds terminal partition size τ (6000 by default), we remove near duplicate query sequences using a iterative graph thresholding algorithm. We create an $m \times m$ adjacency matrix G for such large partitions where m is the number of query sequences in the partition. We define the weight of edge $G(x,y)$ in this graph to be the fraction of genes in the input dataset for which the gene sequences of x and y are identical. Starting from $\beta = 0.99$, we create a second matrix G' such that $G'(x,y) = 1$ if $G(x,y) \geq \beta$, otherwise zero. Then, we compute the number of connected components in G'. We gradually decrease β until the number of connected components in G' is less than τ. Finally, we choose the sequence with the highest gene occupancy from each connected component and filter out other query sequences from the workflow. We report a mapping from each selected sequence to the removed members of its connected component. This mapping can be used to add the removed

duplicates back as a polytomy at the end.

Selecting outgroups

Let $\mathscr{L}(i)$ and $\mathscr{E}(i)$ denote the set of backbone sequences (leaves) and edges in the partition defined by color i. Partitions computed by Algorithm 3 consist of disjoint edge and leaf sets. However, two or three partitions may be adjacent to each other at a "junction" node. We define a scheme to make partitions overlap by adding to each partition some leaves from other partitions, which we call outgroup sequences. More formally, let v be a node and let three edges adjacent to v be denoted with (u_i, v), $i \in \{1,2,3\}$ such that $(u_i, v) \in \mathscr{E}(c_i)$. v is a junction node if $c_1 \neq c_2$ or $c_1 \neq c_3$ or both. In this case, we first reroot T at node u_1. We first consider case where $c_1 \neq c_2$ is true. We denote left and right child of u_2 with l and r respectively. For $j \in \{l, r\}$, we select one leaf x that is a descendant of j and minimizes $d(x, j)$ (Fig. 7.13). Note that $C(x)$ may be different than c_2 when there is another partition below u_2. We additionally select the descendant leaf of u_2 with maximum gene occupancy if it is not identical to one of the two selected leafs earlier. We expand the list of selected outgroup sequences with the closest and the most occupant descendants of u_3 if $c_1 \neq c_3$. This selection strategy adds two to six outgroup sequences to c_1 due to the junction node v. We repeat this procedure for all colors adjacent to v. We traverse internal nodes of T and apply this procedure to all junction nodes.

Let $O(i)$ be the set of outgroup sequences designated to be added to color i. We call the tree T induced to $\mathscr{L}_i \cup O_i$ *incremental-cons* tree for partition i. Furthermore, we call the tree T induced to O_i the *updates-cons* tree for partition i. These trees are used to contstrain ASTRAL tree search. After refinement of backbone and query sequences in each partition, the resulting trees can be stitched together by recreating each junction node in the coloring of T.

Partition tree refinement

uDance infers a species tree on the set of all backbone and query sequences for each partition using ASTRAL-constrained. First, uDance performs unconstrained ML gene tree

Algorithm 3: Placement-weighted clustering

Input: A tree $T^o = (V, E)$, a threshold α,
and partition size limit S

current_color $\leftarrow 0$
$C(e) \leftarrow -1$ **for** $e \in E$
$B(v) \leftarrow 0$ **for** $v \in V$
$Q(v) \leftarrow 0$ **for** $v \in V$
for $u \in$ *post order traversal of internal nodes of* T^o **do**
    l,$r \leftarrow$ left and right child of u
    if $Q(l) + w_l + Q(r) + w_r + w_u \leq S$ ***or***
    $B(l) + b_l + B(r) + b_r < \alpha$ ***or***
    $b_l \leq 10^{-5}$ ***and*** $w_l > 0$ ***or***
    $b_r \leq 10^{-5}$ ***and*** $w_r > 0$ ***or***
    $b_u \leq 10^{-5}$ ***and*** $w_u > 0$ **then**
        $Q(u) \leftarrow Q(l) + w_l + Q(r) + w_r$
        $B(u) \leftarrow max(B(l) + b_l, B(r) + b_r)$
    else if $Q(l) + w_l + Q(r) + w_r \leq S$ **then**
        paint(l,current_color); paint(r,current_color)
        current_color += 1; $B(u) \leftarrow 0$
    else if $min(Q(l) + w_l, Q(r) + w_r) + w_u > S$ **then**
        paint(l,current_color); paint(r,current_color+1)
        current_color += 2; $B(u) \leftarrow 0$
    else if $Q(l) + w_l \geq Q(r) + w_r$ **then**
        paint(l,current_color); current_color += 1
        $B(u) \leftarrow B(r) + b_r$
    else
        paint(r,current_color); current_color += 1
        $B(u) \leftarrow B(l) + b_l$
return *Clusters of edges defined by their assigned color*
Function `paint(`*u,color*`)`**:**
    for $v \in$ *descendants of* u **do**
        if $u \neq v$ ***and*** $C(v) = -1$ **then**
            $C(v) \leftarrow color$
    return

inference with the user-specified tool (RAxML-NG by default, RAxML-8, or IQTree-2) and phylogenetic model. User also has the option to specify number of starting trees and make model selection automatic using ModelTest-NG [45]. Regardless of the inference tool used, uDance computes the branch supports with IQTree's ultrafast bootstrap approximation (aBayes) [6].

The final stage in the phylogenomic inference workflow is constrained species tree inference using ASTRAL-constrained. The inference is performed twice using two separate constraint trees, *incremental-cons* and *updates-cons*. The incremental-cons includes the set of the backbone and out-group species in the partition and enforces the outputted ASTRAL phylogeny to fully match the input phylogeny when queries are ignored. The update-cons only includes the outgroup species, allowing both query and backbone species to freely move during the search. uDance has three modes for refinement tree selection. The first option selects ASTRAL runs with incremental-cons for all partitions, resulting in a uDance tree that fully matches the backbone tree topology, again, when inserted query sequences are ignored. The second option selects ASTRAL runs with updates-cons for all partitions. The third option is *maximum-qs* approach: uDance picks the ASTRAL tree with higher quartet score among the two ASTRAL runs for each partition. The maximum-qs is the default method we will use throughout. We estimate branch lengths of the species tree in substitution unit using a new algorithm that we developed and made available as part of ASTRAL (v5.17.2).

Stitching

We use $A(c)$ to denote the output ASTRAL tree for partition c. We start the stitching procedure with a bottom up traversal of the internal nodes of the backbone tree. When we come across a junction node v, for every pair of edges $i, j \in \{(v,u_1),(v,u_2),(v,u_3)\}$ such that $C(i) \neq C(j)$, we perform the following operation. Let c_1 and c_2 denote the colors $C(i)$ and $C(j)$. In $A(c_1)$, we first find a leaf $x \in O(c_2)$. That leaf exists since there are outgroups in partition c_2 corresponding to the adjacency of c_1 and c_2. We root $A(c_1)$ at x and find the MRCA of the outgroups representing c_2. We iterate the ancestor nodes of the MRCA until we find a

node y whose sibling tree contains a leaf from $\mathscr{L}(c_1) \cup \mathscr{O}(c_1)$ (a non-query sequence). The node y corresponds to the junction node v and therefore all descendants of y are removed from $A(c_1)$, including any backbone and query sequences since they are located outside the partition boundary and group with outgroup sequences. We repeat this procedure for c_2 as well and stitch two partitions at the node y found in both partitions. Finally, we repeat the stitching process for all junction nodes v until all partitions are stitched.

Quality control

Sequence quality.

Prior to phylogenetic placement stage, uDance filters out extremely gappy sites (sites with at least 95% gaps by default) in the input MSAs using TrimAl [34] on order to save time and memory. Furthermore, uDance use TAPER [259] to mask small errors that occur in the MSAs. At gene tree inference stage, uDance computes an induced MSA and remove all-gappy sites or each partition gene pair. If this MSA has fewer than 100 sites, the gene is discarded in that partition due to strong evidence that gene tree estimation from short fragmentary sequences is problematic [210]. In addition, any sequences (fragments) with fewer than 75 base-pairs in each partition for each gene are filtered out from the induced MSA. To detect further errors in the MSA, uDance infers a quick ML gene tree for each gene using FastTree-2 and perform outlier sequence detection and removal with TreeShrink [145] using its default parameters.

Backbone quality.

Errors in the input backbone topology can negatively affect the phylogenetic placement step. uDance performs two kinds of backbone filtering before the placement stage. First, a misplaced sequence in the backbone can attract queries of its kind to the wrong location (and thus wrong partition). uDance uses APPLES-2 to remove and phylogenetically place back every sequence in the backbone and measure the topological distance between the original and new location. uDance removes the sequences when the difference is larger than $\log_2 n - 1$ from the backbone where n is the number of backbone sequences (call these suspect sequences). We chose

this equation because in a balanced tree, this equation gives the height of the tree in number of branches and a decent choice for a cutoff for species that relocated to a distant location in the tree. uDance then re-estimates branch lengths of the backbone tree with suspect sequences pruned; next, uDance inserts back suspect sequences using APPLES-2 and keeps those that no longer place no more than $\log_2 n - 1$ edges farther than the original position. Note that this can happen because after removing all suspect sequences, backbone tree branch lengths can change and such changes can be consequential. Second, uDance uses TreeCluster [12] to cluster the backbone tree with *max* clustering strategy and threshold 0.7. In each cluster, one-dimensional k-means clustering detects the outlier sequences in terms of gene occupancy and removes them from the backbone. The rationale is that sequence with low occupancy with respect to the nearby sequences is suspected to be misplaced.

Gene tree quality.

At species tree inference stage, uDance computes the median branch support for all the input gene trees and clusters them based on the median branch support using one dimensional k-means clustering with $k = 2$. If the cluster with the smaller centroid has less than 20% of all the genes present in the partition and the difference between centroids of the two clusters is larger than 0.1, all genes in the cluster with the smaller centroid is discarded. uDance proceeds with contracting low support branches using the user-specified threshold. We use 0.33 and 0.66 for simulated and biological dataset respectively (note that aBayes branch supports range between 0.33 and 1).

Backbone inference

In addition to the ability to operate with a user-provided backbone tree, uDance can also infer a de-novo backbone when none is available. uDance uses a new procedure named Mainlines to select a subset of n sequences in the backbone tree with high diversity. Mainlines begins with creating a concatenation MSA by selecting $\lceil \frac{l}{k} \rceil$ least gappy sites from each k gene

MSA. Here, l is the number of sites in the concatenation alignment (5000 by default) and can be specified by the user. Next, FastTree-2 infers a phylogenetic tree using GTR+G or LG+G model depending on whether input sequences are nucleotide or AA. Mainlines uses this tree solely for subsampling n backbone species from the entire data set. The selection is carried out by exploring the range of plausible threshold values for TreeCluster (max option) via binary search. The search stop when either a threshold that results in exactly n clusters is found or the threshold difference between two consecutive iterations is below 0.0001. After Mainlines determines the set of backbone sequences, uDance executes the phylogenomic inference pipeline, which infers ML gene trees (default using RAxML-NG) and an ASTRAL species tree. The only difference between the pipeline used during the backbone inference stage is that the ASTRAL search has no tree constraints.

Distributed Implementation

We implement uDance using the scalable bioinformatics workflow engine Snakemake [114]. uDance is flexible and largely configurable. uDance is supported in multiple operating systems (Linux, macOS), easy to install, and thanks to Snakemake, easily deployable in HPC platforms in a distributed fashion. uDance is publicly available on https://github.com/balabanmetin/uDance.

7.4.2 Simulations

We perform a set of simulations using SimPhy [149], starting from a default model condition (named mc-2-full) and deriving other eight model conditions by adjusting simulation parameters. We simulate 10 replicates per model condition. Simulation parameters are chosen with reference to Web of Life microbial data set.

Default model condition.

We simulate a 10,000 taxa species tree under Birth-death process with fixed speciation and extinction rate of 5×10^{-7} and 4.16×10^{-7} respectively. The number of generations in

the tree is fixed to 10^9. Each replicate has 500 gene trees and ILS and HGT constitutes the two sources of gene tree discordance. Gene trees have 0.03 mean nRF distance to the species tree due to ILS prior to introduction of HGT events. We set the rate of HGT events so that the average discordance across all replicates is 0.38. We note that the amount of discordance due to HGT is much higher than ILS because we wanted to remain similar to our microbial reference dataset, WoL. The probability of a horizontal gene transfer event between two taxa is inversely proportional to their distance in the species tree. For each gene, we use INDELIBLE [69] to simulate multiple sequence alignments under GTR+G model. In each replicate, we draw two gene sequence length hyperparameters λ and ν from Uniform(5.5, 6.5) and Uniform(0.1, 0.2) respectively. The length of each gene sequence within a replicate is drawn from Lognormal(λ, ν). This parametric process results in lengths ranging between 169 and 869, and averages 406 base pairs.

We randomly delete a single consequent chunk of characters in each sequence through the following process in which the deleted range is more probable to be on tips of the sequence than the center. The ratio of the deletion length to the total length is drawn from Beta(ρ, 1-ρ) where ρ is a hyperparameter drawn from Uniform(0.2,0.6). The location of the center of the deleted range is drawn from Beta(0.4,0.4) distribution where the 0 and 1 represents the leftmost and rightmost eligible center location respectively. Finally, we realign modified sequences using UPP [176].

Gene tree estimation error, measured by nRF between true gene and tree estimated with FastTree-2 under GTR+G model, depends on SimPhy sequence mutation rate besides the gene sequence length and alignment gappiness parameters. We adjust the mutation rate so that the average error is approximately 0.45. In particular, the overall mutation rate is 4×10^{-8} per generation and there are rate multipliers per gene, per species, and per gene/species, which ensure deviations from ultrametricity. With average error set to 0.45 and true discordance around 0.38, the discordance from the estimated gene trees to the true species is around 0.58, which is similar to the levels of discordance observed on the WoL dataset.

Derived model conditions:

We create another model condition named mc-1-full from the previous one by (1) making two adjustments: (1) changing the distribution from which λ is drawn to Uniform(6, 7) and (2) decreasing the HGT rate so that gene tree discordance is 0.35. We generate mc-1 and mc-2 by selecting only the first 100 out of the 500 genes from mc-1-full and mc-2-full respectively. Finally, we derive five additional model conditions mc-3-100, mc-3-200, mc-3-300, mc-3-400, and mc-3-500 by selecting the gene trees in ranges 1-100, 101-200, 201-300, 301-400, and 401-500 after all genes in mc-2-full dataset are sorted low-to-high according to the nRF between their estimated gene tree and true species tree.

7.4.3 Biological Data

Web of Life 2

Data preparation.

A total of 656,574 reference bacterial and archaeal genomes were retrieved from the NCBI genome database on May 14, 2020. This included genomes from both RefSeq and GenBank [81], with the former prioritized if the same genome is found in both sources. Two sets of marker genes were independently inferred from the genomes. First, 400 global marker genes were inferred using PhyloPhlAn [215] v1 (commit 2c0e61a), with default parameters, on the amino acid sequences of the open reading frames (ORFs) predicted by Prodigal [99] v2.6.3, with default parameters. Of which, the 381 genes previously validated [99] were selected. Second, 37 core marker genes were inferred PhyloSift [44] v1.0.1 from the genome sequences, with default parameters. For the 11 of 37 these core marker genes that were present in both the core and global marker set, we selected the ones in the global set. Amino acid and nucleotide sequences of the marker genes were extracted using previously developed scripts. We removed any genome with fewer than 66 marker genes. We retained only seven of 26 new core genes as we removed any gene with fewer than median sequence length 150 AAs among the curated genomes. The new curated set is a superset of all sequences in the WoL reference data set. For each 381 marker

gene, we aligned the query AA sequences onto the corresponding MSA from the WoL data set using UPP [176], masking alignment insertion sites. Since the WoL dataset did not include the 7 core ribosomal genes, we computed the backbone MSAs for those genes ourselves using UPP. We excluded marker gene p0127 from the analysis as UPP failed to finish in 48 hours. 1412 genomes with GUNC [180] clade seperation score ≥ 0.50 and contamination portion ≥ 0.25 are suspected to be contaminated or chimeric and removed from the data set (Fig. 7.14). After removing duplicate sequences that share identical AA sequences, the number of unique genomes in the data set reduced to 296,745.

Reconstruction of the 16k tree.

Using an automated procedure (see Supplementary Material), we selected 6,056 species for insertion on the WoL phylogeny. These sequences were chosen in a way that sought to increase taxon sampling of key groups with low sampling (see Fig. 7.3a). We performed two rounds of uDance (v1.1.0) to update the WoL tree with the selected sequences. We instructed uDance to use to the entire global marker gene set at phylogenetic placement stage. We ran uDance with partition size parameter 1,000, which resulted in 20 partitions. For gene tree estimation inside uDance, we opted to use RAxML-NG [115] with LG+G [122] model and three starting trees. Finally, in this uDance run, we set the low support branch contraction in threshold to 0.9 and minimum gene occupancy threshold to 30. 241 genomes were unplaceable in the output tree after the first run. The number of unplaceable sequences was less than 25 in all partitions except one partition where 187 genomes, mostly classified as members of Myxoccocota and Bdellovibrionota phyla, were dropped out. We ran a second round of uDance with partition size parameter 2,000 and re-inserted 172 of these 187 unplaceable sequences in the second round. We refer to the resulting tree tree with 15,956 genomes as the 16k tree.

Reconstruction of the 200k tree.

We performed one round of uDance (v1.6.0) where we updated the 16k tree with the remaining 280,792 query sequences in the data set. We did not add the queries that were

unplaceable in the 16k run to the query set in this run. In order to speed up phylogenetic placement, we sorted the marker genes based on the quartet distance between their gene tree and the species tree in the WoL dataset and selected the top 68 marker genes. In an earlier study [15], we found that phylogenetic placement on the WoL dataset using 50 marker genes is nearly as accurate as using the full set. However, the cutoff number 68 was determined based on a trade off between the number of Archaea-rich genes and the total number of genes in the selection (Figs. 7.11 and 7.12). Unlike the 16k run, we used the 68 selected marker genes during the phylogenetic placement stage in uDance. We set partition size to 2,500, which created 78 partitions. After near-duplicate removal in some large subsets, the total number of query and backbone sequences in all partitions combined equaled to 201,316; thus, roughly 80,000 genomes that were nearly (but not fully) identical to many other genomes were dropped. We opted to use RAxML-NG with LG+G model and two starting trees inside uDance. Finally, in this uDance run, we set the low support branch contraction in threshold to 0.66 and minimum gene occupancy threshold to 30. The resulting tree, which we call the 200k tree, contains 199,330 sequences. 1,986 sequences were either unplaceable or were removed in one of the quality control steps in the workflow.

Taxonomy decoration.

In this analysis, we used NCBI (retrieved on 2020-07-01) and GTDB (release 207) taxonomy databases. Taxonomy decoration and consistency analysis is performed using tax2tree [155]. To compute consistency of a tree with NCBI database, we first decorate the tree using NCBI taxonomy and then the decorated taxonomy with the source NCBI taxonomy. When decorating using NCBI database, we only performed assignment at phylum and super-phylum rank (also called clade. Examples are PVC, Parcubacteria, and DPANN). We removed any suffixes(such as _A, _B) of the names of paraphyletic ranks before the decoration with GTDB database.

Visualization.

The trees in this study are visualized with iTOLv5 [129]. Unique colors were assigned to selected phyla and classes and according to the taxonomic decoration using GTDB database. To display the 16k trees in a page, we collapsed the clades that represent a single phyla and with fewer than 400 leaves or that represent a single class. We matched the phyla and class level visualisation in the 200k tree to 16k tree. After collapsing, we grouped a clade of phyla or classes if each one had fewer than 20 and 40 members for the 16k and 200k tree respectively. We added numerical suffixes to the names of the paraphyletic ranks. We dropped names for remaining clades that were assigned alpha-numeric temporary names (e.g., UBA3054) in GTDB.

7.4.4 Methods Compared

We compare uDance with the following methods on the simulated HGT dataset.

The first method is the coalescent based two-step phylogenomic approach where the first step is gene tree inference and the second step is species tree inference using ASTRAL-MP. We perform gene tree inference with FastTree-2 using GTR+G model of evolution. Systematic exploration of phylogenomic approach using alternative gene tree inference tools such as IQTree-2 and RAxML-NG is not feasible on the HGT data set due to computational cost of these tools.

The second approach is the concatenation-based species tree inference where we create a concatenation MSA (also called supermatrix) and perform phylogenetic inference with FastTree-2 using GTR+G model. Once again, FastTree-2 is the only tool that can handle the large inputs in our dataset.

7.4.5 Acknowledgements

Chapter 7, in full, is currently being prepared for submission for publication of the material. “Balaban, M., Jiang Y., Zhu Q., McDonald D., Knight R., Mirarab S. Growing phylogenomic trees at the ultra-large scale using divide and conquer”. The dissertation author

was the primary investigator and first author of this paper.

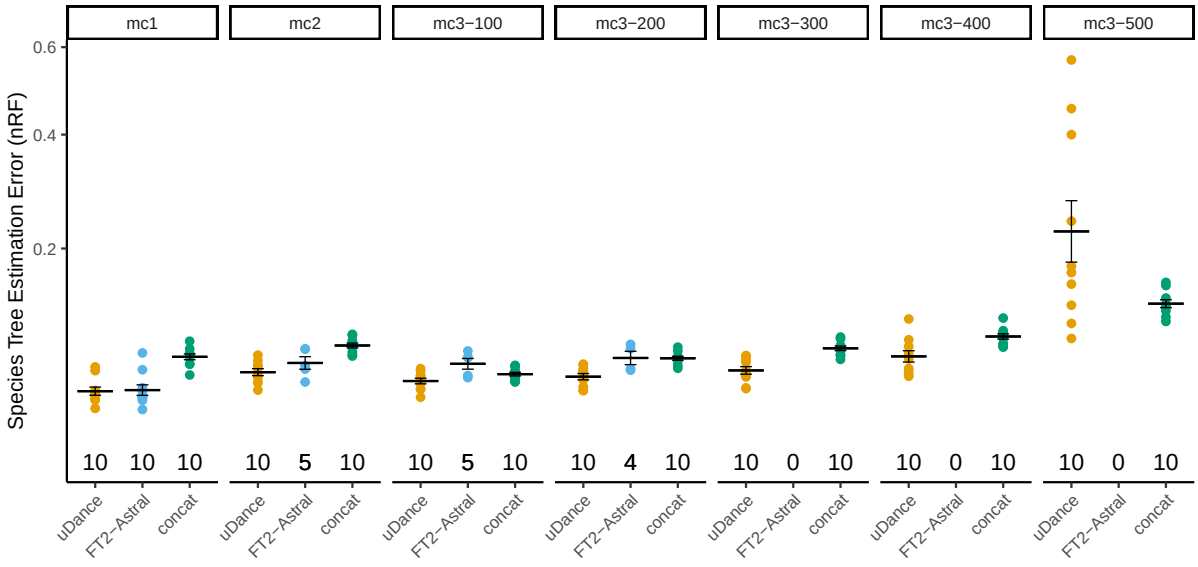


Figure 7.5. Species tree estimation error measured using nRF on simulated data with 100 genes.

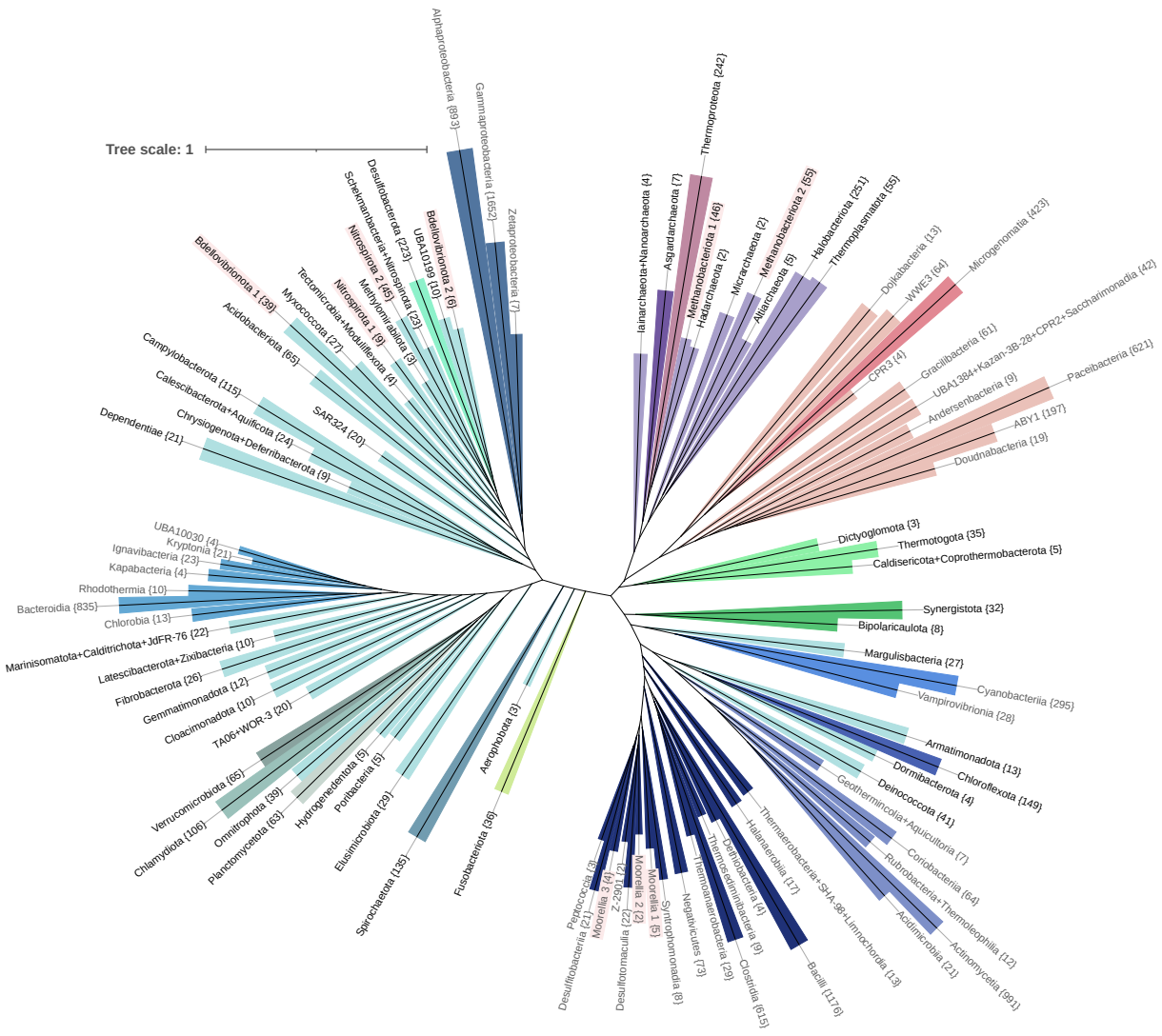


Figure 7.6. The 10K ASTRAL tree decorated with GTDB taxonomy.

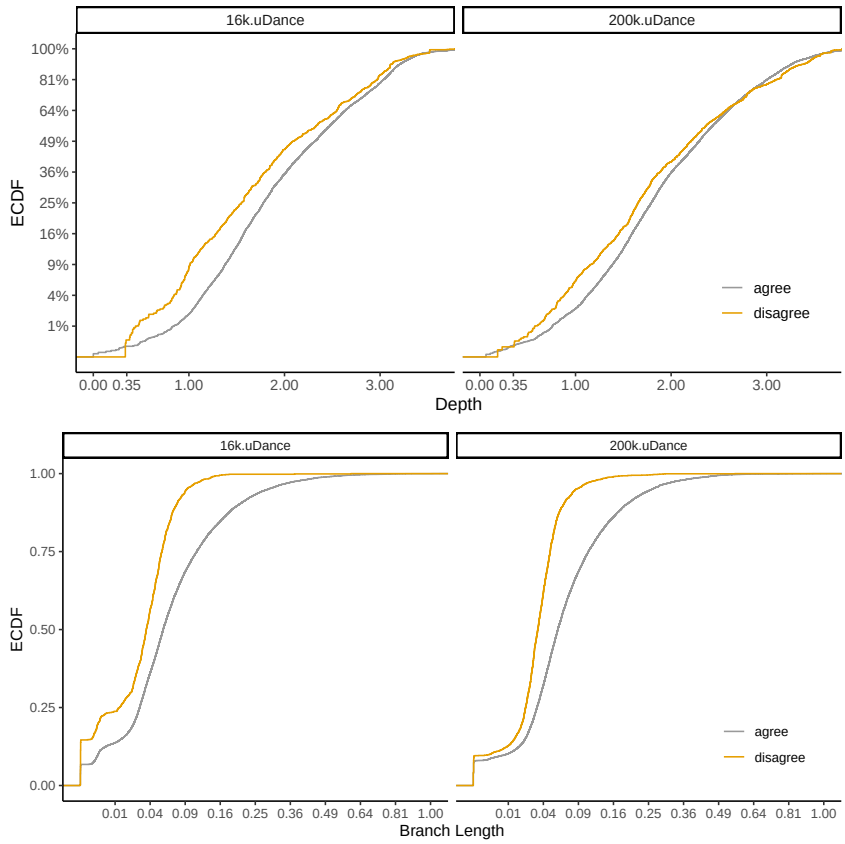


Figure 7.7. ECDF of depth and the branch length of agreeing and disagreeing branches between the backbone and output phylogenies.

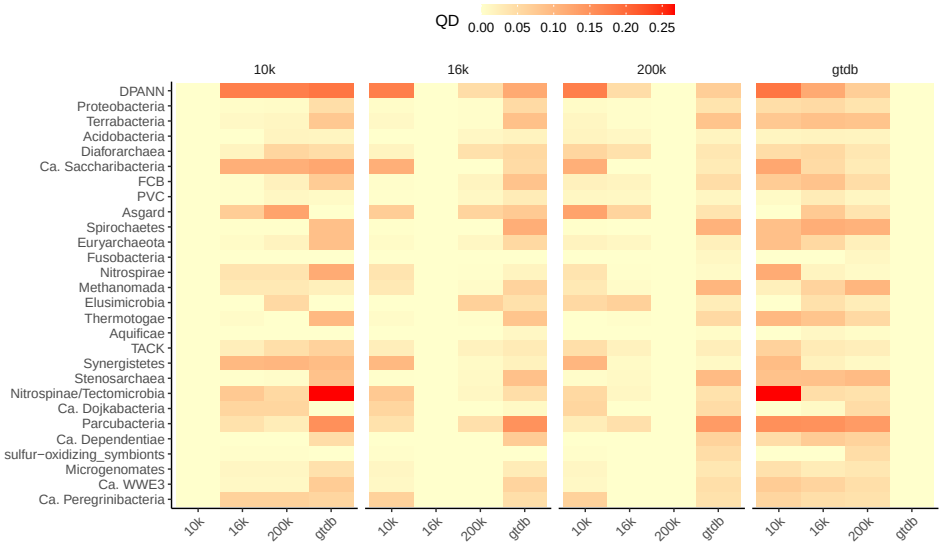


Figure 7.8. All by all comparison between the 10k, 16k, 200k, and GTDB trees on NCBI defined phyla and super-phyla.

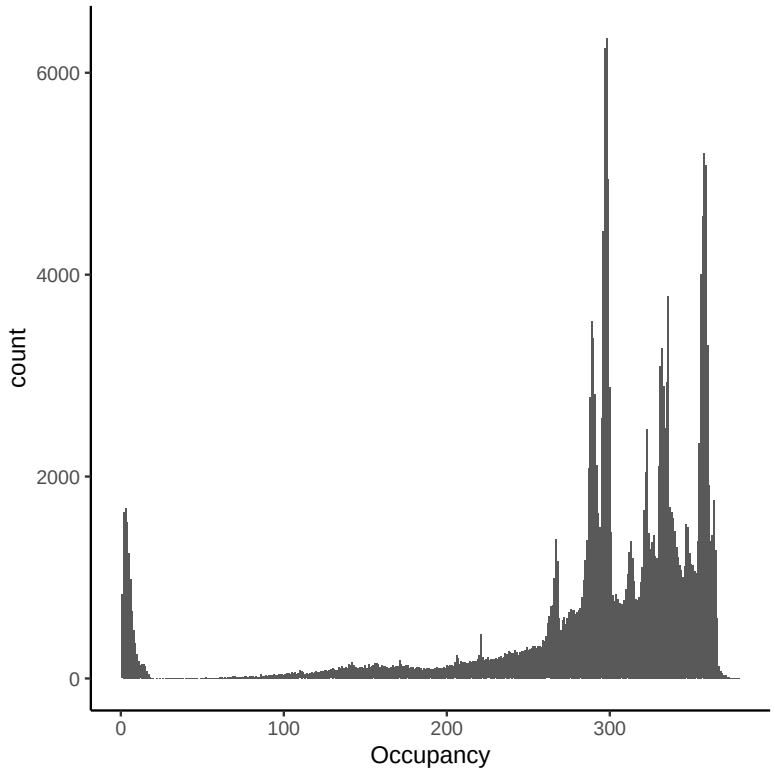


Figure 7.9. The distribution of number of marker genes per sequence in WoL2 dataset.

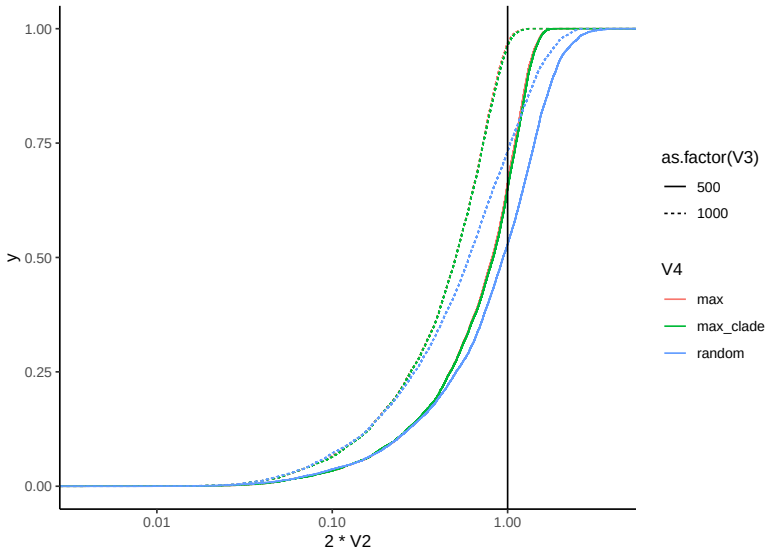


Figure 7.10. Mainlines backbone selection strategy. ECDF of novelty of query sequences with respect to a backbone tree with N sequences induced from the full true species tree of HGT dataset. With $N = 1000$ sequences selected using TreeCluster, for more than 95% of the query sequences, the novelty score is less than one. Novelty score is defined as two times the terminal branch length of the query when placed on the true location on the backbone tree.

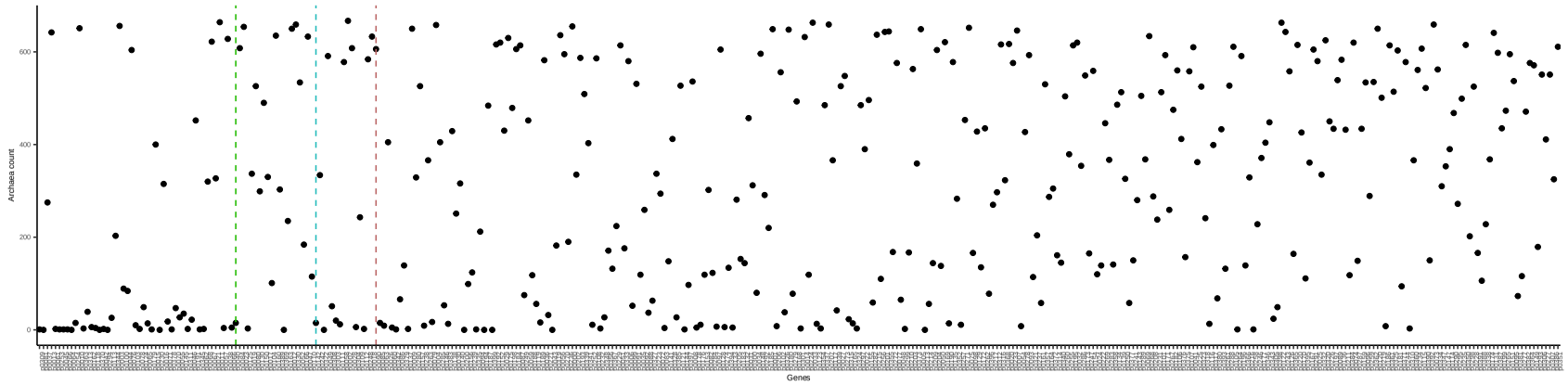


Figure 7.11. X-axis is the 381 marker genes used in WoL study sorted by quartet distance to its gene tree to the WoL species tree. in increasing order. Number of Archaea sequences in WoL tree that include indicated marker the gene. Vertical lines are drawn at 50th, 70th and 100th marker gene.

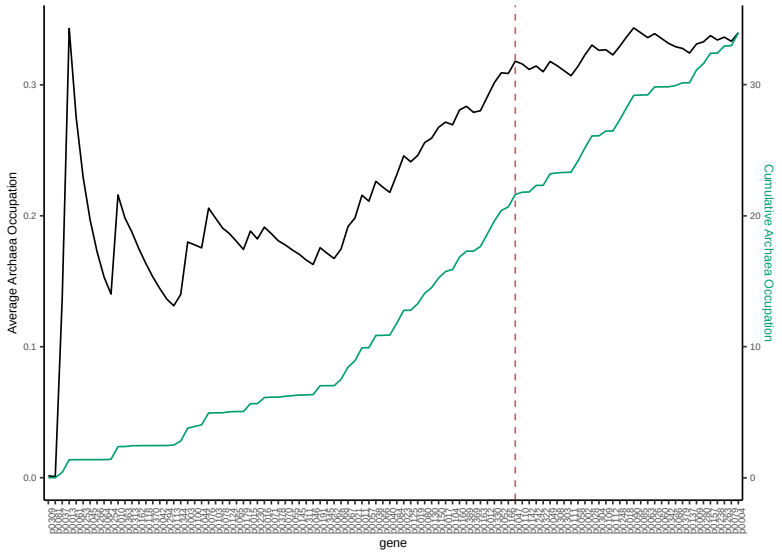


Figure 7.12. How we determine APPLES-2 marker gene set. We adopt "best" marker gene strategy described in APPLES-2. In order to improve placement speed, we only use a subset of marker genes. We picked a local maxima of average Archaea occupancy at 68th marker gene, which also ensures that in average Archaea sequences have at least 20 marker genes. The set of Archaea used in computation of these two statistics are taken from WoL tree.

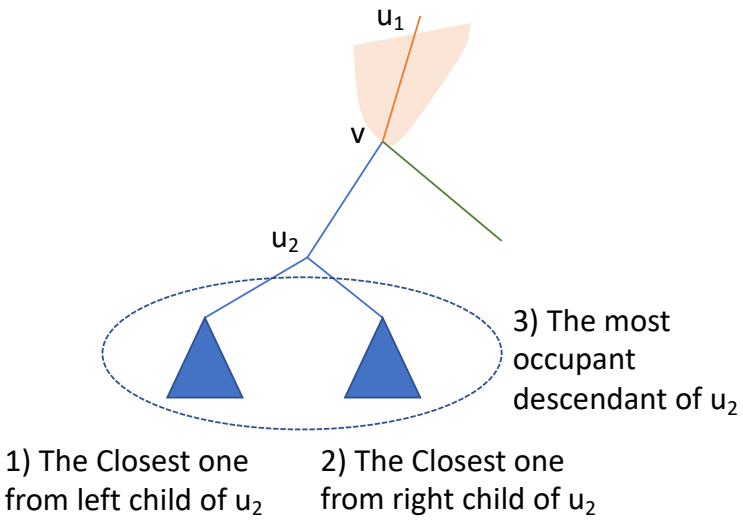


Figure 7.13. Outgroup taxa selection strategy. Two to three taxa are chosen from the partition c_2 (blue) to be added to the partition c_1 (orange).

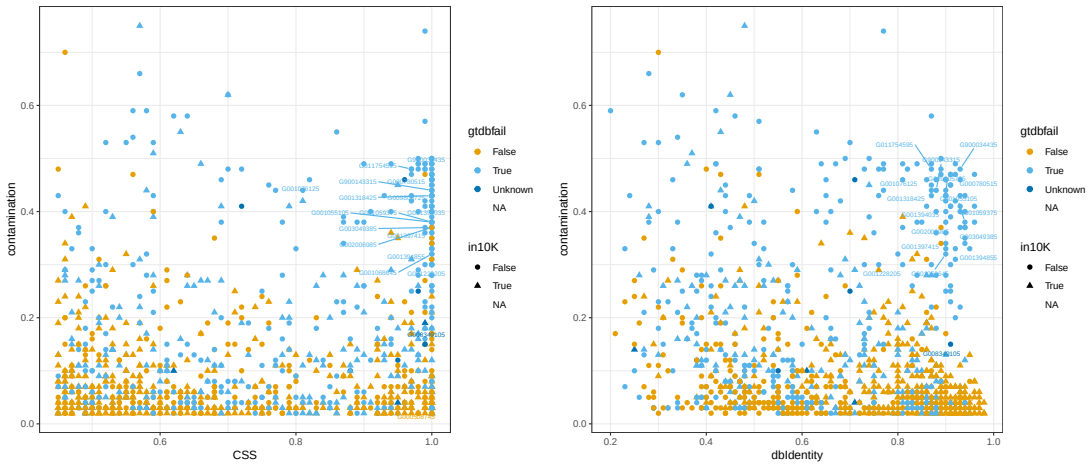


Figure 7.14. Two dot plots comparing (1) contamination ratio-vs-CSS and (2) contamination ratio-vs-GUNC database identity for the species in the 16K tree that are ”chimeric” (CSS ¿0.45). We colored each point based on whether the sequence passed QC in GTDB or not. Triangle points are the published WoL tree, and round points are the new 6K taxa we added in the 16K tree. In these figures, We annotated 17 taxa in the 16K tree that might be reducing the accuracy of uDance and APPLES-2 in large clusters (subtrees) that include some of the densely sampled species such as Salmonella, E. coli, TB, etc. The pattern is clear that these contaminated genomes can be characterized by a large contamination ratio, near 100% CSS, and high database identity. We do not remove high CSS taxa if their contamination percentage is low, since uDance performs whole-genome-based placement, and it's tolerant to low levels of contamination. Removing taxa satisfying both CSS $>= 0.5$ and Contamination ratio $>= 0.25$ removes 195 taxa from the 16K tree. 171 of them (87%) fail QC in GTDB. Of 195, 37 taxa are also present in WoL tree. 29 of these 37 don't pass GTDB QC.

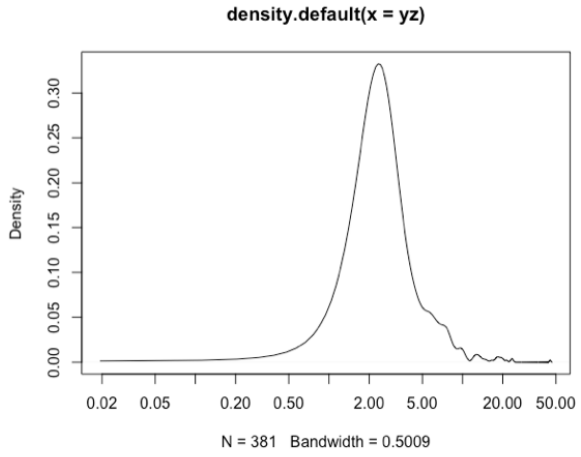


Figure 7.15. WoL heights distribution. This distribution is similar to lognormal. Height is defined as median root to tip distance after minvar rooting.

Appendix A

Supplementary materials for "TreeCluster: Clustering Biological Sequences using Phylogenetic Trees"

A.1 Proofs and supplementary algorithms

A.1.1 Proofs for the Max-diameter min-cut partitioning problem

Proof for Theorem 1. We use induction. The base case for the induction is the simple rooted tree with root u and two leaves u_l and u_r. If $w_l + w_r > \alpha$ the algorithm cuts the longer branch whereas if $w_l + w_r \leq \alpha$ no branch is cut. In both cases, the theorem holds.

The inductive hypothesis is that for a node u, the algorithm has computed $A(u_l)$, $A(u_r)$, $B(u_l)$, and $B(u_r)$ optimally. We need to prove that a solution other than the one computed by our algorithm i) cannot have a lower number of clusters, call it $A'(u)$, and ii) when $A'(u) = A(u)$, cannot have a lower distance to the farthest connected leaf, call it $B'(u)$.

When $B(u_l) + w_l + B(u_r) + w_r \leq \alpha$, we have $A(u) = A(u_l) + A(u_r) - 1$, which is the minimum possible by inductive hypothesis and the fact that the number of clusters cannot go down by more than one on node u. Also, $B(u)$ is optimal by construction.

When $B(u_l) + w_l + B(u_r) + w_r > \alpha$, without loss of generality, assume that $B(u_l) + w_l \geq B(u_r) + w_r$ and thus, the algorithm cuts the (u, u_l) branch, getting $A(u) = A(u_l) + A(u_r)$ and $B(u) = B(u_r) + w_r$. Note that $A'(u) < A(u)$ is only possible if $A'(u_l) = A(u_l)$ and $A'(u_r) = A(u_r)$

and we do not cut any branch at u in the alternative clustering. However, this scenario is *not* possible because

$$B'(u_l) + w_l + B'(u_r) + w_r \geq B(u_l) + w_l + B(u_r) + w_r > \alpha$$

where the first inequality follows from the inductive hypothesis and the final inequality shows that we will have to cut a branch in any alternative setting. Finally, we need to show that an alternative solution with $A'(u) = A(u)$ but $B'(u) < B(u)$ is not possible. The inequality requires that either $B'(u_l) < B(u_l)$ or $B'(u_r) < B(u_r)$. First, consider the $B'(u_l) < B(u_l)$ case, which is possible only if $A'(u_l) = A(u_l) + 1$. Note that $A'(u) = A(u)$ requires $A'(u_r) = A(u_r)$ (and thus $B'(u_r) = B(u_r)$) and that $B'(u_l) + w_l + B(u_r) + w_r < \alpha$, which is possible. Under this condition, we find:

$$B'(u) = max(B'(u_l) + w_l, B(u_r) + w_r) \geq B(u_r) + w_r = B(u) \tag{A.1}$$

If instead $B'(u_r) < B(u_r)$, similar conditions can be written, resulting in

$$B'(u) = max(B(u_l) + w_l, B'(u_r) + w_r) \geq B(u_l) + w_l \geq B(u_r) + w_r = B(u) \tag{A.2}$$

Thus, $A(u)$ and $B(u)$ are optimal when $B(u_l) + w_l + B(u_r) + w_r > \alpha$.

□

Proof for Corollary 1. Let o_r and o_l denote the right and the left child of the root of T^o. Every edge in T can be mapped to T^o except the edge (o_r, o_l), from which we define a mapping to (o, o_r) (w.l.o.g). Using this mapping, the optimal clustering (i.e., optimal cut-set) on T can be translated to an alternative Max-diameter min-cut partitioning on T^o. However, by Theorem 1, $A(o)$ is optimal and cannot be improved by any alternative partitioning. Since any admissible clustering on T^o is also admissible on T, Algorithm 1 minimizes the number of clusters.

□

A.1.2 Linear-time solution for the Sum-length min-cut partitioning problem

Algorithm 4: Linear-time solution for Sum-length min-cut partitioning

Input: A tree $T^o = (V, E)$ and a threshold α

$B(u) \leftarrow 0$ **for** $v \in V$
for $u \in$ *post order traversal of internal nodes of* T^o **do**
  if $B(u_l) + w_l + B(u_r) + w_r > \alpha$ **then**
    if $B(u_l) + w_l \leq B(u_r) + w_r$ **then**
      $E \leftarrow E - \{(u, u_r)\}$
      $B(u) \leftarrow B(u_l) + w_l$
    else
      $E \leftarrow E - \{(u, u_l)\}$
      $B(u) \leftarrow B(u_r) + w_r$
  else
    $B(u) \leftarrow B(u_l) + w_l + B(u_r) + w_r$
return *Leafsets of every connected component in* T^o

We now show that Algorithm 4 is correct. Let $A(u)$ be the minimum number of clusters under U all with a diameter less than α; i.e., $A(o)$ is the objective function.

Theorem 4. *Algorithm 1 computes a clustering with minimum $A(o)$ for rooted tree T^o. In addition, among all possible such clusterings, the algorithm picks the solution with minimum $B(o)$.*

Proof. The proof uses induction. The base case for the induction is the simple rooted tree with root u and two leaves u_l and u_r. If $w_l + w_r > \alpha$, the algorithm cuts the longer branch, whereas if $w_l + w_r \leq \alpha$, no branch is cut. In both cases, the theorem holds.

The inductive hypothesis is that, for a node u, the algorithm has computed $A(u_l)$, $A(u_r)$, $B(u_l)$, and $B(u_r)$ optimally. We need to prove that a solution other than the one computed by our algorithm i) cannot have a lower number of clusters, call it $A'(u)$, and ii) when $A'(u) = A(u)$, cannot have a lower distance to the farthest connected leaf, call it $B'(u)$.

When $B(u_l) + w_l + B(u_r) + w_r \leq \alpha$, we have $A(u) = A(u_l) + A(u_r) - 1$, which is the

minimum possible by the inductive hypothesis along with the fact that the number of clusters cannot decrease by more than one on node u. Also, $B(u)$ is optimal by construction.

When $B(u_l)+w_l+B(u_r)+w_r > \alpha$, without loss of generality, assume that $B(u_l)+w_l \geq B(u_r)+w_r$, and thus, the algorithm cuts the (u,u_l) branch, resulting in $A(u) = A(u_l)+A(u_r)$ and $B(u) = B(u_r)+w_r$. Note that $A'(u) < A(u)$ is only possible if $A'(u_l) = A(u_l)$ and $A'(u_r) = A(u_r)$ and we do not cut any branch at u in the alternative clustering. However, this scenario is *not* possible because

$$B'(u_l)+w_l+B'(u_r)+w_r \geq B(u_l)+w_l+B(u_r)+w_r > \alpha$$

where the first inequality follows from the inductive hypothesis and the final inequality shows that we will have to cut a branch in any alternative setting. Finally, we need to show that an alternative solution with $A'(u) = A(u)$ but $B'(u) < B(u)$ is not possible. The inequality requires that either $B'(u_l) < B(u_l)$ or $B'(u_r) < B(u_r)$. First, consider the $B'(u_l) < B(u_l)$ case, which is possible only if $A'(u_l) = A(u_l)+1$. Note that $A'(u) = A(u)$ requires $A'(u_r) = A(u_r)$ (and thus $B'(u_r) = B(u_r)$) and that $B'(u_l)+w_l+B(u_r)+w_r < \alpha$, which is possible. Under this condition, we find

$$B'(u) = B'(u_l)+w_l+B(u_r)+w_r \geq B(u_r)+w_r = B(u) \tag{A.3}$$

If, instead, $B'(u_r) < B(u_r)$, similar conditions can be written, resulting in

$$B'(u) = B(u_l)+w_l+B'(u_r)+w_r \geq B(u_l)+w_l \geq B(u_r)+w_r = B(u) \tag{A.4}$$

Thus, $A(u)$ and $B(u)$ are optimal when $B(u_l)+w_l+B(u_r)+w_r > \alpha$.

□

Corollary 3. *Let C' be the cut-set obtained by running Algorithm 1 on any arbitrary rooting T^o of unrooted tree T. C' optimally solves the Max-diameter min-cut partitioning problem.*

Proof. Let o_r and o_l denote the right and the left child of the root of T^o. Every edge in T can be mapped to T^o except the edge (o_r, o_l), from which we define a mapping to (o, o_r) (w.l.o.g). Using this mapping, the optimal clustering (i.e., the optimal cut-set) on T can be translated to an alternative Max-diameter min-cut partitioning on T^o. However, by Theorem 1, $A(o)$ is optimal and cannot be improved by any alternative partitioning. Since any admissible clustering on T^o is also admissible on T, Algorithm 1 minimizes q.

□

A.1.3 Proofs for the Single-linkage min-cut partitioning problem

Proof of Proposition 1. ($\Leftarrow$) If $d(a,b) \leq \alpha$ but a and b are in distinct clusters L_a, L_b respectively, N can be reduced by one by simply merging L_a and L_b. $f_T(L_a \cup L_b) \leq \alpha$ is satisfied if for any split of $L_a \cup L_b$, there exists a pair of leaves that are from distinct splits and are within α threshold. For any pair of non-empty sets S and S' that satisfy $S \subset L_a$ and $S' \subset L_b$, we have $\min_{j \in S \cup S', k \in (L_a \cup L_b) - (S \cup S')} d(j,k) \leq \min_{j \in S, k \in L_a - S} d(j,k) \leq \alpha$ and $\min_{j \in S \cup (L_b - S'), k \in S' \cup (L_a - S)} d(j,k) \leq \min_{j \in S, k \in L_a - S} d(j,k) \leq \alpha$. On the other hand, $\min_{j \in L_a, k \in L_b - S} d(j,k) \leq d(a,b) \leq \alpha$. This concludes that for $L = L_a \cup L_b$, $f_T(L) \leq \alpha$ is satisfied. L_a and L_b can still be merged if the chain $\mathscr{H}$ described above exists. It is trivial to show that there is a link $\langle c_i, c_{i+1} \rangle$ in $\mathscr{H}$ such that $c_i \in L_a$ and $c_{i+1} \notin L_a$. Using the argument above, we can iterate over $\mathscr{H}$ and keep merging clusters (and decrease N) every time we see such a link until we finally merge L_a with L_b.

($\Rightarrow$) We describe a procedure to compute the chain $\mathscr{H}$. If a and b in the same cluster L, $\min_{k \in L - \{a\}} d(a,k) \leq \max_{S \subset L} \{ \min_{j \in S, k \in L - S} d(j,k) \} \leq \alpha$ holds, implying that there is a leaf c_1 in set $L - \{a\}$ such that $d(a, c_1) \leq \alpha$. If $c_1 = b$, theorem follows. If $c_1 \neq b$, we union a and c_1, call the union set L_a, and add the link $a \rightarrow c_1$ to $\mathscr{H}'$. Iteratively, we find the pair $\langle j,k \rangle$ that yields to $\min_{j \in L_a, k \in L - L_a} d(j,k)$, add the link $j \rightarrow k$ to $\mathscr{H}'$, and add k to L_a until we finally add b to L_a. The elements forming the path between a, and b in $\mathscr{H}'$, which can be computed using depth-first-search, constitute a valid chain $\mathscr{H}$. □

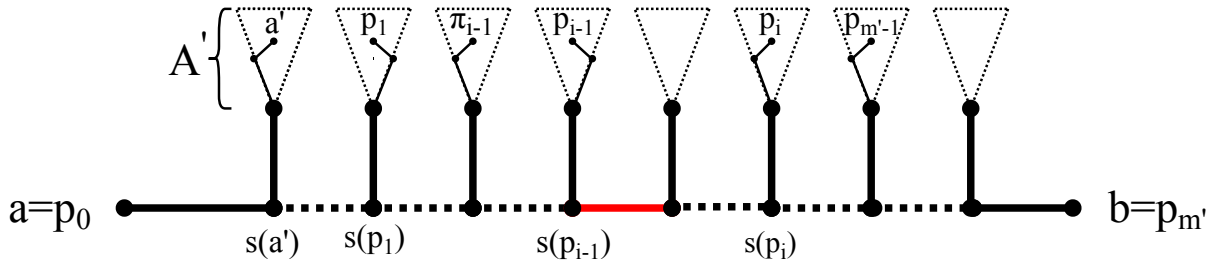


Figure A.1. A sketch showing the setup for constructing the chain $\mathscr{H}$.

Proof for Theorem 2. Let $a \rightsquigarrow b$ be the path between leaves a and b on T. Fixing a and b, for each node j, we use the term *support of* j, denoted by $s(j)$, to refer to the unique node on all the three paths $a \rightsquigarrow b$, $a \rightsquigarrow j$, and $b \rightsquigarrow j$. We refer to a group of leaves that share a mutual support with respect to a and b as a *bubble* (e.g. triangles in Fig A.1). Among all bubbles branching out of $a \rightsquigarrow b$, let the one with the closest support to a be A'. We name the leaf closest to a on A' as a' (Fig A.1).

We start with the observation that if $d(a,b) \leq \alpha$ holds, the algorithm will never cut any edge on $a \rightsquigarrow b$. For every internal node u on $a \rightsquigarrow b$, let v and w be the adjacent nodes on $a \rightsquigarrow u$ and $u \rightsquigarrow b$, respectively. Also, let p_a be the closest leaf to u whose support $s(p_a)$ is on $a \rightsquigarrow u$, and let p_b be the closest leaf to u whose support $s(p_b)$ is on $u \rightsquigarrow b$. Now, note that $d(p_a,u)+d(u,p_b) \leq d(a,u)+d(u,b) \leq \alpha$ holds, so regardless of the rooting, (v,u) and (u,w) are never cut by Algorithm 2.

If a chain $\mathscr{H}$ exists, due to the previous observation, there are no cuts on $c_i \rightsquigarrow c_{i+1}$ for every $0 \leq i \leq m$. Consequently, a and b are connected through a path and are thus in the same cluster.

Assume Algorithm 2 places a and b on the same cluster, i.e., it does not cut any edge on $a \rightsquigarrow b$. We present a procedure to generate a chain $\mathscr{H}$ as described in Definition 5. But we first need some definitions. We define $p_0 = a$ and $p_{m'} = b$. For $1 \leq i \leq m'$, let p_i denote the closest leaf to p_{i-1} whose support $s(p_i)$ is on $p_{i-1} \rightsquigarrow b$ and $s(p_i) \neq s(p_{i-1})$; i.e., p_i is in the bubble to the right of the bubble of p_{i-1}. Conversely, for $1 \leq i \leq m'$, let π_i denote the closest leaf to p_i

whose support is on $a \leftrightsquigarrow s(p_{i-1})$; i.e., is in a bubble to the left of p_i. Also define $\pi_1 = a$. We can also show that every $\pi_i \in \{p_0 \dots p_{i-1}\}$. If a π_i is not equal to one of $\{p_0 \dots p_{i-1}\}$, then, $s(\pi_i)$ has to be on $s(p_{j-1}) \leftrightsquigarrow s(p_j)$ for some j. However, we would have $d(p_{j-1}, \pi_i) \leq d(p_{j-1}, p_j)$, which contradicts the definition of p_i.

Now we construct the chain. The fact that Algorithm 2 retains $(a, s(a'))$ indicates that $min(d(a,a'), d(a,p_1)) = d(a,p_1) \leq \alpha$; therefore, we add $a \rightarrow p_1$ to an auxiliary graph $\mathscr{H}'$. Now, consider Algorithm 2 when it processes the node $s(p_{i-1})$ for $1 < i$. The fact that the first edge on path $s(p_{i-1}) \leftrightsquigarrow s(p_i)$ (shown in red in Fig A.1) is not cut indicates that either $d(\pi_{i-1}, p_i) \leq \alpha$ or $d(p_{i-1}, p_i) \leq \alpha$. Depending on which is true, we add a link from $\pi_{i-1} \rightarrow p_i$ or $p_{i-1} \rightarrow p_i$ to $\mathscr{H}'$. We repeat this process for all i until we reach $i = m'$, where we add an edge to $p_{m'} = b$. Noting that $\pi_i \in \{p_0 \dots p_{i-1}\}$, the $\mathscr{H}'$ graph becomes a directed tree, rooted at a with a directed path to the leaf b. This directed path constitutes the valid chain $\mathscr{H}$. □

A.1.4 Mean-diameter clustering with clade constraint

Algorithm 5: AVERAGE DIAMETER CLADE Average diameter clade min-cut partitioning

for $u \in$ *post order traversal of* T^o **do**
  $totPairDist[u] \leftarrow 0$; $totLeafDist[u] \leftarrow 0$;
  if u ***in*** $\mathscr{L}$ **then**
    $numLeaves[u] \leftarrow 1$; $avgPairDist[u] \leftarrow 0$;
  else
    $numLeaves[u] \leftarrow numLeaves[u_l] + numLeaves[u_r]$;
    $totPairDist[u] \leftarrow totPairDist[u_l] + totPairDist[u_r] + totLeafDist[u_l] \times numLeaves[u_r] + totLeafDist[u_r] \times numLeaves[u_l]$;
    $totLeafDist[u] \leftarrow totLeafDist[u_l] + w_l \times numLeaves[u_l] + totLeafDist[u_r] + w_r \times numLeaves[u_r]$;
    $avgPairDist[u] \leftarrow totPairDist[u] / \binom{numLeaves[u]}{2}$;
$toExplore \leftarrow$ queue containing the root of T^o;
while $toExplore \neq \emptyset$ **do**
  $curr \leftarrow toExplore.dequeue()$;
  if u ***not in*** $\mathscr{L}$ ***and*** $avgPairDist[u] > \alpha$ **then**
    $E \leftarrow E \setminus (u, u_l)$; $E \leftarrow E \setminus (u, u_r)$;
    $toExplore.enqueue(u_l)$; $toExplore.enqueue(u_r)$;
return *Leafsets of every connected component in* T^o

All optimal solutions

Lemma 6. *Let* $\{e_1, e_2, \cdots, e_m\}$ *be the set of edges in an unrooted tree* T*. Consider the following algorithm: root* T *at* e_j *and run Algorithm 1, and let* $\mathscr{S}_j$ *denote the set of edges cut by the algorithm in this run. Any optimal clustering for* T *has to draw its cut-set from* $\Sigma = \cup_{j=1}^{m} \mathscr{S}_j$*.*

Proof. The proof is by contradiction. Assume there is an optimal cut-set $\mathscr{S}'$ that contains an edge e_i such that $e_i \notin \Sigma$. Consider the rooting of T at e_i. Denote the root of this tree as v, the immediate left and right branches of v as e_l and e_r, and the left and right child nodes of v as v_l and v_r. Note that the concatenation of e_l and e_r corresponds to e_i in T; thus, $e_l \notin \mathscr{S}_j$ and $e_r \notin \mathscr{S}_j$. When e_i is removed from T, two new trees form, called T_l (the one containing the node v_l) and T_r (the one containing the node v_r). If p cuts in $\mathscr{S}'$ are in T_r, and if q cuts in $\mathscr{S}'$ are in T_l, then $|\mathscr{S}'| = p+q+1$. The number of cuts in $\mathscr{S}'$ and $\mathscr{S}_j$ are equal, and e_l and e_r are not cut, which implies that either the tree rooted by v_l or v_r has an alternative clustering with one less cut. By the design of Algorithm 1, if this was the case, the algorithm would have chosen the alternative cut. □

A.2 Commands and parameters

Ancestral state reconstruction using TreeTime.

Each cluster tree is first rooted at its balance point using MinVar rooting version (commit 8c1581a):

```
$ python FastRoot.py -i unrooted_tree.nwk -m MV
-o rooted_tree.nwk
```

Before performing maximum likelihood ancestral state reconstruction, we inferred GTR parameters from the input tree using RAxML v8.2.12:

```
$ raxmlHPC-PTHREADS -f e -t ../input_tree.nwk -s aln.fa
-m GTRGAMMA -n tre -T 4
```

We manually hardcoded those parameters into built-in TN93 parameters matrix in treetime software (v0.5.5) and reconstructed ancestral states of rooted tree using this command:

```
$ treetime ancestral --tree rooted_tree.nwk --aln aln.fa
--outdir outdir --gtr TN93
```

Listing A.1. Default FAVITES Parameters

```
{
    "ContactNetworkGenerator": "Communities",
    "cn_generators": [{
        "ContactNetworkGenerator": "BarabasiAlbert",
        "num_cn_nodes": 5000,
        "num_edges_from_new": 5
    }]*20,
    "cn_p_across": 1/(2*19*5000),
    "NodeEvolution": "VirusTreeSimulator",
    "vts_growthRate": 2.851904,
    "vts_max_attempts": 100,
    "vts_model": "logistic",
    "vts_n0": 1,
    "vts_t50": -2,
    "NumBranchSample": "Single",
    "NumTimeSample": "Once",
    "SeedSelection": "Random",
    "num_seeds": 15000,
    "SeedSequence": "VirusNonHomYuleHeightGTRGamma",
    "seed_height": 25,
    "seed_speciation_rate_func": "exp(-t**2)+1",
    "viral_sequence_type": "HIV1-B-DNA-POL-LITTLE",
    "seqgen_freq_a": 0.392,
    "seqgen_freq_c": 0.165,
    "seqgen_freq_g": 0.212,
```

```
"seqgen_freq_t": 0.232,
"seqgen_a_to_c": 1.765707,
"seqgen_a_to_g": 9.587649,
"seqgen_a_to_t": 0.691915,
"seqgen_c_to_g": 0.863348,
"seqgen_c_to_t": 10.282617,
"seqgen_g_to_t": 1.0,
"seqgen_gamma_shape": 0.405129,
"seqgen_num_gamma_rate_categories": "",
"SequenceEvolution": "GTRGammaSeqGen",
"Sequencing": "Perfect",
"SourceSample": "Random",
"TimeSample": "GranichFirstART",
"TransmissionTimeSample": "HIVARTGranichGEMF",
"end_time": 10,
"hiv_freq_ns": 0,
"hiv_freq_i3": 0,
"hiv_freq_i4": 0,
"hiv_freq_a3": 0,
"hiv_freq_a4": 0,
"hiv_freq_d": 0,
"hiv_freq_s": 100000-15000,
"hiv_freq_i1": 50,
"hiv_freq_i2": 5094,
"hiv_freq_a1": 9,
"hiv_freq_a2": 15000-50-5094-9,
```

```
"hiv_a1_to_a2": 4.333333333333333,
"hiv_a1_to_d": 0,
"hiv_a1_to_i1": 0.48,
"hiv_a2_to_a3": 0,
"hiv_a2_to_d": 0,
"hiv_a2_to_i2": 0.48,
"hiv_a3_to_a4": 0,
"hiv_a3_to_d": 0,
"hiv_a3_to_i3": 0,
"hiv_a4_to_d": 0,
"hiv_a4_to_i4": 0,
"hiv_i1_to_a1": 1.0,
"hiv_i1_to_d": 0,
"hiv_i1_to_i2": 8.666666666666666,
"hiv_i2_to_a2": 1.0,
"hiv_i2_to_d": 0,
"hiv_i2_to_i3": 0,
"hiv_i3_to_a3": 0,
"hiv_i3_to_d": 0,
"hiv_i3_to_i4": 0,
"hiv_i4_to_a4": 0,
"hiv_i4_to_d": 0,
"hiv_ns_to_d": 0,
"hiv_ns_to_s": 999999,
"hiv_s_to_d": 0,
"hiv_s_to_i1_by_a1": 0.005625,
```

```
    "hiv_s_to_i1_by_a2": 0,
    "hiv_s_to_i1_by_a3": 0,
    "hiv_s_to_i1_by_a4": 0,
    "hiv_s_to_i1_by_i1": 0.1125,
    "hiv_s_to_i1_by_i2": 0.0225,
    "hiv_s_to_i1_by_i3": 0,
    "hiv_s_to_i1_by_i4": 0,
    "hiv_s_to_i1_seed": 0,
    "TreeUnit": "TruncatedNormal",
    "tree_rate_loc": 0.0008,
    "tree_rate_max": float('inf'),
    "tree_rate_min": 0,
    "tree_rate_scale": 0.0005,
    "ContactNetwork": "NetworkX",
    "Driver": "Default",
    "EndCriteria": "GEMF",
    "Logging": "File",
    "NodeAvailability": "Perfect",
    "TransmissionNodeSample": "GEMF",
    "TreeNode": "Simple",
    "gemf_path": "GEMF",
    "hmmemit_path": "hmmemit",
    "java_path": "java",
    "out_dir": "FAVITES_output",
    "seqgen_path": "seq-gen",
}
```

Appendix B

Supplementary materials for "APPLES: Scalable Distance-based Phylogenetic Placement with or without Alignments"

B.1 Proofs and derivations

Recall the following notations.

- For any node u and exponents $a \in \mathbb{Z}$ and $b \in \mathbb{N}^+$, let
  - $S(a,b,u) = \sum_{i \in g(u)} \delta_{qi}^a d_{ui}^b$
  - $R(a,b,u) = \sum_{i \notin g(u)} \delta_{qi}^a d_{p(u)i}^b$ defined for $u \in V \setminus \{1\}$
- For $b = 0$, let $S(a,0,u) = \sum_{i \in g(u)} \delta_{qi}^a$ and let $S'(a,u)$ be a shorthand for $S(a,0,u)$. Similarly, let $R(a,0,u) = R'(a,u) = \sum_{i \notin g(u)} \delta_{qi}^a$.

B.1.1 Proof of Lemma 1

Proof. Recall the dynamic programming recursions of Equations 3.3 and 3.4:

$$S(a,b,u) = \sum_{j=0}^{b} \sum_{v \in c(u)} l(v)^j \binom{b}{j} S(a,b-j,v), \quad \text{for } u \notin \mathscr{L} \setminus \{1\}$$

$$R(a,b,u) = \sum_{j=0}^{b} \Big(l(p(u))^j \binom{b}{j} R(a,b-j,p(u)) + \sum_{v \in sib(u)} l(v)^j \binom{b}{j} S(a,b-j,v) \Big) \text{ for } u \notin \{1,1'\}$$

Since u is not a leaf, for each leaf $i \in g(u)$, there exists a $v \in c(u)$ such that the directed path from u to i passes through v. Therefore every leaf i can be grouped under its corresponding v.

$$\begin{aligned} S(a,b,u) = \sum_{i \in g(u)} \delta_{qi}^a d_{ui}^b = \sum_{v \in c(u)} \sum_{i \in g(v)} \delta_{qi}^a \big(l(v) + d_{vi}\big)^b &= \sum_{j=0}^{b} \sum_{v \in c(u)} \sum_{i \in g(v)} \delta_{qi}^a d_{vi}^{b-j} l(v)^j \binom{b}{j} \\ &= \sum_{j=0}^{b} \sum_{v \in c(u)} l(v)^j \binom{b}{j} S(a,b-j,v) \end{aligned}$$

Similarly, given the condition $u \neq 1$, for each leaf $i \notin g(u)$, either (1) there exists $v \in sib(u)$ such that the directed path from $p(u)$ to i passes through v, or (2) undirected path between i and $p(u)$ passes through $p(p(u))$.

$$\begin{aligned} R(a,b,u) &= \sum_{i \notin g(u)} \delta_{qi}^a d_{p(u)i}^b = \sum_{v \in sib(u)} \sum_{i \in g(v)} \delta_{qi}^a \big(l(v) + d_{vi}\big)^b + \sum_{i \notin g(p(u))} \delta_{qi}^a \big(l(p(u)) + d_{p(p(u))i}\big)^b \\ &= \sum_{j=0}^{b} \sum_{v \in sib(u)} \sum_{i \in g(v)} \delta_{qi}^a d_{vi}^{b-j} l(v)^j \binom{b}{j} + \sum_{j=0}^{b} \sum_{i \notin g(p(u))} \delta_{qi}^a d_{p(p(u))i}^{b-j} l(p(u))^j \binom{b}{j} \\ &= \sum_{j=0}^{b} \sum_{v \in sib(u)} l(v)^j \binom{b}{j} S(a,b-j,v) + \sum_{j=0}^{b} l(p(u))^j \binom{b}{j} R(a,b-j,p(u)) \end{aligned}$$

Boundary conditions follow from definitions. For $u \notin \mathscr{L} \setminus \{1\}$, since $d_{ii} = 0$, we have

$S(a,b,u) = 0$ and it's trivial to see $S'(a,u) = \delta_{qu}^{a}$. For $R(,,)$ recursions, the boundary case happens at the unique child of the root, which we denote as $1'$. Based on the definition, since the only $i \notin g(1')$ is 1, and $d_{p(1')1}^{b} = 0$, we trivially have $R(a,b,1') = 0$. For $b = 0$, $R'(a,1') = \delta_{q1}^{a}$.

A post-order traversal on T^* can compute $S(a,b,u)$, and a subsequent pre-order traversal can compute $R(a,b,u)$, both in constant time and using constant memory per node. Recall that a and b are both no more than k, which is a constant. Thus, time and memory complexity of this dynamic programming is $\Theta(bn)$, which translates to $\Theta(n)$ in least squares setting, where $b \leq 2$. □

B.1.2 Proof of Lemma 2.

Recall $w_{qi} = \delta_{qi}^{-k}$ and that Equation 2:

$$Q(P) = \sum_{i=1}^{n} w_{qi}(\delta_{qi} - d_{qi}(P))^2 = \sum_{i=1}^{n} \delta_{qi}^{-k}(\delta_{qi} - d_{qi}(P))^2$$

Proof. Equation 3.2 can be re-written as:

$$Q(P) = \sum_{i \in g(u)} \delta_{qi}^{-k}\big(\delta_{qi} - d_{ui}(P) - x_1 + x_2 - l(u)\big)^2 + \sum_{i \notin g(u)} \delta_{qi}^{-k}\big(\delta_{qi} - d_{p(u)i}(P) - x_1 - x_2\big)^2 \tag{B.1}$$

By simple rearrangement of the terms, we can rewrite Equation B.1 as follows.

$$\begin{aligned} Q(P(u,x_1,x_2)) = {} & R'(2-k,u) + S'(2-k,u) + R(-k,2,u) + S(-k,2,u) \\ & + 2\big(x_1 + x_2\big)\big(R(-k,1,u) - R'(1-k,u)\big) \\ & + 2\big(x_1 + l(u) - x_2\big)\big(S(-k,1,u) - S'(1-k,u)\big) \\ & + \big(x_1 + x_2\big)^2 R(-k,1,u) + \big(x_1 + l(u) - x_2\big)^2 S(-k,1,u) \\ & - 2R(1-k,1,u) - 2S(1-k,1,u) \end{aligned} \tag{B.2}$$

Note that computing $Q(P(u,x_1,x_2))$ requires only $S(,,u)$ and $R(,,u)$ values and $l(u)$. Thus, computing $Q(P)$ requires only computing $S(a,b,u)$ and $R(a,b,u)$ values for $-k \leq a \leq 2-k$ and

$0 \leq b \leq 2$.

□

Proof of Lemma 3.

Recall definitions

$$S(a,b,u) = \sum_{i \in g(u)} \delta_{qi}^{a} d_{ui}^{b} \ \ (\text{for } b > 0) \qquad \text{and} \qquad S'(a,u) = S(a,0,u) = \sum_{i \in g(u)} \delta_{qi}^{a}$$

$$R(a,b,u) = \sum_{i \notin g(u)} \delta_{qi}^{a} d_{p(u)i}^{b} \ \ (\text{for } b > 0) \qquad \text{and} \qquad R'(a,u) = R(a,0,u) = \sum_{i \notin g(u)} \delta_{qi}^{a}$$

and recall Eq. B.1:

$$Q(P) = \sum_{i \in g(u)} \delta_{qi}^{-k} \big(\delta_{qi} - d_{ui}(P) - x_1 + x_2 - l(u)\big)^2 + \sum_{i \notin g(u)} \delta_{qi}^{-k} \big(\delta_{qi} - d_{p(u)i}(P) - x_1 - x_2\big)^2 .$$

Proof. We take the derivative of Eq. B.1 with respect to x_1 and set it equal to zero:

$$\begin{aligned}
\frac{\partial Q(P)}{\partial x_1} &= -2 \sum_{i \in g(u)} \delta_{qi}^{-k} \big(\delta_{qi} - d_{ui}(P) - x_1 + x_2 - l(u)\big) \\
&\quad -2 \sum_{i \notin g(u)} \delta_{qi}^{-k} \big(\delta_{qi} - d_{p(u)i}(P) - x_1 - x_2\big) = 0 \\
&\Longrightarrow \Big(\sum_{i \in g(u)} \delta_{qi}^{-k} + \sum_{i \notin g(u)} \delta_{qi}^{-k} \Big) x_1 + \Big(- \sum_{i \in g(u)} \delta_{qi}^{-k} + \sum_{i \notin g(u)} \delta_{qi}^{-k} \Big) x_2 \\
&\quad - \sum_{i \in g(u)} \delta_{qi}^{1-k} - \sum_{i \notin g(u)} \delta_{qi}^{1-k} \\
&\quad + \sum_{i \in g(u)} \delta_{qi}^{-k} d_{ui}(P) + \sum_{i \notin g(u)} \delta_{qi}^{-k} d_{p(u)i}(P) + l(u) \sum_{i \in g(u)} \delta_{qi}^{-k} = 0 \\
&\Longrightarrow \big(S'(-k,u) + R'(-k,u)\big) x_1 + \big(-S'(-k,u) + R'(-k,u)\big) x_2 = \\
&\quad S'(1-k,u) + R'(1-k,u) - S(-k,1,u) - R(-k,1,u) - l(u) S'(-k,u)
\end{aligned}$$

Similarly,

$$\begin{aligned}
\frac{\partial Q(P)}{\partial x_2} &= 2\sum_{i\in g(u)} \delta_{qi}^{-k}\big(\delta_{qi} - d_{ui}(P) - x_1 + x_2 - l(u)\big) \\
&\quad -2\sum_{i\notin g(u)} \delta_{qi}^{-k}\big(\delta_{qi} - d_{p(u)i}(P) - x_1 - x_2\big) = 0 \\
&\Longrightarrow \Big(-\sum_{i\in g(u)} \delta_{qi}^{-k} + \sum_{i\notin g(u)} \delta_{qi}^{-k}\Big)x_1 + \Big(\sum_{i\in g(u)} \delta_{qi}^{-k} + \sum_{i\notin g(u)} \delta_{qi}^{-k}\Big)x_2 \\
&\quad + \sum_{i\in g(u)} \delta_{qi}^{1-k} - \sum_{i\notin g(u)} \delta_{qi}^{1-k} - \sum_{i\in g(u)} \delta_{qi}^{-k} d_{ui}(P) \\
&\quad + \sum_{i\notin g(u)} \delta_{qi}^{-k} d_{p(u)i}(P) - l(u)\sum_{i\in g(u)} \delta_{qi}^{-k} = 0 \\
&\Longrightarrow \big(-S'(-k,u) + R'(-k,u)\big)x_1 + \big(+S'(-k,u) + R'(-k,u)\big)x_2 = \\
&\quad -S'(1-k,u) + R'(1-k,u) + S(-k,1,u) - R(-k,1,u) + l(u)S'(-k,u)
\end{aligned}$$

These two linear equations have a unique solution for the pair x_1, x_2 if and only if the following matrix has the full rank:

$$H = \begin{bmatrix} R'(-k,u) + S'(-k,u) & R'(-k,u) - S'(-k,u) \\ R'(-k,u) - S'(-k,u) & R'(-k,u) + S'(-k,u) \end{bmatrix}.$$

Determinant of H is $\det(H) = 4R'(-k,u)S'(-k,u)$. Assuming that $\delta_{qi} > 0$ for all $i \in \mathscr{L}$, both $R'(-k,u) > 0$ and $S'(-k,u) > 0$ hold. Therefore, H has the full rank. However, $\delta_{qi} = 0$ for $q \neq i$ can be encountered on real data, especially for low divergence times, low evolutionary rates, or short sequences. In this case, APPLES is designed to place q on the pendant edge of i with $x_1 = 0$ and $x_2 = l(i)$. In case there are multiple leaves i that satisfy $\delta_{qi} = 0$ for $q \neq i$, we pick one of them arbitrarily. □

B.1.3 Proof of Theorem 3

Proof. First, using two traversals of the tree, we compute all the $S(a,b,u)$ and $R(a,b,u)$ values by Lemma 1. To find the optimal placement edge, we first optimize $Q(P(u,x_1,x_2))$ for all $u \in V \setminus \{1\}$. By Lemma 3, this task requires only constant time after the precompuations. Then, for each node, we compute $Q(P(u,x_1,x_2))$ in constant time for the optimal u,x_1,x_2 by Lemma 2. Thus, each node is processed in linear time and the whole optimization requires linear time. Note that the system of equations (shown in Lemma 3) will not have a solution iff $\delta_{qi} \leq 0$ for some i; if there is $\delta_{qi} = 0$, we make q sister to i, breaking ties arbitrarily. □

Proof of Lemma 4

Proof. Eigenvalues of the Hessian matrix of $Q(P(u,x_1,x_2))$ are $2R'(-k,u)$ and $2S'(-k,u)$, which are both non-negative since $\delta_{qi} \geq 0$ for $i \in \mathscr{L}$. Thus, the Hessian matrix is positive semidefinite and therefore $P(u,x_1,x_2)$ is a convex function of x_1 and x_2. □

B.2 Supplementary Tables

Table B.1. GenBank accession numbers and URLs for the dataset of 22 Anopheles genomes

Species	GenBank assembly accession	URL
Anopheles albimanus	GCA_000349125.1	http://www.insect-genome.com/data/genome_download/Anopheles_albimanus/Anopheles_albimanus_genomic.fasta.gz
Anopheles arabiensis	GCA_000349185.1	http://www.insect-genome.com/data/genome_download/Anopheles_arabiensis/Anopheles_arabiensis_genomic.fasta.gz
Anopheles atroparvus	GCA_000473505.1	http://www.insect-genome.com/data/genome_download/Anopheles_atroparvus/Anopheles_atroparvus_genomic.fasta.gz
Anopheles christyi	GCA_000349165.1	http://www.insect-genome.com/data/genome_download/Anopheles_christyi/Anopheles_christyi_genomic.fasta.gz
Anopheles coluzzii	-	http://www.insect-genome.com/data/genome_download/Anopheles_coluzzii/Anopheles_coluzzii_genomic.fasta.gz
Anopheles culicifacies	GCA_000473375.1	http://www.insect-genome.com/data/genome_download/Anopheles_culicifacies/Anopheles_culicifacies_genomic.fasta.gz
Anopheles darlingi	GCA_000211455.3	http://www.insect-genome.com/data/genome_download/Anopheles_darlingi/Anopheles_darlingi_genomic.fasta.gz
Anopheles dirus	GCA_000349145.1	http://www.insect-genome.com/data/genome_download/Anopheles_dirus/Anopheles_dirus_genomic.fasta.gz
Anopheles epiroticus	GCA_000349105.1	http://www.insect-genome.com/data/genome_download/Anopheles_epiroticus/Anopheles_epiroticus_genomic.fasta.gz
Anopheles farauti	GCA_000956265.1	http://www.insect-genome.com/data/genome_download/Anopheles_farauti/Anopheles_farauti_genomic.fasta.gz
Anopheles funestus	GCA_000349085.1	http://www.insect-genome.com/data/genome_download/Anopheles_funestus/Anopheles_funestus_genomic.fasta.gz
Anopheles gambiae	GCA_000150785.1	http://www.insect-genome.com/data/genome_download/Anopheles_gambiae/Anopheles_gambiae_genomic.fasta.gz
Anopheles koliensis	GCA_000956275.1	http://www.insect-genome.com/data/genome_download/Anopheles_koliensis/Anopheles_koliensis_genomic.fasta.gz
Anopheles maculatus	GCA_000473185.1	http://www.insect-genome.com/data/genome_download/Anopheles_maculatus/Anopheles_maculatus_genomic.fasta.gz
Anopheles melas	GCA_000473525.2	http://www.insect-genome.com/data/genome_download/Anopheles_melas/Anopheles_melas_genomic.fasta.gz
Anopheles merus	GCA_000473845.2	http://www.insect-genome.com/data/genome_download/Anopheles_merus/Anopheles_merus_genomic.fasta.gz
Anopheles minimus	GCA_000349025.1	http://www.insect-genome.com/data/genome_download/Anopheles_minimus/Anopheles_minimus_genomic.fasta.gz
Anopheles nili	GCA_000439205.1	http://www.insect-genome.com/data/genome_download/Anopheles_nili/Anopheles_nili_genomic.fasta.gz
Anopheles punctulatus	GCA_000956255.1	http://www.insect-genome.com/data/genome_download/Anopheles_punctulatus/Anopheles_punctulatus_genomic.fasta.gz
Anopheles quadriannulatus	GCA_000349065.1	http://www.insect-genome.com/data/genome_download/Anopheles_quadriannulatus/Anopheles_quadriannulatus_genomic.fasta.gz
Anopheles sinensis	GCA_000441895.2	http://www.insect-genome.com/data/genome_download/Anopheles_sinensis/Anopheles_sinensis_genomic.fasta.gz
Anopheles stephensis	GCA_000300775.2	http://www.insect-genome.com/data/genome_download/Anopheles_stephensi/Anopheles_stephensi_genomic.fasta.gz

Table B.2. GenBank accession numbers and URLs for the dataset of 21 Drosophila genomes

Species	GenBank assembly accession	URL
Drosophila ananassae	GCA_000005115.1	http://www.insect-genome.com/data/genome_download/Drosophila_ananassae/Drosophila_ananassae_genomic.fasta.gz
Drosophila biarmipes	GCA_000233415.2	http://www.insect-genome.com/data/genome_download/Drosophila_biarmipes/Drosophila_biarmipes_genomic.fasta.gz
Drosophila bipectinata	GCA_000236285.2	http://www.insect-genome.com/data/genome_download/Drosophila_bipectinata/Drosophila_bipectinata_genomic.fasta.gz
Drosophila elegans	GCA_000224195.2	http://www.insect-genome.com/data/genome_download/Drosophila_elegans/Drosophila_elegans_genomic.fasta.gz
Drosophila erecta	GCA_000005135.1	http://www.insect-genome.com/data/genome_download/Drosophila_erecta/Drosophila_erecta_genomic.fasta.gz
Drosophila eugracilis	GCA_000236325.2	http://www.insect-genome.com/data/genome_download/Drosophila_eugracilis/Drosophila_eugracilis_genomic.fasta.gz
Drosophila ficusphila	GCA_000220665.2	http://www.insect-genome.com/data/genome_download/Drosophila_ficusphila/Drosophila_ficusphila_genomic.fasta.gz
Drosophila grimshawi	GCA_000005155.1	http://www.insect-genome.com/data/genome_download/Drosophila_grimshawi/Drosophila_grimshawi_genomic.fasta.gz
Drosophila kikkawai	GCA_000224215.2	http://www.insect-genome.com/data/genome_download/Drosophila_kikkawai/Drosophila_kikkawai_genomic.fasta.gz
Drosophila melanogaster	GCA_000778455.1	http://www.insect-genome.com/data/genome_download/Drosophila_melanogaster/Drosophila_melanogaster_genomic.fasta.gz
Drosophila miranda	GCA_000269505.2	http://www.insect-genome.com/data/genome_download/Drosophila_miranda/Drosophila_miranda_genomic.fasta.gz
Drosophila mojavensis	GCA_000005175.1	http://www.insect-genome.com/data/genome_download/Drosophila_mojavensis/Drosophila_mojavensis_genomic.fasta.gz
Drosophila persimilis	GCA_000005195.1	http://www.insect-genome.com/data/genome_download/Drosophila_persimilis/Drosophila_persimilis_genomic.fasta.gz
Drosophila rhopaloa	GCA_000236305.2	http://www.insect-genome.com/data/genome_download/Drosophila_rhopaloa/Drosophila_rhopaloa_genomic.fasta.gz
Drosophila sechellia	GCA_000005215.1	http://www.insect-genome.com/data/genome_download/Drosophila_sechellia/Drosophila_sechellia_genomic.fasta.gz
Drosophila simulans	GCA_000259055.1	http://www.insect-genome.com/data/genome_download/Drosophila_simulans/Drosophila_simulans_genomic.fasta.gz
Drosophila suzukii	GCA_000472105.1	http://www.insect-genome.com/data/genome_download/Drosophila_suzukii/Drosophila_suzukii_genomic.fasta.gz
Drosophila takahashii	GCA_000224235.2	http://www.insect-genome.com/data/genome_download/Drosophila_takahashii/Drosophila_takahashii_genomic.fasta.gz
Drosophila virilis	GCA_000005245.1	http://www.insect-genome.com/data/genome_download/Drosophila_virilis/Drosophila_virilis_genomic.fasta.gz
Drosophila willistoni	GCA_000005925.1	http://www.insect-genome.com/data/genome_download/Drosophila_willistoni/Drosophila_willistoni_genomic.fasta.gz
Drosophila yakuba	GCA_000005975.1	http://www.insect-genome.com/data/genome_download/Drosophila_yakuba/Drosophila_yakuba_genomic.fasta.gz

Table B.3. GenBank accession numbers of microbial species used in contamination removal.

Species	GenBank assembly accession
Pasteurella langaaensis	GCA_003096995.1
Providencia stuartii	GCA_001558855.2
Serratia marcescens	GCA_000783915.2
Shigella flexneri	GCA_000006925.2
Commensalibacter intestini	GCA_002153535.1
Acetobacter malorum	GCA_002153605.1
Acetobacter pomorum	GCA_002456135.1
Lactobacillus plantarum	GCA_000203855.3
Lactobacillus brevis	GCA_003184305.1
Enterococcus faecalis	GCA_002208945.2
Vagococcus teuberi	GCA_001870205.1
Wolbachia	GCA_000022285.1

Table B.4. Assembly-free placement of genome-skims. We show the percentage of correct placements (those that do not increase Δe), average delta error (Δe), and maximum delta error (e_{max}) for APPLES, assignment to the CLOSEST species, and the placement to the position in the backbone (DE-NOVO) over the 61 (a), 22 (b), and 21 (c) placements. Results are shown for skims with 0.1 and 0.5Gbp of reads. Delta error is the increase in the number missing branches between the reference tree and the backbone tree after placing each query.

	0.1G			0.5G		
	%	Δe	e_{max}	%	Δe	e_{max}
(a) **Columbicola**						
APPLES*	97	0.03	1	92	0.08	1
APPLES-ME	84	0.28	5	87	0.21	5
APPLES-HYBRID	87	0.16	2	87	0.16	2
CLOSEST	54	1.15	7	58	0.91	8
DE-NOVO	98	0.02	1	92	0.08	1
(b) **Anopheles**						
APPLES*	95	0.05	1	95	0.05	1
APPLES-ME	95	0.05	1	95	0.05	1
APPLES-HYBRID	95	0.05	1	95	0.05	1
CLOSEST	91	0.09	1	95	0.05	1
DE-NOVO	95	0.05	1	95	0.05	1
(c) **Drosophila**						
APPLES*	71	0.29	1	71	0.33	2
APPLES-ME	67	0.42	2	67	0.48	2
APPLES-HYBRID	67	0.33	1	67	0.38	2
CLOSEST	57	0.62	3	57	0.57	2
DE-NOVO	71	0.29	1	71	0.33	2

B.3 Commands

B.3.1 Sampling Clades

For sampling clades of size at most 250 from a tree "tree.nwk", we used the TreeCluster package, available at https://github.com/niemasd/TreeCluster.

```
#!/bin/bash
python TreeCluster/TreeCluster.py -i 250 -o clusters.txt
-t tree.nwk -m count_max_clade
```

B.3.2 Computing Distance Matrices

APPLES version 1.2.0 can compute JC69 distances between nucleotide sequences. Version 1.1.0 internally uses FastME* (based on FastME version 2.1.6.1) which is available at https://github.com/balabanmetin/FastME-personal-copy. We computed distance matrices based on other models (e.g. TN93+Γ) using following FastME command:

```
#!/bin/bash
fastme -c -dT --gamma=${gamma} -i aln_dna.phy -O dist.mat -T 1
```

where ${*gamma*} is Γ the rate variable.

B.3.3 Backbone tree estimation

When multiple sequence alignment is available, we used the following RAxML command to compute backbone tree for all datasets except RNAsim-VS dataset. We used RAxML version 7.2.6

```
#!/bin/bash
raxmlHPC-PTHREADS -m GTRGAMMA -p 88 -n REF -s aln_dna.phy -T 6
```

For RNAsim-VS dataset, we used FastTreeMP version 2.1.10 for estimating backbone topology. We run FastTreeMP with the following command:

```
#!/bin/bash
FastTreeMP -nosupport -gtr -gamma -nt -log tree.log
< aln_dna.fa > tree.nwk
```

For alignment free datasets such as Drosophila dataset, we computed backbone tree using FastME* (based on FastME version 2.1.6.1) which is available at https://github.com/balabanmetin/FastME-personal-copy. FastME* is run with the following command:

```
#!/bin/bash
fastme -i dist.mat -o tree.nwk -T 1
```

Note that we performed Jukes-Cantor correction on the distance matrix "dist.mat" before running FastME*.

B.3.4 Backbone tree branch length re-estimation

When multiple sequence alignment is available, we used FastME* to recompute backbone tree branch lengths for all datasets except RNAsim-VS dataset. We run FastME* with the following command:

```
#!/bin/bash
fastme -dJ -i aln_dna.phy -u RAxML_result.REF -o tree_me.nwk
```

For RNAsim-VS dataset, we used RAxML version 7.2.6 for re-estimating ML based branch lengths and used that tree for performing placements using pplacer. RAxML is run with the following command:

```
#!/bin/bash
raxmlHPC-PTHREADS -f e -t tree.nwk -m GTRGAMMA -s aln_dna.phy
-n REF -p 1984 -T 8
```

On the other hand, we used RAxML version 8.2.12 to compute backbone tree (branch lengths update only) and ML model parameters required for EPA-ng:

```
#!/bin/bash
raxmlHPC-PTHREADS -f e -t tree.nwk -m GTRGAMMA -s aln_dna.fa
-n REF8 -p 1984 -T 8
```

For all large alignments with 10,000 or more sequences in RNASim-VS, RAxML version 8.2.12 fail to run due to estimated gamma rate being either too small or too large. In those cases, we ran the following command to use GTRCAT model instead:

```
#!/bin/bash
raxmlHPC-PTHREADS -f e -t tree.nwk -m GTRCAT -s aln_dna.fa
-n REF8 -p 1984 -T 8
```

For the same dataset, we used FastTree again for re-estimating Minimum Evolution based branch lengths and used that tree for performing placements using APPLES. FastTree is run with the following command:

```
#!/bin/bash
FastTreeMP -nosupport -nt -nome -noml -log tree.log
-intree tree.nwk < aln_dna.fa > tree_me.nwk
```

B.3.5 Performing placement

We performed phylogenetic placement of a query using pplacer version 1.1.alpha19-0-g807f6f3 with the following command:

```
#!/bin/bash
pplacer -m GTR -s RAxML_info.REF -t backbone.nwk
-o query.jplace aln_dna.fa -j 1
```

We performed EPA-ng placements using version 0.3.5 with the following command:

```
#!/bin/bash
epa-ng --ref-msa aln_dna.fa --tree backbone.nwk
```

```
--query query.fa --outdir . --model RAxML_info.REF8 --redo -T 1
```

Except in our RNASim-VS, RNASim-QS and estimated alignments analyses, we used APPLES version 1.1.0 (can be found at https://github.com/balabanmetin/apples/releases) for placement. When alignment is not present, we performed the placement running the following command:

```
#!/bin/bash
python ~/apples/least_squares_placement.py -t backbone.tree
-d dist.mat -a FM -s MLSE -p > apples.nwk
```

When alignment is present, we used the following command instead:

```
#!/bin/bash
python ~/apples/least_squares_placement.py -t backbone.nwk
-a aln_dna.phy -a FM -s MLSE -p > apples.nwk
```

For RNASim-VS, RNASim-QS and estimated alignments analyses, we used APPLES version 1.2.0 and ran with the following command:

```
#!/bin/bash
python ~/apples/run_apples.py -t backbone.nwk -s aln_dna.fa
-q query.fa -T $numcores -o apples.jplace
```

where $*numcores* depends on the number of cores designated for the analysis.

B.3.6 Working with estimated backbone and query-to backbone alignments

We created a version of SEPP within which APPLES integrated. This version is available at https://github.com/balabanmetin/sepp. We performed placement on RNAsim-AE dataset with 10,000 sequences using SEPP+APPLES with the following commmand:

```
#!/bin/bash
```

```
python ~/sepp/run_sepp.py -t estimated_backbone.nwk
-a estimated_aln.fa -r RAxML_info.REF -f query.fa -pl apples
-x 28 -A 1000 -o apples
```

On the same dataset, we ran SEPP in default mode using the following commmand:

```
#!/bin/bash
python ~/sepp/run_sepp.py -t estimated_backbone.nwk
-a estimated_aln.fa -r RAxML_info.REF8 -f query.fa -pl pplacer
-x 28 -A 1000  -P 1000 -o pplacer
```

Appendix C

Supplementary materials for "Phylogenetic double placement of mixed samples"

C.1 Additional Methods

C.1.1 Proofs

Proof of Proposition 2. Recall that L is the Least Common ancestor of A, B, R. The lowest value of δ_{AR} is achieved when $\delta_{LR} = y$ and $\delta_{LA} = \delta_{LB} = 0$. On the other hand, the highest value of δ_{AR} is achieved when $\delta_{LR} = 0$ and $\delta_{LA} = \delta_{LB} = y$. The results are obtained by plugging in the values. □ □

Proof of Proposition 3. If $R = A$, $d_1 = 0$ and $d_2 = d_3$ must be true. Plugging in these values in SIMP-JAC formula, we get

$$\hat{\delta}_{MA} = 1 - \left(\frac{2}{3-(1-d_3)^k}\right)^{\frac{1}{k}}.$$

Let R_j be a genome in $\mathscr{R}$ which is different than A and B. Let f be a function such that

$$f(k) = (1-\delta_{AR_j})^k + (1-\delta_{BR_j})^k - (1-(\delta_{AR_j}+\delta_{BR_j}+\delta_{AB})/2)^k.;.$$

$\hat{\delta}_{MR_j} < \hat{\delta}_{MA}$ holds only if $1 < f(k)$. Consequently,

$$f(k) \leq (1-\delta_{AR_j})^k + (1-\delta_{BR_j})^k - (1-\delta_{AR_j}-\delta_{BR_j})^k$$

□

due to triangle inequality. For $k = 2$,

$$(1-\delta_{AR_j})^2 + (1-\delta_{BR_j})^2 - (1-\delta_{AR_j}-\delta_{BR_j})^2 = 1 - 2\delta_{AR_j}\delta_{BR_j}$$

While k increases, $f(k)$ is a monotonically non-increasing function, since its derivative with respect to k is non-positive. Therefore $f(k) < 1$ unless either $\delta_{AR_j} = 0$ or $\delta_{BR_j} = 0$ is true. □

C.1.2 Derivatives

Derivatives of objective functions.

Let T^* be the rooting of T at leaf 1 (Figure 5.8). For node $u \in V$, let $p(u)$ denote its parent, $l(u)$ its length, and $g(u)$ denote the set of leaves at or below u (i.e., those that have u on their path to the root), all with respect to T^*. Also, let x_1 and x_2 denote pendant and distal edge lengths of placement tree P and x_3 and x_4 denote those of placement tree Q. Eq. 5.5 can be re-written as:

$$\begin{aligned} F(x^A_*, x^B_*, x_1, x_2, x_3, x_4) = &\sum_{i \in g(u)} \left(x^A_i - d^P_{ui} - x_1 + x_2 - l(u)\right)^2 + \sum_{i \notin g(u)} \left(x^A_i - d^P_{p(u)i} - x_1 - x_2\right)^2 + \\ &\sum_{i \in g(v)} \left(x^B_i - d^Q_{vi} - x_3 + x_4 - l(v)\right)^2 + \sum_{i \notin g(v)} \left(x^B_i - d^Q_{p(v)i} - x_3 - x_4\right)^2 \end{aligned} \tag{C.1}$$

For fixed placement edges of u and v, $x^A_*, x^B_*, x_1, x_2, x_3$ and x_4 are variables, and the rest are constants. Therefore Jacobian matrix of function F is $\mathbf{J} = \begin{bmatrix} \frac{\partial F}{\partial x^A_*} & \frac{\partial F}{\partial x^B_*} & \frac{\partial F}{\partial x_1} & \frac{\partial F}{\partial x_2} & \frac{\partial F}{\partial x_3} & \frac{\partial F}{\partial x_4} \end{bmatrix}$

with following partial derivatives;

$$
\begin{aligned}
\frac{\partial F}{\partial x_i^A} &= \begin{cases} 2\big(x_i^A - d_{ui}^P - x_1 + x_2 - l(u)\big) & i \in g(u) \\ 2\big(x_i^A - d_{p(u)i}^P - x_1 - x_2\big) & i \notin g(u) \end{cases} \\
\frac{\partial F}{\partial x_i^B} &= \begin{cases} 2\big(x_i^B - d_{vi}^Q - x_3 + x_4 - l(u)\big) & i \in g(v) \\ 2\big(x_i^B - d_{p(v)i}^Q - x_3 - x_4\big) & i \notin g(v) \end{cases} \\
\frac{\partial F}{\partial x_1} &= \sum_{i \in g(u)} -2\big(x_i^A - d_{ui}^P - x_1 + x_2 - l(u)\big) + \sum_{i \notin g(u)} -2\big(x_i^A - d_{p(u)i}^P - x_1 - x_2\big) \\
\frac{\partial F}{\partial x_2} &= \sum_{i \in g(u)} 2\big(x_i^A - d_{ui}^P - x_1 + x_2 - l(u)\big) + \sum_{i \notin g(u)} -2\big(x_i^A - d_{p(u)i}^P - x_1 - x_2\big) \\
\frac{\partial F}{\partial x_3} &= \sum_{i \in g(v)} -2\big(x_i^B - d_{vi}^Q - x_3 + x_4 - l(v)\big) + \sum_{i \notin g(v)} -2\big(x_i^B - d_{p(v)i}^Q - x_3 - x_4\big) \\
\frac{\partial F}{\partial x_4} &= \sum_{i \in g(v)} 2\big(x_i^B - d_{vi}^Q - x_3 + x_4 - l(v)\big) + \sum_{i \notin g(v)} -2\big(x_i^B - d_{p(v)i}^Q - x_3 - x_4\big)
\end{aligned}
\tag{C.2}
$$

Hessian of function F is sparse $(2n+4) \times (2n+4)$ matrix, with only following non-zero entries:

$$
\begin{aligned}
\frac{\partial^2 F}{\partial^2 x_1} &= \frac{\partial^2 F}{\partial^2 x_2} = \frac{\partial^2 F}{\partial^2 x_3} = \frac{\partial^2 F}{\partial^2 x_4} = 2 * n \\
\frac{\partial^2 F}{\partial x_1 \partial x_2} &= -2 * |g(u)| + 2 * (n - |g(u)|) \\
\frac{\partial^2 F}{\partial x_3 \partial x_4} &= -2 * |g(v)| + 2 * (n - |g(v)|) \\
\frac{\partial^2 F}{\partial x_i^A \partial x_1} &= \begin{cases} 2 & i \in g(u) \\ -2 & i \notin g(u) \end{cases} \\
\frac{\partial^2 F}{\partial x_i^B \partial x_3} &= \begin{cases} 2 & i \in g(v) \\ -2 & i \notin g(v) \end{cases} \\
\frac{\partial^2 F}{\partial x_i^A \partial x_2} &= \frac{\partial^2 F}{\partial x_i^B \partial x_4} = -2
\end{aligned}
\tag{C.3}
$$

Derivatives of non-linear constraints:

In Eq. 4, we replace $\hat{\delta}_{Mi}$ with $h^{-1}(\hat{\delta}_{Mi})$ on the LHS, and on RHS, we replace all $h(x_i^A)$ terms with $h^{-1}(h(x_i^A)) = x_i^A$ (ditto for x_i^B) We substitute x_3 with its estimation denoted with $\hat{d}_3$. Subsequently, equation is rearranged in the following form;

$$G(x_i^A, x_i^B) = 2(1-x_i^A)^k + 2(1-x_i^B)^k - 2(1-\frac{x_i^A + x_i^B + \hat{d}_3}{2})^k\big) - (1-h^{-1}(\hat{\delta}_{Mi}))^k(3-(1-\hat{d}_3)^k) = 0 \quad \text{(C.4)}$$

Here i denotes the index of the backbone leaf. Therefore there is one constraint corresponding to one species in the backbone tree. In function G, x_i^A are (hidden) variables and other terms are constants. Note that no two constraints share a variable. Therefore we can safely write one Jacobian $\mathbf{J}$ and Hessian $\mathbf{H}$ of function G per constraint. The Jacobian has the following form $\mathbf{J} = \begin{bmatrix} \frac{\partial G}{\partial x_i^A} & \frac{\partial G}{\partial x_i^B} \end{bmatrix}$ with following partial derivatives;

$$\begin{aligned} \frac{\partial G}{\partial x_i^A} &= -2k(1-x_i^A)^{k-1} + k(1-\frac{x_i^A + x_i^B + \hat{d}_3}{2})^{k-1} \\ \frac{\partial G}{\partial x_i^B} &= -2k(1-x_i^B)^{k-1} + k(1-\frac{x_i^A + x_i^B + \hat{d}_3}{2})^{k-1} \end{aligned} \quad \text{(C.5)}$$

The Hessian matrix $\mathbf{H} = \begin{bmatrix} \frac{\partial^2 G}{\partial^2 x_i^A} & \frac{\partial^2 G}{\partial x_i^A \partial x_i^B} \\ \frac{\partial^2 G}{\partial x_i^A \partial x_i^B} & \frac{\partial^2 G}{\partial^2 x_i^B} \end{bmatrix}$ has the following entries

$$\begin{aligned} \frac{\partial^2 G}{\partial^2 x_i^A} &= 2k(k-1)(1-x_i^A)^{k-2} - \frac{k(k-1)}{2}(1-\frac{x_i^A + x_i^B + \hat{d}_3}{2})^{k-2} \\ \frac{\partial^2 G}{\partial^2 x_i^B} &= 2k(k-1)(1-x_i^B)^{k-1} - \frac{k(k-1)}{2}(1-\frac{x_i^A + x_i^B + \hat{d}_3}{2})^{k-1} \\ \frac{\partial^2 G}{\partial x_i^A \partial x_i^B} &= -\frac{k(k-1)}{2}(1-\frac{x_i^A + x_i^B + \hat{d}_3}{2})^{k-1} \end{aligned} \quad \text{(C.6)}$$

.

C.2 Supplementary Tables

Table C.1. List of SRRs and URLs for Drosophila species used in real data experiment.

Species	Run	URL
Drosophila bipectinata	SRR6425989	https://www.ncbi.nlm.nih.gov/sra/?term=SRR6425989
Drosophila erecta	SRR6425990	https://www.ncbi.nlm.nih.gov/sra/?term=SRR6425990
Drosophila ananassae	SRR6425991	https://www.ncbi.nlm.nih.gov/sra/?term=SRR6425991
Drosophila biarmipes	SRR6425992	https://www.ncbi.nlm.nih.gov/sra/?term=SRR6425992
Drosophila mauritiana	SRR6425993	https://www.ncbi.nlm.nih.gov/sra/?term=SRR6425993
Drosophila eugracilis	SRR6425995	https://www.ncbi.nlm.nih.gov/sra/?term=SRR6425995
Drosophila mojavensis	SRR6425997	https://www.ncbi.nlm.nih.gov/sra/?term=SRR6425997
Drosophila persimilis	SRR6425998	https://www.ncbi.nlm.nih.gov/sra/?term=SRR6425998
Drosophila simulans	SRR6425999	https://www.ncbi.nlm.nih.gov/sra/?term=SRR6425999
Drosophila virilis	SRR6426000	https://www.ncbi.nlm.nih.gov/sra/?term=SRR6426000
Drosophila pseudoobscura	SRR6426001	https://www.ncbi.nlm.nih.gov/sra/?term=SRR6426001
Drosophila sechellia	SRR6426002	https://www.ncbi.nlm.nih.gov/sra/?term=SRR6426002
Drosophila willistoni	SRR6426003	https://www.ncbi.nlm.nih.gov/sra/?term=SRR6426003
Drosophila yakuba	SRR6426004	https://www.ncbi.nlm.nih.gov/sra/?term=SRR6426004

Table C.2. GenBank accession numbers of yeast species/strains.

Species/Strains	GenBank accession
Candida bracarensis	GCA_001077315.1
Candida castellii	GCA_001046935.1
Candida glabrata	GCF_000002545.3
Candida nivariensis	GCA_001046915.1
Nakaseomyces bacillisporus	GCA_001046975.1
Nakaseomyces delphensis	GCA_001039675.1
Naumovozyma castellii	GCF_000237345.1
Saccharomyces arboricola	GCF_000292725.1
Saccharomyces bayanus	GCA_000167035.1
Saccharomyces boulardii	GCA_001413975.1
Saccharomyces cerevisiae	GCF_000146045.2
Saccharomyces eubayanus	GCF_001298625.1
Saccharomyces jurei	GCA_900290405.1
Saccharomyces kudriavzevii	GCA_900682665.1
Saccharomyces mikatae	GCA_000167055.1
Saccharomyces paradoxus	GCA_002079145.1
Saccharomyces pastorianus	GCA_001515485.2
Saccharomyces uvarum	GCA_002242645.1
VIN7	GCA_000326105.1
yHAB38	GCA_009666655.1
yHQL555	GCA_009666385.1

Appendix D

Supplementary materials for "Genome-wide alignment-free phylogenetic distance estimation under a no strand-bias model"

D.1 Supplementary Methods

Sarmashghi et al. [209] showed that genomic distance $\hat{d}$ can be estimated on genome skim data as follows:

$$\hat{d} = 1 - \Big(\frac{(\zeta_1 L_1 + \zeta_2 L_2)\mathrm{C}}{\eta_1 \eta_2 (L_1 + L_2)}\Big)^{\frac{1}{k}} \tag{D.1}$$

where ζ_i and η_i are functions of sequencing error rate, coverage, read length, and k (detailed definition of these variables are given in the original paper). We add the correction for background matches to this equation:

$$\hat{d} = 1 - \Big(\frac{(\zeta_1 L_1 + \zeta_2 L_2)(\mathrm{C} - \mathbb{E}[\tilde{\mathrm{C}}])}{\eta_1 \eta_2 (L_1 + L_2)}\Big)^{\frac{1}{k}} \tag{D.2}$$

We use this equation for not just $\hat{d}$ but also $\hat{d}_{AC}$,$\hat{d}_{AG}$,$\hat{d}_{AT}$, and $\hat{d}_{CG}$ values.

D.2 Supplementary Tables

Table D.1. Different model conditions (MC), used for simulating genome sequences

.

MC	1	2	3	4	5	6	7	8	9
α	1	1	1	1	1	1	1/4	1	1
δ	1	1/4	1	1	1/4	1	1/4	1/4	1
γ	16	4	4	32	8	8	1	1	1

Table D.2. Comparison of different methods to the ASTRAL tree on 10 sets of bacterial dataset. Best results are shown in bold.

	# taxa (branches)	Number of branch mismatches with ASTRAL			
		NSB-TK4	NSB-JC	Jellyfish-JC	Skmer-JC
Set 1	34(31)	**9**	11	15	13
Set 2	43(40)	11	11	**10**	**10**
Set 3	32(29)	**5**	**5**	8	8
Set 4	38(35)	**7**	**7**	8	8
Set 5	43(40)	**11**	12	12	12
Set 6	46(40)	**14**	**14**	**14**	**14**
Set 7	34(31)	**5**	**5**	6	6
Set 8	41(38)	16	16	**14**	15
Set 9	39(36)	**7**	**7**	**7**	**7**
Set 10	86(83)	**35**	37	39	36
Sum	433 (403)	**120**	125	133	129

Table D.3. Distance error (deviation from additivity) for NSB-TK4 and Jellyfish-JC on bacterial dataset.

	# taxa (branches)	TotalFM (NSB-TK4)	totalFM (Jellyfish-JC)	TotalOLS (NSB-TK4)	TotalOLS (Jellyfish-JC)
Set 1	34(31)	**0.8209**	1.29	**0.032**	0.0436
Set 2	43(40)	**0.8032**	0.8686	**0.0114**	0.0121
Set 3	32(29)	**0.2758**	0.3534	**0.0132**	0.0139
Set 4	38(35)	**1.7227**	1.926	0.1134	**0.094**
Set 5	43(40)	**2.3761**	2.5785	0.1688	**0.1367**
Set 6	46(40)	**2.7291**	2.9287	0.2003	**0.1633**
Set 7	34(31)	**0.6379**	0.8103	**0.0309**	0.0323
Set 8	41(38)	**1.8973**	2.0946	0.1186	**0.0893**
Set 9	39(36)	**0.9773**	1.2537	**0.0478**	0.0494
Set 10	86(83)	**42.0776**	55.9276	**3.4154**	4.0831
Sum	433 (403)	**54.3179**	70.0314	**4.1518**	4.7177

Table D.4. GenBank accession numbers of yeast species/strains.

Species/Strains	GenBank accession
Saccharomyces arboricola	GCF_000292725.1
Saccharomyces cerevisiae	GCF_000146045.2
Saccharomyces eubayanus	GCF_001298625.1
Saccharomyces jurei	GCA_900290405.1
Saccharomyces kudriavzevii	GCA_900682665.1
Saccharomyces mikatae	GCA_000167055.1
Saccharomyces paradoxus	GCA_002079145.1
Saccharomyces uvarum	GCA_002242645.1

Table D.5. Topological and distance error for NSB-TK4, Jellyfish-JC, and Skmer-JC on yeast dataset.

	Coverage	RF (NSB-TK4) (Jellyfish-JC)	RF (Skmer-JC)	TotalFM (NSB-TK4)	TotalFM (Jellyfish-JC)	TotalFM (Skmer-JC)
Genome skim	1×	0	1	**0.0028**	0.0050	0.0063
Genome skim	2×	0	0	**0.0024**	0.0039	0.0072
Genome skim	4×	0	1	**0.0023**	0.0039	0.0083
Genome skim	8×	0	1	**0.0019**	0.0034	0.0084
Assembly	n.a.	0	1	**0.0020**	0.0034	0.0082

Bibliography

[1] O. ADEBALI, A. BIRCAN, D. ÇIRCI, B. IŞLEK, Z. KILINÇ, B. SELÇUK, and B. TURHAN. Phylogenetic analysis of SARS-CoV-2 genomes in Turkey. *TURKISH JOURNAL OF BIOLOGY*, 44(3):146–156, 6 2020. ISSN 1303-6092. doi: 10.3906/biy-2005-35. URL http://journals.tubitak.gov.tr/biology/issues/biy-20-44-si-1/biy-44-si-1-3-2005-35.pdf.

[2] J. L. Aldous, S. K. Pond, A. Poon, S. Jain, H. Qin, J. S. Kahn, M. Kitahata, B. Rodriguez, A. M. Dennis, S. L. Boswell, R. Haubrich, and D. M. Smith. Characterizing HIV transmission networks across the United States. *Clinical infectious diseases : an official publication of the Infectious Diseases Society of America*, 55(8):1135–43, 2012. ISSN 1537-6591. doi: 10.1093/cid/cis612. URL http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3529609&tool=pmcentrez&rendertype=abstract.

[3] E. S. Allman, J. A. Rhodes, and S. Sullivant. Statistically Consistent k -mer Methods for Phylogenetic Tree Reconstruction. *Journal of Computational Biology*, 24(2):153–171, 2 2017. ISSN 1066-5277. doi: 10.1089/cmb.2015.0216. URL http://online.liebertpub.com/doi/10.1089/cmb.2015.0216.

[4] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215(3):403–410, 10 1990. ISSN 00222836. doi: 10.1016/S0022-2836(05)80360-2. URL http://www.ncbi.nlm.nih.gov/pubmed/2231712.

[5] A. Amir, D. McDonald, J. A. Navas-Molina, E. Kopylova, J. T. Morton, Z. Zech Xu, E. P. Kightley, L. R. Thompson, E. R. Hyde, A. Gonzalez, and R. Knight. Deblur Rapidly Resolves Single-Nucleotide Community Sequence Patterns. *mSystems*, 2(2), 2017. doi: 10.1128/mSystems.00191-16. URL https://msystems.asm.org/content/2/2/e00191-16.

[6] M. Anisimova, M. Gil, J.-F. Dufayard, C. Dessimoz, and O. Gascuel. Survey of Branch Support Methods Demonstrates Accuracy, Power, and Robustness of Fast Likelihood-based Approximation Schemes. *Systematic Biology*, 60(5):685–699, 10 2011. ISSN 1076-836X. doi: 10.1093/sysbio/syr041. URL https://academic.oup.com/sysbio/article/60/5/685/1644562.

[7] M. Ashburner, C. A. Ball, J. A. Blake, D. Botstein, H. Butler, J. M. Cherry, A. P.

Davis, K. Dolinski, S. S. Dwight, J. T. Eppig, M. A. Harris, D. P. Hill, L. Issel-Tarver, A. Kasarskis, S. Lewis, J. C. Matese, J. E. Richardson, M. Ringwald, G. M. Rubin, and G. Sherlock. Gene ontology: Tool for the unification of biology, 2000. ISSN 10614036.

[8] F. Asnicar, A. M. Thomas, F. Beghini, C. Mengoni, S. Manara, P. Manghi, Q. Zhu, M. Bolzan, F. Cumbo, U. May, J. G. Sanders, M. Zolfo, E. Kopylova, E. Pasolli, R. Knight, S. Mirarab, C. Huttenhower, and N. Segata. Precise phylogenetic analysis of microbial isolates and genomes from metagenomes using PhyloPhlAn 3.0. *Nature Communications*, 11(1):2500, 12 2020. ISSN 2041-1723. doi: 10.1038/s41467-020-16366-7. URL http://dx.doi.org/10.1038/s41467-020-16366-7http://www.nature.com/articles/s41467-020-16366-7.

[9] D. N. Baker and B. Langmead. Dashing: Fast and accurate genomic distances with HyperLogLog. *Genome Biology*, 2019. ISSN 1474760X. doi: 10.1186/s13059-019-1875-0.

[10] M. Balaban and S. Mirarab. Phylogenetic double placement of mixed samples. *Bioinformatics*, 36(Supplement_1):i335–i343, 7 2020. ISSN 1367-4803. doi: 10.1093/bioinformatics/btaa489. URL https://academic.oup.com/bioinformatics/article/36/Supplement_1/i335/5870522.

[11] M. Balaban, S. Sarmashghi, and S. Mirarab. APPLES: Fast Distance-based Phylogenetic Placement. *bioRxiv*, 69(3):475566, 1 2018. ISSN 1063-5157. doi: 10.1101/475566. URL http://biorxiv.org/content/early/2018/11/23/475566.abstract.

[12] M. Balaban, N. Moshiri, U. Mai, X. Jia, and S. Mirarab. TreeCluster: Clustering biological sequences using phylogenetic trees. *PLOS ONE*, 14(8):e0221068, 8 2019. ISSN 1932-6203. doi: 10.1371/journal.pone.0221068. URL https://doi.org/10.1371/journal.pone.0221068http://dx.plos.org/10.1371/journal.pone.0221068.

[13] M. Balaban, S. Sarmashghi, and S. Mirarab. APPLES: Scalable Distance-Based Phylogenetic Placement with or without Alignments. *Systematic Biology*, 69(3):566–578, 5 2020. ISSN 1063-5157. doi: 10.1093/sysbio/syz063. URL https://academic.oup.com/sysbio/article/69/3/566/5572672.

[14] M. Balaban, S. Sarmashghi, and S. Mirarab. APPLES: Scalable Distance-Based Phylogenetic Placement with or without Alignments. *Systematic Biology*, 69(3):566–578, 2020. ISSN 1076836X. doi: 10.1093/sysbio/syz063.

[15] M. Balaban, Y. Jiang, D. Roush, Q. Zhu, and S. Mirarab. Fast and accurate distance-based phylogenetic placement using divide and conquer. *Molecular Ecology Resources*, 22 (3):1213–1227, apr 2022. ISSN 1755-098X. doi: 10.1111/1755-0998.13527. URL https://onlinelibrary.wiley.com/doi/10.1111/1755-0998.13527.

[16] A. Bankevich, S. Nurk, D. Antipov, A. A. Gurevich, M. Dvorkin, A. S. Kulikov, V. M.

Lesin, S. I. Nikolenko, S. Pham, A. D. Prjibelski, A. V. Pyshkin, A. V. Sirotkin, N. Vyahhi, G. Tesler, M. A. Alekseyev, and P. A. Pevzner. SPAdes: A new genome assembly algorithm and its applications to single-cell sequencing. *Journal of Computational Biology*, 2012. ISSN 10665277. doi: 10.1089/cmb.2012.0021.

[17] P. Barbera, A. M. Kozlov, L. Czech, B. Morel, D. Darriba, T. Flouri, and A. Stamatakis. EPA-ng: Massively Parallel Evolutionary Placement of Genetic Sequences. *Systematic Biology*, 68(2):365–369, mar 2019. ISSN 1063-5157. doi: 10.1093/sysbio/syy054. URL https://academic.oup.com/sysbio/article/68/2/365/5079844.

[18] P. Barbera, A. M. Kozlov, L. Czech, B. Morel, D. Darriba, T. Flouri, and A. Stamatakis. EPA-ng: Massively Parallel Evolutionary Placement of Genetic Sequences. *Systematic Biology*, 68(2):365–369, 3 2019. ISSN 1063-5157. doi: 10.1093/sysbio/syy054. URL https://academic.oup.com/sysbio/article/68/2/365/5079844.

[19] G. Benoit, P. Peterlongo, M. Mariadassou, E. Drezen, S. Schbath, D. Lavenier, and C. Lemaitre. Multiple comparative metagenomics using multiset k -mer counting. *PeerJ Computer Science*, 2:e94, 11 2016. ISSN 2376-5992. doi: 10.7717/peerj-cs.94. URL https://peerj.com/articles/cs-94.

[20] S. a. Berger and A. Stamatakis. Aligning short Reads to Reference Alignments and Trees. *Bioinformatics*, 27(15):2068–75, 6 2011. ISSN 1367-4811. doi: 10.1093/bioinformatics/btr320. URL http://www.ncbi.nlm.nih.gov/pubmed/21636595.

[21] S. A. Berger, D. Krompass, and A. Stamatakis. Performance, Accuracy, and Web Server for Evolutionary Placement of Short Sequence Reads under Maximum Likelihood. *Systematic Biology*, 60(3):291–302, 5 2011. ISSN 1076-836X. doi: 10.1093/sysbio/syr010. URL http://sysbio.oxfordjournals.org/cgi/content/abstract/60/3/291http://sysbio.oxfordjournals.org/content/60/3/291.abstracthttp://sysbio.oxfordjournals.org/content/60/3/291.full.pdfhttp://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3078422&tool=pmc.

[22] W. A. Beyer, M. L. Stein, T. F. Smith, and S. M. Ulam. A molecular sequence metric and evolutionary trees. *Mathematical Biosciences*, 19(1-2):9–25, feb 1974. ISSN 00255564. doi: 10.1016/0025-5564(74)90028-5. URL https://linkinghub.elsevier.com/retrieve/pii/0025556474900285.

[23] D. Bezemer, A. Cori, O. Ratmann, A. van Sighem, H. S. Hermanides, B. E. Dutilh, L. Gras, N. Rodrigues Faria, R. van den Hengel, A. J. Duits, P. Reiss, F. de Wolf, and C. Fraser. Dispersion of the HIV-1 Epidemic in Men Who Have Sex with Men in the Netherlands: A Combined Mathematical Model and Phylogenetic Analysis. *PLoS Medicine*, 12(11):e1001898, 2015. doi: 10.1371/journal.pmed.1001898. URL http://journals.plos.org/plosmedicine/article?id=10.1371/journal.pmed.1001898.

[24] A. Bhattacharjee and M. S. Bayzid. Machine learning based imputation techniques for

estimating phylogenetic trees from incomplete distance matrices. *BMC Genomics*, 21 (1):497, 12 2020. ISSN 1471-2164. doi: 10.1186/s12864-020-06892-5. URL https://bmcgenomics.biomedcentral.com/articles/10.1186/s12864-020-06892-5.

[25] M. Bogusz and S. Whelan. Phylogenetic Tree Estimation With and Without Alignment: New Distance Methods and Benchmarking. *Systematic Biology*, 66(2):218–231, 2016. ISSN 1063-5157. doi: 10.1093/sysbio/syw074. URL https://doi.org/10.1093/sysbio/syw074.

[26] K. Bohmann, S. Mirarab, V. Bafna, and M. T. P. Gilbert. Beyond DNA barcoding: The unrealized potential of genome skim data in sample identification. *Molecular Ecology*, 29(14):2521–2534, 7 2020. ISSN 0962-1083. doi: 10.1111/mec.15507. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/mec.15507.

[27] B. M. Boyd, J. M. Allen, N. Nguyen, A. D. Sweet, T. Warnow, M. D. Shapiro, S. M. Villa, S. E. Bush, D. H. Clayton, and K. P. Johnson. Phylogenomics using Target-restricted Assembly Resolves Intra-generic Relationships of Parasitic Lice (Phthiraptera: Columbicola). *Systematic Biology*, page syx027, jan 2017. ISSN 1063-5157. doi: 10.1093/sysbio/syx027. URL https://academic.oup.com/sysbio/article-lookup/doi/10.1093/sysbio/syx027.

[28] A. Brady and S. L. Salzberg. Phymm and PhymmBL: metagenomic phylogenetic classification with interpolated Markov models. *Nature methods*, 6(9):673–6, 9 2009. ISSN 1548-7105. doi: 10.1038/nmeth.1358. URL http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=2762791&tool=pmcentrez&rendertype=abstract.

[29] D. G. Brown and J. Truszkowski. LSHPlace: fast phylogenetic placement using locality-sensitive hashing. *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, pages 310–9, 11 2013. ISSN 2335-6936. doi: 10.1142/9789814447973{_}0031. URL http://www.ncbi.nlm.nih.gov/pubmed/23424136.

[30] A. Bush, R. Sollmann, A. Wilting, K. Bohmann, B. Cole, H. Balzter, C. Martius, A. Zlinszky, S. Calvignac-Spencer, C. A. Cobbold, T. P. Dawson, B. C. Emerson, S. Ferrier, M. T. P. Gilbert, M. Herold, L. Jones, F. H. Leendertz, L. Matthews, J. D. A. Millington, J. R. Olson, O. Ovaskainen, D. Raffaelli, R. Reeve, M.-O. Rödel, T. W. Rodgers, S. Snape, I. Visseren-Hamakers, A. P. Vogler, P. C. L. White, M. J. Wooster, and D. W. Yu. Connecting Earth observation to high-throughput biodiversity data. *Nature Ecology & Evolution*, 1(7):41559–017, 7 2017. ISSN 2397-334X. doi: 10.1038/s41559-017-0176. URL https://www.nature.com/articles/s41559-017-0176%5Cninternal-pdf://0.0.9.188/s41559-017-0176.html%5Cnhttp://www.nature.com/articles/s41559-017-0176%5Cnhttp://files/2491/Bushetal.-2017-ConnectingEarthobservationtohigh-throughputbi.pdf%5Cnhttp://f.

[31] B. Bushnell. Bbtools software package. *URL http://sourceforge. net/projects/bbmap*,

2014.

[32] Y. Cai and Y. Sun. ESPRIT-Tree: hierarchical clustering analysis of millions of 16S rRNA pyrosequences in quasilinear computational time. *Nucleic Acids Research*, 39(14): e95–e95, aug 2011. ISSN 0305-1048. doi: 10.1093/nar/gkr349. URL https://academic.oup.com/nar/article-lookup/doi/10.1093/nar/gkr349.

[33] B. J. Callahan, P. J. McMurdie, M. J. Rosen, A. W. Han, A. J. A. Johnson, and S. P. Holmes. DADA2: High-resolution sample inference from Illumina amplicon data. *Nature Methods*, 13:581, 5 2016. URL https://doi.org/10.1038/nmeth.3869http://10.0.4.14/nmeth.3869https://www.nature.com/articles/nmeth.3869#supplementary-information.

[34] S. Capella-Gutiérrez, J. M. Silla-Martínez, and T. Gabaldón. trimAl: a tool for automated alignment trimming in large-scale phylogenetic analyses. *Bioinformatics*, 25(15):1972–1973, 8 2009. ISSN 1367-4803. doi: 10.1093/bioinformatics/btp348. URL https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btp348.

[35] L. L. Cavalli-Sforza and A. W. Edwards. Phylogenetic analysis. Models and estimation procedures. *American journal of human genetics*, 19(3 Pt 1):233–57, 5 1967. ISSN 0002-9297. URL http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=1706274&tool=pmcentrez&rendertype=abstracthttp://www.ncbi.nlm.nih.gov/pubmed/6026583http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC1706274.

[36] E. Chargaff. Structure and function of nucleic acids as cell constituents. *Federation proceedings*, 10(3):654–9, 9 1951. ISSN 0014-9446. URL http://www.ncbi.nlm.nih.gov/pubmed/14887699.

[37] W. Chen, C. K. Zhang, Y. Cheng, S. Zhang, and H. Zhao. A Comparison of Methods for Clustering 16S rRNA Sequences into OTUs. *PLoS ONE*, 8(8):e70837, aug 2013. ISSN 1932-6203. doi: 10.1371/journal.pone.0070837. URL https://dx.plos.org/10.1371/journal.pone.0070837.

[38] L. J. Clarke, J. Soubrier, L. S. Weyrich, and A. Cooper. Environmental metabarcodes for insects: In silico PCR reveals potential for taxonomic bias. *Molecular Ecology Resources*, 14(6):1160–1170, 2014. ISSN 17550998. doi: 10.1111/1755-0998.12265.

[39] E. Coissac, P. M. Hollingsworth, S. Lavergne, and P. Taberlet. From barcodes to genomes: extending the concept of DNA barcoding. *Molecular Ecology*, 25(7):1423–1428, 4 2016. ISSN 09621083. doi: 10.1111/mec.13549. URL http://doi.wiley.com/10.1111/mec.13549.

[40] A. R. Conn, N. I. M. Gould, and P. L. Toint. *Trust Region Methods*. Society for Industrial and Applied Mathematics, 1 2000. ISBN 978-0-89871-460-9. doi: 10.1137/1.9780898719857. URL http://epubs.siam.org/doi/book/10.1137/1.9780898719857.

[41] A. Criscuolo. A fast alignment-free bioinformatics procedure to infer accurate distance-based phylogenetic trees from genome assemblies. *Research Ideas and Outcomes*, 5, 6 2019. ISSN 2367-7163. doi: 10.3897/rio.5.e36178. URL https://riojournal.com/article/36178/.

[42] Q. Dai, Y. Yang, and T. Wang. Markov model plus k-word distributions: a synergy that produces novel statistical measures for sequence comparison. *Bioinformatics*, 24(20): 2296–2302, 10 2008. ISSN 1460-2059. doi: 10.1093/bioinformatics/btn436. URL https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btn436.

[43] A. E. Darling, G. Jospin, E. Lowe, F. A. Matsen, H. M. Bik, and J. A. Eisen. PhyloSift: phylogenetic analysis of genomes and metagenomes. *PeerJ*, 2:e243, 1 2014. ISSN 2167-8359. doi: 10.7717/peerj.243. URL https://peerj.com/articles/243.

[44] A. E. Darling, G. Jospin, E. Lowe, F. A. Matsen, H. M. Bik, and J. A. Eisen. PhyloSift: Phylogenetic analysis of genomes and metagenomes. *PeerJ*, 2014. ISSN 21678359. doi: 10.7717/peerj.243.

[45] D. Darriba, D. Posada, A. M. Kozlov, A. Stamatakis, B. Morel, and T. Flouri. ModelTest-NG: A New and Scalable Tool for the Selection of DNA and Protein Evolutionary Models. *Molecular Biology and Evolution*, 2020. ISSN 15371719. doi: 10.1093/molbev/msz189.

[46] C. Daskalakis and S. Roch. Alignment-free phylogenetic reconstruction: Sample complexity via a branching process analysis. *Annals of Applied Probability*, 23(2):693–721, 2013. ISSN 10505164. doi: 10.1214/12-AAP852.

[47] W. H. E. Day and D. Sankoff. Computational complexity of inferring phylogenies from chromosome inversion data. *Journal of Theoretical Biology*, 124(2):213–218, 1987. ISSN 10958541. doi: 10.1016/S0022-5193(87)80263-1.

[48] T. Z. DeSantis, P. Hugenholtz, N. Larsen, M. Rojas, E. L. Brodie, K. Keller, T. Huber, D. Dalevi, P. Hu, and G. L. Andersen. Greengenes, a Chimera-Checked 16S rRNA Gene Database and Workbench Compatible with ARB. *Appl. Environ. Microbiol.*, 72(7):5069–5072, 7 2006. ISSN 0099-2240. doi: 10.1128/AEM.03006-05. URL http://aem.asm.org/cgi/content/abstract/72/7/5069http://www.ncbi.nlm.nih.gov/pmc/articles/PMC1489311/.

[49] R. Desper and O. Gascuel. Fast and Accurate Phylogeny Reconstruction Algorithms Based on the Minimum-Evolution Principle. *Journal of Computational Biology*, 9(5):687–705, 10 2002. ISSN 1066-5277. doi: 10.1089/106652702761034136. URL http://www.liebertonline.com/doi/abs/10.1089/106652702761034136http://www.liebertpub.com/doi/10.1089/106652702761034136.

[50] R. Desper and O. Gascuel. Fast and accurate phylogeny minimum-evolution principle. *J Comput Biol*, 9(5):687–705, 2002. ISSN 1066-5277.

[51] R. Desper and O. Gascuel. Theoretical Foundation of the Balanced Minimum Evolution Method of Phylogenetic Inference and Its Relationship to Weighted Least-Squares Tree Fitting. *Molecular Biology and Evolution*, 21(3):587–598, 2004. ISSN 07374038. doi: 10.1093/molbev/msh049.

[52] S. Dodsworth. Genome skimming for next-generation biodiversity analysis. *Trends in Plant Science*, 20(9):525–527, 9 2015. ISSN 13601385. doi: 10.1016/j.tplants.2015.06.012. URL https://linkinghub.elsevier.com/retrieve/pii/S1360138515001764.

[53] P. Donkersley, G. Rhodes, R. W. Pickup, K. C. Jones, E. F. Power, G. A. Wright, and K. Wilson. Nutritional composition of honey bee food stores vary with floral composition. *Oecologia*, 185(4):749–761, Dec 2017. ISSN 0029-8549. doi: 10.1007/s00442-017-3968-3. URL http://link.springer.com/10.1007/s00442-017-3968-3.

[54] B. Dunn and G. Sherlock. Reconstruction of the genome origins and evolution of the hybrid lager yeast Saccharomyces pastorianus. *Genome Research*, 18(10):1610–1623, 8 2008. ISSN 1088-9051. doi: 10.1101/gr.076075.108. URL http://www.genome.org/cgi/doi/10.1101/gr.076075.108.

[55] D. Earl, N. Nguyen, G. Hickey, R. S. Harris, S. Fitzgerald, K. Beal, I. Seledtsov, V. Molodtsov, B. J. Raney, H. Clawson, J. Kim, C. Kemena, J.-M. Chang, I. Erb, A. Poliakov, M. Hou, J. Herrero, W. J. Kent, V. Solovyev, A. E. Darling, J. Ma, C. Notredame, M. Brudno, I. Dubchak, D. Haussler, and B. Paten. Alignathon: a competitive assessment of whole-genome alignment methods. *Genome Research*, 24(12):2077–2089, 12 2014. ISSN 1088-9051. doi: 10.1101/gr.174920.114. URL http://www.biorxiv.org/content/biorxiv/early/2014/03/10/003285.full.pdfhttp://genome.cshlp.org/lookup/doi/10.1101/gr.174920.114.

[56] S. R. Eddy. Profile hidden Markov models. *Bioinformatics*, pages 755–763, 1998. URL http://bioinformatics.oxfordjournals.org/content/14/9/755.short.

[57] S. R. Eddy. A new generation of homology search tools based on probabilistic inference. *Genome Inform*, 23(1):205–211, 10 2009. ISSN 0919-9454. URL http://www.ncbi.nlm.nih.gov/pubmed/20180275.

[58] R. C. Edgar. Search and clustering orders of magnitude faster than BLAST. *Bioinformatics*, 26(19):2460–2461, 10 2010. ISSN 1367-4803. URL http://dx.doi.org/10.1093/bioinformatics/btq461.

[59] R. C. Edgar. UNOISE2: improved error-correction for Illumina 16S and ITS amplicon sequencing. *bioRxiv*, 2016. doi: 10.1101/081257. URL https://www.biorxiv.org/content/early/2016/10/15/081257.

[60] A. J. Enright, S. Van Dongen, and C. A. Ouzounis. An efficient algorithm for large-

scale detection of protein families. *Nucleic acids research*, 30(7):1575–84, 2002. ISSN 1362-4962. URL http://www.ncbi.nlm.nih.gov/pubmed/11917018{%}0Ahttp://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC101833.

[61] P. Erdos, M. Steel, L. Szekely, and T. Warnow. A few logs suffice to build (almost) all trees: Part II. *Theoretical Computer Science*, 221(1-2):77–118, 1999. ISSN 03043975. doi: 10.1016/S0304-3975(99)00028-6.

[62] S. H. Eshleman, S. E. Hudelson, A. D. Redd, L. Wang, R. Debes, Y. Q. Chen, C. A. Martens, S. M. Ricklefs, E. J. Selig, S. F. Porcella, S. Munshaw, S. C. Ray, E. Piwowar-Manning, M. McCauley, M. C. Hosseinipour, J. Kumwenda, J. G. Hakim, S. Chariyalertsak, G. De Bruyn, B. Grinsztejn, N. Kumarasamy, J. Makhema, K. H. Mayer, J. Pilotto, B. R. Santos, T. C. Quinn, M. S. Cohen, and J. P. Hughes. Analysis of genetic linkage of HIV from couples enrolled in the HIV prevention trials network 052 trial. *Journal of Infectious Diseases*, 204(12):1918–1926, 2011. doi: 10.1093/infdis/jir651. URL https://academic.oup.com/jid/article/204/12/1918/1021440.

[63] H. Fan, A. R. Ives, Y. Surget-Groba, and C. H. Cannon. An assembly and alignment-free method of phylogeny reconstruction from next-generation sequencing data. *BMC Genomics*, 16(1):522, 12 2015. ISSN 1471-2164. doi: 10.1186/s12864-015-1647-5. URL http://www.biomedcentral.com/1471-2164/16/522.

[64] J. S. Farris. Estimating Phylogenetic Trees from Distance Matrices. *The American Naturalist*, 106(951):645–668, sep 1972. ISSN 0003-0147. doi: 10.1086/282802. URL https://www.journals.uchicago.edu/doi/10.1086/282802.

[65] J. Felsenstein. Evolutionary trees from DNA sequences: A maximum likelihood approach. *Journal of Molecular Evolution*, 17(6):368–376, 11 1981. ISSN 0022-2844. doi: 10.1007/BF01734359. URL http://link.springer.com/10.1007/BF01734359.

[66] J. Felsenstein. Inferring phytogenies. *Sunderland*, 2003.

[67] K. Findley, J. Oh, J. Yang, S. Conlan, C. Deming, J. A. Meyer, D. Schoenfeld, E. Nomicos, M. Park, H. H. Kong, and J. A. Segre. Topographic diversity of fungal and bacterial communities in human skin. *Nature*, 498(7454):367–370, 6 2013. ISSN 0028-0836. doi: 10.1038/nature12171. URL http://www.nature.com/articles/nature12171.

[68] W. M. Fitch and E. Margoliash. Construction of Phylogenetic Trees. *Science*, 155 (3760):279–284, jan 1967. ISSN 0036-8075. doi: 10.1126/science.155.3760.279. URL http://www.sciencemag.org/cgi/doi/10.1126/science.155.3760.279.

[69] W. Fletcher and Z. Yang. INDELible: A flexible simulator of biological sequence evolution. *Molecular Biology and Evolution*, 26(8):1879–1888, 2009. ISSN 07374038. doi: 10.1093/molbev/msp098.

[70] W. Fletcher and Z. Yang. INDELible: A Flexible Simulator of Biological Sequence Evolution. *Molecular Biology and Evolution*, 26(8):1879–1888, 8 2009. ISSN 0737-4038. doi: 10.1093/molbev/msp098. URL https://academic.oup.com/mbe/article-lookup/doi/10.1093/molbev/msp098.

[71] D. Forsdyke. Relative roles of primary sequence and (G + C)% in determining the hierarchy of frequencies of complementary trinucleotide pairs in DNAs of different species. *Journal of Molecular Evolution*, 41(5), 11 1995. ISSN 0022-2844. doi: 10.1007/BF00175815. URL http://link.springer.com/10.1007/BF00175815.

[72] D. R. Forsdyke. Success of alignment-free oligonucleotide (k-mer) analysis confirms relative importance of genomes not genes in speciation and phylogeny. *Biological Journal of the Linnean Society*, 128(2):239–250, 7 2019. ISSN 0024-4066. doi: 10.1093/biolinnean/blz096. URL https://academic.oup.com/biolinnean/advance-article/doi/10.1093/biolinnean/blz096/5533233.

[73] D. R. Forsdyke. Neutralism versus selectionism: Chargaff's second parity rule, revisited. *Genetica*, 149(2):81–88, 4 2021. ISSN 0016-6707. doi: 10.1007/s10709-021-00119-5. URL https://link.springer.com/10.1007/s10709-021-00119-5.

[74] N. Galtier and J. Lobry. Relationships Between Genomic G+C Content, RNA Secondary Structures, and Optimal Growth Temperature in Prokaryotes. *Journal of Molecular Evolution*, 44(6):632–636, 6 1997. ISSN 0022-2844. doi: 10.1007/PL00006186. URL http://link.springer.com/10.1007/PL00006186.

[75] S. R. Gill, M. Pop, R. T. DeBoy, P. B. Eckburg, P. J. Turnbaugh, B. S. Samuel, J. I. Gordon, D. A. Relman, C. M. Fraser-Liggett, and K. E. Nelson. Metagenomic Analysis of the Human Distal Gut Microbiome. *Science*, 312(5778):1355–1359, 6 2006. ISSN 1095-9203. doi: 10.1126/science.1124234. URL http://www.sciencemag.org/content/312/5778/1355.abstracthttp://www.sciencemag.org/content/312/5778/1355.full.pdfhttp://www.sciencemag.org/cgi/content/abstract/312/5778/1355http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3027896/http://www.ncbi.nlm.nih.gov.

[76] A. Gonzalez, J. A. Navas-Molina, T. Kosciolek, D. McDonald, Y. Vázquez-Baeza, G. Ackermann, J. DeReus, S. Janssen, A. D. Swafford, S. B. Orchanian, J. G. Sanders, J. Shorenstein, H. Holste, S. Petrus, A. Robbins-Pianka, C. J. Brislawn, M. Wang, J. R. Rideout, E. Bolyen, M. Dillon, J. G. Caporaso, P. C. Dorrestein, and R. Knight. Qiita: rapid, web-enabled microbiome meta-analysis. *Nature Methods*, 2018. ISSN 15487105. doi: 10.1038/s41592-018-0141-9.

[77] A. Gonzalez, J. A. Navas-Molina, T. Kosciolek, D. McDonald, Y. Vázquez-Baeza, G. Ackermann, J. DeReus, S. Janssen, A. D. Swafford, S. B. Orchanian, J. G. Sanders, J. Shorenstein, H. Holste, S. Petrus, A. Robbins-Pianka, C. J. Brislawn, M. Wang, J. R. Rideout, E. Bolyen, M. Dillon, J. G. Caporaso, P. C. Dorrestein, and R. Knight. Qi-

ita: rapid, web-enabled microbiome meta-analysis. *Nature Methods*, 15(10):796–798, 2018. ISSN 1548-7105. doi: 10.1038/s41592-018-0141-9. URL https://doi.org/10.1038/s41592-018-0141-9.

[78] J. K. Goodrich, S. C. Di Rienzi, A. C. Poole, O. Koren, W. A. Walters, J. G. Caporaso, R. Knight, and R. E. Ley. Conducting a microbiome study. *Cell*, 158(2):250–262, 2014.

[79] H. Gourlé, O. Karlsson-Lindsjö, J. Hayer, and E. Bongcam-Rudloff. Simulating Illumina metagenomic data with InSilicoSeq. *Bioinformatics*, 2019. ISSN 14602059. doi: 10.1093/bioinformatics/bty630.

[80] S. Guo, L.-S. Wang, and J. Kim. Large-scale simulation of RNA macroevolution by an energy-dependent fitness model. *arXiv*, 0912.2326, 12 2009. URL http://arxiv.org/abs/0912.2326.

[81] D. H. Haft, M. DiCuccio, A. Badretdin, V. Brover, V. Chetvernin, K. O'Neill, W. Li, F. Chitsaz, M. K. Derbyshire, N. R. Gonzales, M. Gwadz, F. Lu, G. H. Marchler, J. S. Song, N. Thanki, R. A. Yamashita, C. Zheng, F. Thibaud-Nissen, L. Y. Geer, A. Marchler-Bauer, and K. D. Pruitt. RefSeq: An update on prokaryotic genome annotation and curation. *Nucleic Acids Research*, 2018. ISSN 13624962. doi: 10.1093/nar/gkx1068.

[82] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, 9 2020. ISSN 0028-0836. doi: 10.1038/s41586-020-2649-2. URL http://www.nature.com/articles/s41586-020-2649-2.

[83] M. Hasegawa, H. Kishino, and T. a. Yano. Dating of the human-ape splitting by a molecular clock of mitochondrial DNA. *Journal of Molecular Evolution*, 1985. ISSN 00222844. doi: 10.1007/BF02101694.

[84] B. Haubold. Alignment-free phylogenetics and population genetics. *Briefings in Bioinformatics*, 15(3):407–418, 5 2014. ISSN 1467-5463. doi: 10.1093/bib/bbt083. URL https://academic.oup.com/bib/article-lookup/doi/10.1093/bib/bbt083.

[85] P. D. N. Hebert, A. Cywinska, S. L. Ball, and J. R. deWaard. Biological identifications through DNA barcodes. *Proceedings of the Royal Society B: Biological Sciences*, 270 (1512):313–321, 2 2003. ISSN 0962-8452. doi: 10.1098/rspb.2002.2218. URL http://rspb.royalsocietypublishing.org/cgi/doi/10.1098/rspb.2002.2218.

[86] S. Henikoff and J. G. Henikoff. Amino acid substitution matrices from protein blocks. *Proceedings of the National Academy of Sciences of the United States of America*, 1992. ISSN 00278424. doi: 10.1073/pnas.89.22.10915.

[87] D. M. Hillis and J. J. Bull. An Empirical Test of Bootstrapping as a Method for Assessing Confidence in Phylogenetic Analysis. *Systematic Biology*, 42(2):182–192, 1993. ISSN 1063-5157. doi: 10.1093/sysbio/42.2.182. URL http://sysbio.oxfordjournals.org/content/42/2/182.short.

[88] D. M. Hillis, M. W. Allard, and M. M. Miyamoto. Analysis of DNA Sequence Data: Phylogenetic Inference. *Methods in Enzymology*, 1993. ISSN 15577988. doi: 10.1016/0076-6879(93)24035-S.

[89] D. M. Hillis, D. D. Pollock, J. A. McGuire, and D. J. Zwickl. Is sparse taxon sampling a problem for phylogenetic inference? *Systematic biology*, 52(1):124–126, 2003. ISSN 1063-5157. doi: 10.1080/10635150390132911.

[90] C. E. Hinchliff, S. a. Smith, J. F. Allman, J. G. Burleigh, R. Chaudhary, L. M. Coghill, K. A. Crandall, J. Deng, B. T. Drew, R. Gazis, K. Gude, D. S. Hibbett, L. a. Katz, H. D. Laughinghouse, E. J. McTavish, P. E. Midford, C. L. Owen, R. H. Ree, J. a. Rees, D. E. Soltis, T. L. Williams, and K. a. Cranston. Synthesis of phylogeny and taxonomy into a comprehensive tree of life. *Proceedings of the National Academy of Sciences*, 112(41):12764–12769, 2015. ISSN 0027-8424. doi: 10.1073/pnas.1423041112. URL http://www.pnas.org/lookup/doi/10.1073/pnas.1423041112.

[91] M. Höhl and M. A. Ragan. Is multiple-sequence alignment required for accurate inference of phylogeny? *Systematic Biology*, 56(2):206–221, 2007. ISSN 10635157. doi: 10.1080/10635150701294741.

[92] W. Huang, L. Li, J. R. Myers, and G. T. Marth. ART: A next-generation sequencing read simulator. *Bioinformatics*, 28(4):593–594, 2012. ISSN 13674803. doi: 10.1093/bioinformatics/btr708.

[93] S. Hué, J. P. Clewley, P. A. Cane, and D. Pillay. HIV-1 pol gene variation is sufficient for reconstruction of transmissions in the era of antiretroviral therapy. *AIDS (London, England)*, 18(5):719–28, 2004. ISSN 0269-9370. doi: 10.1097/00002030-200403260-00002. URL http://www.ncbi.nlm.nih.gov/pubmed/15075506.

[94] S. Hué, A. E. Brown, M. Ragonnet-Cronin, S. Lycett, D. T. Dunn, E. Fearnhill, D. I. Dolling, A. Pozniak, D. Pillay, V. C. Delpech, and A. J. Leigh Brown. Phylogenetic analyses reveal HIV-1 infections between men misclassified as heterosexual transmissions. *Aids*, 28(13):1967–1975, 2014. ISSN 0269-9370. doi: 10.1097/QAD.0000000000000383. URL http://content.wkhealth.com/linkback/openurl?sid=WKPTLP:landingpage&an=00002030-201408240-00014.

[95] G. J. Hughes, E. Fearnhill, D. Dunn, S. J. Lycett, A. Rambaut, A. J. Leigh Brown, and UK HIV Drug Resistance Collaboration. Molecular phylodynamics of the heterosexual HIV epidemic in the United Kingdom. *PLoS pathogens*, 5(9):

e1000590, 9 2009. ISSN 1553-7374. doi: 10.1371/journal.ppat.1000590. URL http://dx.plos.org/10.1371/journal.ppat.1000590http://www.ncbi.nlm.nih.gov/pubmed/19779560http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC2742734.

[96] D. H. Huson. SplitsTree: analyzing and visualizing evolutionary data. *Bioinformatics*, 14(1):68–73, 1998. ISSN 1367-4803. doi: 10.1093/bioinformatics/14.1.68. URL https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/14.1.68.

[97] D. H. Huson, S. M. Nettles, and T. J. Warnow. Disk-covering, a fast-converging method for phylogenetic tree reconstruction. *Journal of computational biology*, 6(3-4):369–86, 1999. ISSN 1066-5277. doi: 10.1089/106652799318337. URL http://www.ncbi.nlm.nih.gov/pubmed/10582573.

[98] D. H. Huson, L. Vawter, and T. J. Warnow. Solving large scale phylogenetic problems using DCM2. *Proceedings / ... International Conference on Intelligent Systems for Molecular Biology ; ISMB. International Conference on Intelligent Systems for Molecular Biology*, pages 118–129, 1999. ISSN 15530833.

[99] D. Hyatt, G. L. Chen, P. F. LoCascio, M. L. Land, F. W. Larimer, and L. J. Hauser. Prodigal: Prokaryotic gene recognition and translation initiation site identification. *BMC Bioinformatics*, 2010. ISSN 14712105. doi: 10.1186/1471-2105-11-119.

[100] C. Jain, L. M. Rodriguez-R, A. M. Phillippy, K. T. Konstantinidis, and S. Aluru. High-throughput ANI Analysis of 90K Prokaryotic Genomes Reveals Clear Species Boundaries. *bioRxiv*, 9(1):5114, 12 2017. ISSN 2041-1723. doi: 10.1101/225342. URL http://www.nature.com/articles/s41467-018-07641-9.

[101] S. Janssen, D. McDonald, A. Gonzalez, J. A. Navas-Molina, L. Jiang, Z. Z. Xu, K. Winker, D. M. Kado, E. Orwoll, M. Manary, S. Mirarab, and R. Knight. Phylogenetic Placement of Exact Amplicon Sequences Improves Associations with Clinical Information. *mSystems*, 3(3):00021–18, 4 2018. ISSN 2379-5077. doi: 10.1128/mSystems.00021-18. URL http://msystems.asm.org/lookup/doi/10.1128/mSystems.00021-18.

[102] O. Jeffroy, H. Brinkmann, F. Delsuc, and H. Philippe. Phylogenomics: the beginning of incongruence? *Trends in Genetics*, 22(4):225–231, 2006. ISSN 01689525. doi: 10.1016/j.tig.2006.02.003. URL http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?cmd=Retrieve&db=PubMed&dopt=Citation&list_uids=16490279.

[103] Y. Jiang, M. Balaban, Q. Zhu, and S. Mirarab. DEPP: Deep Learning Enables Extending Species Trees using Single Genes. *bioRxiv (abstract in RECOMB 2021)*, page 2021.01.22.427808, 1 2021. ISSN 1063-5157. doi: 10.1101/2021.01.22.427808. URL http://biorxiv.org/content/early/2021/01/24/2021.01.22.427808.abstract.

[104] L. Jin and M. Nei. Limitations of the evolutionary parsimony method of phylogenetic

analysis. *Molecular Biology and Evolution*, 7(1):82–102, 01 1990. ISSN 0737-4038. doi: 10.1093/oxfordjournals.molbev.a040588. URL https://doi.org/10.1093/oxfordjournals.molbev.a040588.

[105] T. H. Jukes and C. R. Cantor. Evolution of protein molecules. In M. H. N, editor, *In Mammalian protein metabolism, Vol. III (1969), pp. 21-132*, volume III, pages 21–132. Academic Press, 1969. doi: 10.1234/12345678.

[106] S.-R. Jun, G. E. Sims, G. A. Wu, and S.-H. Kim. Whole-proteome phylogeny of prokaryotes by feature frequency profiles: An alignment-free method with optimal feature resolution. *Proceedings of the National Academy of Sciences*, 107(1):133–138, 2010. ISSN 0027-8424. doi: 10.1073/pnas.0913033107. URL http://www.pnas.org/cgi/doi/10.1073/pnas.0913033107.

[107] K. Katoh and H. Toh. Recent developments in the MAFFT multiple sequence alignment program. *Brief Bioinform*, 9(4):286–298, 7 2008. ISSN 1477-4054. doi: 10.1093/bib/bbn013. URL http://bib.oxfordjournals.org/cgi/content/abstract/9/4/286.

[108] M. Kimura. *The Neutral Theory of Molecular Evolution*. Cambridge University Press, 10 1983. ISBN 9780521231091. doi: 10.1017/CBO9780511623486. URL https://www.cambridge.org/core/product/identifier/9780511623486/type/book.

[109] A. G. Kluge and J. S. Farris. Quantitative Phyletics and the Evolution of Anurans. *Systematic Biology*, 18(1):1–32, mar 1969. ISSN 1063-5157. doi: 10.1093/sysbio/18.1.1. URL https://academic.oup.com/sysbio/article-lookup/doi/10.1093/sysbio/18.1.1.

[110] S. L. Kosakovsky Pond, S. Weaver, A. J. Leigh Brown, and J. O. Wertheim. HIV-TRACE (TRAnsmission Cluster Engine): a Tool for Large Scale Molecular Epidemiology of HIV-1 and Other Rapidly Evolving Pathogens. *Molecular Biology and Evolution*, 35 (7):1812–1819, 2018. ISSN 0737-4038. doi: 10.1093/molbev/msy016. URL https://academic.oup.com/mbe/article/35/7/1812/4833215.

[111] L. B. Koski and G. B. Golding. The Closest BLAST Hit Is Often Not the Nearest Neighbor. *Journal of Molecular Evolution*, 52(6):540–542, 6 2001. ISSN 0022-2844. doi: 10.1007/s002390010184. URL http://link.springer.com/10.1007/s002390010184.

[112] D. Koslicki, S. Foucart, and G. Rosen. Quikr: A method for rapid reconstruction of bacterial communities via compressive sensing. *Bioinformatics*, 29(17):2096–2102, 2013. ISSN 13674803. doi: 10.1093/bioinformatics/btt336.

[113] D. Koslicki, S. Foucart, and G. Rosen. WGSQuikr: Fast Whole-Genome Shotgun Metagenomic Classification. *PLoS ONE*, 9(3):e91784, 3 2014. ISSN 1932-6203. doi: 10.1371/journal.pone.0091784. URL https://dx.plos.org/10.1371/journal.pone.0091784.

[114] J. Köster, F. Mölder, K. P. Jablonski, B. Letcher, M. B. Hall, C. H. Tomkins-Tinch, V. Sochat, J. Forster, S. Lee, S. O. Twardziok, A. Kanitz, A. Wilm, M. Holtgrewe, S. Rahmann, and S. Nahnsen. Sustainable data analysis with Snakemake. *F1000Research*, 2021. ISSN 1759796X. doi: 10.12688/f1000research.29032.2.

[115] A. M. Kozlov, D. Darriba, T. Flouri, B. Morel, and A. Stamatakis. RAxML-NG: A fast, scalable and user-friendly tool for maximum likelihood phylogenetic inference. *Bioinformatics*, 2019. ISSN 14602059. doi: 10.1093/bioinformatics/btz305.

[116] L. Krause, N. N. Diaz, A. Goesmann, S. Kelley, T. W. Nattkemper, F. Rohwer, R. A. Edwards, and J. Stoye. Phylogenetic classification of short environmental DNA fragments. *Nucleic acids research*, 36(7):2230–9, 4 2008. ISSN 1362-4962. doi: 10.1093/nar/gkn038. URL http://nar.oxfordjournals.org/cgi/content/abstract/36/7/2230.

[117] K. Krogerus, R. Preiss, and B. Gibson. A Unique Saccharomyces cerevisiae × Saccharomyces uvarum Hybrid Isolated From Norwegian Farmhouse Beer: Characterization and Reconstruction. *Frontiers in Microbiology*, 9(SEP):1–15, 9 2018. ISSN 1664-302X. doi: 10.3389/fmicb.2018.02253. URL https://www.frontiersin.org/article/10.3389/fmicb.2018.02253/full.

[118] S. Kundu and J. Misra. A Linear Tree Partitioning Algorithm. *SIAM Journal on Computing*, 6(1):151–154, mar 1977. ISSN 0097-5397. doi: 10.1137/0206012. URL http://epubs.siam.org/doi/10.1137/0206012.

[119] Q. K. Langdon, D. Peris, B. Kyle, and C. T. Hittinger. sppIDer: A Species Identification Tool to Investigate Hybrid Genomes with High-Throughput Sequencing. *Molecular Biology and Evolution*, 9 2018. ISSN 0737-4038. doi: 10.1093/molbev/msy166. URL https://academic.oup.com/mbe/advance-article/doi/10.1093/molbev/msy166/5089242.

[120] Q. K. Langdon, D. Peris, E. P. Baker, D. A. Opulente, H.-V. Nguyen, U. Bond, P. Gonçalves, J. P. Sampaio, D. Libkind, and C. T. Hittinger. Fermentation innovation through complex hybridization of wild and domesticated yeasts. *Nature Ecology & Evolution*, 3(11):1576–1586, 11 2019. ISSN 2397-334X. doi: 10.1038/s41559-019-0998-8. URL http://dx.doi.org/10.1038/s41559-019-0998-8http://www.nature.com/articles/s41559-019-0998-8.

[121] A.-K. Lau, S. Dörrer, C.-A. Leimeister, C. Bleidorn, and B. Morgenstern. Read-SpaM: assembly-free and alignment-free comparison of bacterial genomes with low sequencing coverage. *BMC Bioinformatics*, 20(S20):638, 12 2019. ISSN 1471-2105. doi: 10.1186/s12859-019-3205-7. URL https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-019-3205-7.

[122] S. Q. Le and O. Gascuel. An improved general amino acid replacement matrix. *Molecular Biology and Evolution*, 2008. ISSN 07374038. doi: 10.1093/molbev/msn067.

[123] V. Lefort, R. Desper, and O. Gascuel. FastME 2.0: A comprehensive, accurate, and fast distance-based phylogeny inference program. *Molecular Biology and Evolution*, 32(10):2798–2800, 2015. ISSN 15371719. doi: 10.1093/molbev/msv150. URL http://mbe.oxfordjournals.org/lookup/doi/10.1093/molbev/msv150.

[124] A. J. Leigh Brown, S. Lycett, L. Weinert, G. Hughes, E. Fearnhill, and D. T. Dunn. Transmission network parameters estimated from HIV sequences for a nationwide epidemic. *Journal of Infectious Diseases*, 204(9):1463–1469, 2011. ISSN 00221899. doi: 10.1093/infdis/jir550. URL http://www.scopus.com/inward/record.url?eid=2-s2.0-80053427855&partnerID=tZOtx3y1.

[125] C.-A. Leimeister and B. Morgenstern. kmacs: the k -mismatch average common substring approach to alignment-free sequence comparison. *Bioinformatics*, 30(14):2000–2008, 7 2014. ISSN 1460-2059. doi: 10.1093/bioinformatics/btu331. URL https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btu331.

[126] C.-A. Leimeister, S. Sohrabi-Jahromi, and B. Morgenstern. Fast and accurate phylogeny reconstruction using filtered spaced-word matches. *Bioinformatics*, 33(7):btw776, 1 2017. ISSN 1367-4803. doi: 10.1093/bioinformatics/btw776. URL https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btw776.

[127] T. Leitner, D. Escanilla, C. Franzen, M. Uhlen, and J. Albert. Accurate reconstruction of a known HIV-1 transmission history by phylogenetic tree analysis. *Proceedings of the National Academy of Sciences*, 93(20):10864–10869, 10 1996. ISSN 0027-8424. doi: 10.1073/pnas.93.20.10864. URL http://www.ncbi.nlm.nih.gov/pubmed/8855273%5Cnhttp://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC38248http://www.pnas.org/cgi/doi/10.1073/pnas.93.20.10864.

[128] H. O. Letsch and K. M. Kjer. Potential pitfalls of modelling ribosomal RNA data in phylogenetic tree reconstruction: evidence from case studies in the Metazoa. *BMC evolutionary biology*, 11:146, 1 2011. ISSN 1471-2148. doi: 10.1186/1471-2148-11-146. URL http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3123606&tool=pmcentrez&rendertype=abstract.

[129] I. Letunic and P. Bork. Interactive Tree Of Life (iTOL) v5: an online tool for phylogenetic tree display and annotation. *Nucleic Acids Research*, 49(W1):W293–W296, 7 2021. ISSN 0305-1048. doi: 10.1093/nar/gkab301. URL https://academic.oup.com/nar/article/49/W1/W293/6246398.

[130] L. Li, C. J. Stoeckert, and D. S. Roos. OrthoMCL: Identification of Ortholog Groups for Eukaryotic Genomes. *Genome Research*, 13(9):2178–2189, 9 2003. doi: 10.1101/gr.1224503. URL http://genome.cshlp.org/content/13/9/2178.abstract.

[131] W. Li and A. Godzik. Cd-hit: a fast program for clustering and comparing large sets of

protein or nucleotide sequences. *Bioinformatics*, 22(13):1658–1659, jul 2006. ISSN 1367-4803. doi: 10.1093/bioinformatics/btl158. URL https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btl158.

[132] Li-San Wang, J. Leebens-Mack, P. K. Wall, K. Beckmann, C. W. de Pamphilis, and T. Warnow. The Impact of Multiple Protein Sequence Alignment on Phylogenetic Estimation. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 8(4):1108–1119, 7 2011. ISSN 1545-5963. doi: 10.1109/TCBB.2009.68. URL http://www.ncbi.nlm.nih.gov/pubmed/21566256http://doi.ieeecomputersociety.org/10.1109/TCBB.2009.68http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5235137http://ieeexplore.ieee.org/document/5235137/.

[133] P. Libin, E. V. Eynden, F. Incardona, A. Nowé, A. Bezenchek, A. Sönnerborg, A. M. Vandamme, K. Theys, and G. Baele. PhyloGeoTool: Interactively exploring large phylogenies in an epidemiological context. *Bioinformatics*, 2017. ISSN 14602059. doi: 10.1093/bioinformatics/btx535.

[134] D. Libkind, C. T. Hittinger, E. Valerio, C. Goncalves, J. Dover, M. Johnston, P. Goncalves, and J. P. Sampaio. Microbe domestication and the identification of the wild genetic stock of lager-brewing yeast. *Proceedings of the National Academy of Sciences*, 108(35):14539–14544, 8 2011. ISSN 0027-8424. doi: 10.1073/pnas.1105430108. URL https://www.pnas.org/content/108/35/14539http://www.pnas.org/cgi/doi/10.1073/pnas.1105430108.

[135] B. Linard, K. Swenson, and F. Pardi. Rapid alignment-free phylogenetic identification of metagenomic sequences. *Bioinformatics*, 35(18):3303–3312, 9 2019. ISSN 1367-4803. doi: 10.1093/bioinformatics/btz068. URL https://academic.oup.com/bioinformatics/article/35/18/3303/5303992.

[136] B. Linard, N. Romashchenko, F. Pardi, and E. Rivals. PEWO: a collection of workflows to benchmark phylogenetic placement. *Bioinformatics*, 2020. ISSN 1367-4803. doi: 10.1093/bioinformatics/btaa657.

[137] B. Liu, T. Gibbons, M. Ghodsi, and M. Pop. MetaPhyler: Taxonomic profiling for metagenomic sequences. In *Bioinformatics and Biomedicine (BIBM), 2010 IEEE International Conference on*, page 95–100. IEEE, 2011. ISBN 9781424483051. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5706544.

[138] K. Liu, S. Raghavan, S. M. Nelesen, C. R. Linder, and T. Warnow. Rapid and Accurate Large-Scale Coestimation of Sequence Alignments and Phylogenetic Trees. *Science*, 324(5934):1561–1564, 6 2009. ISSN 1095-9203. doi: 10.1126/science.1171243. URL http://www.sciencemag.org/content/324/5934/1561.abstracthttp://www.sciencemag.org/content/324/5934/1561.full.pdfhttp://www.sciencemag.org/cgi/content/abstract/324/5934/1561http://www.ncbi.nlm.nih.gov/pubmed/19541996.

[139] K. Liu, T. Warnow, M. T. Holder, S. M. Nelesen, J. Yu, A. Stamatakis, and C. R. Linder. SATe-II: Very Fast and Accurate Simultaneous Estimation of Multiple Sequence Alignments and Phylogenetic Trees. *Systematic Biology*, 61(1):90–106, 2011. ISSN 1076836X. doi: 10.1093/sysbio/syr095. URL http://www.ncbi.nlm.nih.gov/pubmed/22139466.

[140] J. R. Lobry. Properties of a general model of DNA evolution under no-strand-bias conditions. *Journal of Molecular Evolution*, 40(3):326–330, 1995. ISSN 14321432. doi: 10.1007/BF00163237.

[141] P. J. Lockhart, M. A. Steel, M. D. Hendy, and D. Penny. Recovering evolutionary trees under a more realistic model of sequence evolution. *Molecular Biology and Evolution*, 11(4):605–612, 7 1994. ISSN 1537-1719. doi: 10.1093/oxfordjournals.molbev.a040136. URL https://academic.oup.com/mbe/article/11/4/605/1007171/Recovering-evolutionary-trees-under-a-more.

[142] C. Lozupone and R. Knight. UniFrac : a New Phylogenetic Method for Comparing Microbial Communities UniFrac : a New Phylogenetic Method for Comparing Microbial Communities. *Applied and environmental microbiology*, 71(12):8228–8235, 2005. ISSN 0099-2240. doi: 10.1128/AEM.71.12.8228.

[143] G. Lunter, A. Rocco, N. Mimouni, A. Heger, A. Caldeira, and J. Hein. Uncertainty in homology inferences: assessing and improving genomic sequence alignment. *Genome research*, 18(2):298–309, 2 2008. ISSN 1088-9051. doi: 10.1101/gr.6725608. URL http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=2203628&tool=pmcentrez&rendertype=abstract.

[144] U. Mai and S. Mirarab. TreeShrink: Efficient Detection of Outlier Tree Leaves. In J. Meidanis and L. Nakhleh, editors, *Comparative Genomics: 15th International Workshop, RECOMB CG 2017, Barcelona, Spain, October 4-6, 2017, Proceedings*, pages 116–140. Springer International Publishing, Cham, 2017. ISBN 978-3-319-67979-2. doi: 10.1007/978-3-319-67979-2{_}7. URL https://doi.org/10.1007/978-3-319-67979-2_7.

[145] U. Mai and S. Mirarab. TreeShrink: fast and accurate detection of outlier long branches in collections of phylogenetic trees. *BMC Genomics*, 19(S5):272, 5 2018. ISSN 1471-2164. doi: 10.1186/s12864-018-4620-2. URL https://bmcgenomics.biomedcentral.com/articles/10.1186/s12864-018-4620-2.

[146] U. Mai, E. Sayyari, and S. Mirarab. Minimum variance rooting of phylogenetic trees and implications for species tree reconstruction. *PLOS ONE*, 12(8):e0182238, 8 2017. ISSN 1932-6203. doi: 10.1371/journal.pone.0182238. URL http://dx.plos.org/10.1371/journal.pone.0182238.

[147] B. L. Maidak. The RDP-II (Ribosomal Database Project). *Nucleic Acids Research*, 29(1):173–174, 1 2001. ISSN 13624962. doi: 10.1093/nar/29.1.173. URL https://academic.oup.

com/nar/article-lookup/doi/10.1093/nar/29.1.173.

[148] J. Mallet. Hybrid speciation. *Nature*, 446(7133):279–283, 3 2007. ISSN 0028-0836. doi: 10.1038/nature05706. URL http://www.nature.com/articles/nature05706.

[149] D. Mallo, L. De Oliveira Martins, and D. Posada. SimPhy : Phylogenomic Simulation of Gene, Locus, and Species Trees. *Systematic Biology*, 65(2):334–344, 3 2016. ISSN 1063-5157. doi: 10.1093/sysbio/syv082. URL http://biorxiv.org/content/early/2015/06/30/021709.abstracthttp://sysbio.oxfordjournals.org/content/early/2015/12/04/sysbio.syv082.short?rss=1https://academic.oup.com/sysbio/article-lookup/doi/10.1093/sysbio/syv082.

[150] G. Marçais and C. Kingsford. A fast, lock-free approach for efficient parallel counting of occurrences of k-mers. *Bioinformatics*, 2011. ISSN 13674803. doi: 10.1093/bioinformatics/btr011.

[151] F. A. Matsen. Phylogenetics and the Human Microbiome. *Systematic Biology*, 64(1): e26–e41, 1 2015. ISSN 1076-836X. doi: 10.1093/sysbio/syu053. URL http://arxiv.org/abs/1407.1794https://academic.oup.com/sysbio/article/64/1/e26/2847641.

[152] F. A. Matsen and S. N. Evans. Edge Principal Components and Squash Clustering: Using the Special Structure of Phylogenetic Placement Data for Sample Comparison. *PLoS ONE*, 8(3):e56859, 3 2013. ISSN 19326203. doi: 10.1371/journal.pone.0056859. URL https://dx.plos.org/10.1371/journal.pone.0056859.

[153] F. A. Matsen, R. B. Kodner, and E. V. Armbrust. pplacer: linear time maximum-likelihood and Bayesian phylogenetic placement of sequences onto a fixed reference tree. *BMC bioinformatics*, 11(1):538, 1 2010. ISSN 1471-2105. doi: 10.1186/1471-2105-11-538. URL http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3098090&tool=pmcentrez&rendertype=abstract.

[154] F. A. Matsen, N. G. Hoffman, A. Gallagher, and A. Stamatakis. A Format for Phylogenetic Placements. *PLoS ONE*, 7(2):e31009, 2 2012. ISSN 1932-6203. doi: 10.1371/journal.pone.0031009. URL https://dx.plos.org/10.1371/journal.pone.0031009.

[155] D. McDonald, M. N. Price, J. Goodrich, E. P. Nawrocki, T. Z. DeSantis, A. Probst, G. L. Andersen, R. Knight, and P. Hugenholtz. An improved Greengenes taxonomy with explicit ranks for ecological and evolutionary analyses of bacteria and archaea. *The ISME journal*, 6(3):610–618, 3 2012. ISSN 1751-7362. doi: 10.1038/ismej.2011.139. URL http://www.nature.com/articles/ismej2011139.

[156] D. McDonald, A. Birmingham, and R. Knight. Context and the human microbiome. *Microbiome*, 3(1):52, 12 2015. ISSN 2049-2618. doi: 10.1186/s40168-015-0117-2. URL http://www.microbiomejournal.com/content/3/1/52.

[157] A. B. R. McIntyre, R. Ounit, E. Afshinnekoo, R. J. Prill, E. Hénaff, N. Alexander, S. S. Minot, D. Danko, J. Foox, S. Ahsanuddin, S. Tighe, N. A. Hasan, P. Subramanian, K. Moffat, S. Levy, S. Lonardi, N. Greenfield, R. R. Colwell, G. L. Rosen, and C. E. Mason. Comprehensive benchmarking and ensemble approaches for metagenomic classifiers. *Genome Biology*, 18(1):182, 2017. ISSN 1474-760X. doi: 10.1186/s13059-017-1299-7. URL https://doi.org/10.1186/s13059-017-1299-7.

[158] S. R. Mehta, S. L. Kosakovsky Pond, J. A. Young, D. Richman, S. Little, and D. M. Smith. Associations between phylogenetic clustering and HLA profile among HIV-infected individuals in San Diego, California. *Journal of Infectious Diseases*, 205(10):1529–1533, 2012. ISSN 00221899. doi: 10.1093/infdis/jis231.

[159] F. Meyer, A. Bremges, P. Belmann, S. Janssen, A. C. McHardy, and D. Koslicki. Assessing taxonomic metagenome profilers with OPAL. *Genome Biology*, 20(1):51, 2019. ISSN 1474-760X. doi: 10.1186/s13059-019-1646-y. URL https://doi.org/10.1186/s13059-019-1646-y.

[160] M. M. Meyer. Revisiting the Relationships Between Genomic G + C Content, RNA Secondary Structures, and Optimal Growth Temperature. *Journal of Molecular Evolution*, 89(3):165–171, 4 2021. ISSN 0022-2844. doi: 10.1007/s00239-020-09974-w. URL http://link.springer.com/10.1007/s00239-020-09974-w.

[161] D. E. Miller, C. Staber, J. Zeitlinger, and R. S. Hawley. Highly Contiguous Genome Assemblies of 15 Drosophila Species Generated Using Nanopore Sequencing. *G3: Genes, Genomes, Genetics*, 8(10):3131–3141, oct 2018. ISSN 2160-1836. doi: 10.1534/g3.118.200160. URL http://www.ncbi.nlm.nih.gov/pubmed/30087105http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC6169393http://g3journal.org/lookup/doi/10.1534/g3.118.200160.

[162] S. Mirarab and T. Warnow. FastSP: linear time calculation of alignment accuracy. *Bioinformatics*, 27(23):3250–8, 2011. ISSN 1367-4811. doi: 10.1093/bioinformatics/btr553. URL http://www.ncbi.nlm.nih.gov/pubmed/21984754.

[163] S. Mirarab and T. Warnow. ASTRAL-II: coalescent-based species tree estimation with many hundreds of taxa and thousands of genes. *Bioinformatics*, 31(12):i44–i52, 6 2015. ISSN 1367-4803. doi: 10.1093/bioinformatics/btv234. URL http://bioinformatics.oxfordjournals.org/cgi/content/long/31/12/i44http://bioinformatics.oxfordjournals.org/lookup/doi/10.1093/bioinformatics/btv234.

[164] S. Mirarab, N. Nguyen, and T. Warnow. SEPP: SATé-Enabled Phylogenetic Placement. *Pacific Symposium On Biocomputing*, pages 247–58, 12 2012. ISSN 17935091. doi: 10.1142/9789814366496{_}0024. URL http://www.ncbi.nlm.nih.gov/pubmed/22174280http://www.worldscientific.com/doi/abs/10.1142/9789814366496_0024.

[165] S. Mirarab, R. Reaz, M. S. Bayzid, T. Zimmermann, M. S. Swenson, and T. Warnow. ASTRAL: genome-scale coalescent-based species tree estimation. *Bioinformatics*, 30 (17):i541–i548, 9 2014. ISSN 1367-4803. doi: 10.1093/bioinformatics/btu462. URL http://bioinformatics.oxfordjournals.org/cgi/content/long/30/17/i541http://bioinformatics.oxfordjournals.org/cgi/doi/10.1093/bioinformatics/btu462https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btu462.

[166] S. Mirarab, N. Nguyen, S. Guo, L.-S. Wang, J. Kim, and T. Warnow. Pasta: ultra-large multiple sequence alignment for nucleotide and amino-acid sequences. *Journal of Computational Biology*, 22(5):377–386, 2015.

[167] S. Mirarab, N. Nguyen, S. Guo, L.-S. Wang, J. Kim, and T. Warnow. PASTA: Ultra-Large Multiple Sequence Alignment for Nucleotide and Amino-Acid Sequences. *Journal of Computational Biology*, 22(05):377–386, 5 2015. ISSN 1066-5277. doi: 10.1089/cmb.2014.0156. URL http://online.liebertpub.com/doi/abs/10.1089/cmb.2014.0156.

[168] D. Mitchell and R. Bridge. A test of Chargaff's second rule. *Biochemical and Biophysical Research Communications*, 340(1):90–94, 2 2006. ISSN 0006291X. doi: 10.1016/j.bbrc.2005.11.160. URL https://linkinghub.elsevier.com/retrieve/pii/S0006291X05027130.

[169] N. Moshiri. Treeswift: a massively scalable python tree package. *bioRxiv*, 2018. doi: 10.1101/325522. URL https://www.biorxiv.org/content/early/2018/08/09/325522.

[170] N. Moshiri, M. Ragonnet-Cronin, J. O. Wertheim, and S. Mirarab. FAVITES: simultaneous simulation of transmission networks, phylogenetic trees and sequences. *Bioinformatics*, 35(11):bty921, 11 2018. ISSN 1367-4803. doi: 10.1093/bioinformatics/bty921. URL https://academic.oup.com/bioinformatics/advance-article/doi/10.1093/bioinformatics/bty921/5161084.

[171] L. Nakhleh. Computational approaches to species phylogeny inference and gene tree reconciliation. *Trends in Ecology and Evolution*, 28(12):719–728, 12 2013. ISSN 1872-8383. doi: 10.1016/j.tree.2013.09.004. URL http://www.sciencedirect.com/science/article/pii/S0169534713002139.

[172] S. Nayfach, Z. J. Shi, R. Seshadri, K. S. Pollard, and N. C. Kyrpides. New insights from uncultivated genomes of the global human gut microbiome. *Nature*, 568(7753):505–510, 4 2019. ISSN 14764687. doi: 10.1038/s41586-019-1058-x. URL http://dx.doi.org/10.1038/s41586-019-1058-x.

[173] L. T. Nguyen, H. A. Schmidt, A. Von Haeseler, and B. Q. Minh. IQ-TREE: A fast and effective stochastic algorithm for estimating maximum-likelihood phylogenies. *Molecular Biology and Evolution*, 32(1), 2015. ISSN 15371719. doi: 10.1093/molbev/msu300.

[174] N. N.-p. Nguyen, S. Mirarab, B. B. Liu, M. Pop, and T. Warnow. TIPP: Taxonomic

Identification and Phylogenetic Profiling. *Bioinformatics*, 30(24):3548–3555, 12 2014. ISSN 1367-4803. doi: 10.1093/bioinformatics/btu721. URL http://bioinformatics.oxfordjournals.org/cgi/doi/10.1093/bioinformatics/btu721.

[175] N.-P. Nguyen, T. Warnow, M. Pop, and B. White. A perspective on 16S rRNA operational taxonomic unit clustering using sequence similarity. *npj Biofilms and Microbiomes*, 2: 16004, 2016. ISSN 2055-5008.

[176] N.-P. N.-p. D. Nguyen, S. Mirarab, K. Kumar, and T. Warnow. Ultra-large alignments using phylogeny-aware profiles. *Genome Biology*, 16(1):124, 12 2015. ISSN 1465-6906. doi: 10.1186/s13059-015-0688-z. URL http://genomebiology.com/2015/16/1/124.

[177] T. H. Ogdenw and M. S. Rosenberg. Multiple sequence alignment accuracy and phylogenetic inference. *Systematic biology*, 55(2):314–328, 2006. ISSN 1063-5157. doi: 10.1080/10635150500541730.

[178] T. E. Oliphant. *A guide to NumPy*, volume 1. Trelgol Publishing USA, 2006.

[179] B. D. Ondov, T. J. Treangen, P. Melsted, A. B. Mallonee, N. H. Bergman, S. Koren, and A. M. Phillippy. Mash: fast genome and metagenome distance estimation using MinHash. *Genome Biology*, 17(1):132, 12 2016. ISSN 1474-760X. doi: 10.1186/s13059-016-0997-x. URL http://download.springer.com/static/pdf/329/art%253A10.1186%252Fs13059-016-0997-x.pdf?originUrl=http%3A%2F%2Fgenomebiology.biomedcentral.com%2Farticle%2F10.1186%2Fs13059-016-0997-x&token2=exp=1490224184~acl=%2Fstatic%2Fpdf%2F329%2Fart%25253A10.1186%25252F.

[180] A. Orakov, A. Fullam, L. P. Coelho, S. Khedkar, D. Szklarczyk, D. R. Mende, T. S. Schmidt, and P. Bork. GUNC: detection of chimerism and contamination in prokaryotic genomes. *Genome Biology*, 2021. ISSN 1474760X. doi: 10.1186/s13059-021-02393-0.

[181] D. H. Parks, M. Chuvochina, D. W. Waite, C. Rinke, A. Skarshewski, P. A. Chaumeil, and P. Hugenholtz. A standardized bacterial taxonomy based on genome phylogeny substantially revises the tree of life. *Nature Biotechnology*, 2018. ISSN 15461696. doi: 10.1038/nbt.4229.

[182] D. H. Parks, M. Chuvochina, P.-A. Chaumeil, C. Rinke, A. J. Mussig, and P. Hugenholtz. A complete domain-to-species taxonomy for Bacteria and Archaea. *Nature Biotechnology*, 38(9):1079–1086, 9 2020. ISSN 1087-0156. doi: 10.1038/s41587-020-0501-8. URL http://www.nature.com/articles/s41587-020-0501-8.

[183] D. H. Parks, M. Chuvochina, P. A. Chaumeil, C. Rinke, A. J. Mussig, and P. Hugenholtz. A complete domain-to-species taxonomy for Bacteria and Archaea. *Nature Biotechnology*, 2020. ISSN 15461696. doi: 10.1038/s41587-020-0501-8. URL http://dx.doi.org/10.1038/s41587-020-0501-8.

[184] A. Parley, S. Hedetniemi, and A. Proskurowski. Partitioning trees: Matching, domination, and maximum diameter. *International Journal of Computer & Information Sciences*, 10(1):55–61, feb 1981. ISSN 0091-7036. doi: 10.1007/BF00978378. URL http://link.springer.com/10.1007/BF00978378.

[185] E. Pasolli, F. Asnicar, S. Manara, M. Zolfo, N. Karcher, F. Armanini, F. Beghini, P. Manghi, A. Tett, P. Ghensi, M. C. Collado, B. L. Rice, C. DuLong, X. C. Morgan, C. D. Golden, C. Quince, C. Huttenhower, and N. Segata. Extensive Unexplored Human Microbiome Diversity Revealed by Over 150,000 Genomes from Metagenomes Spanning Age, Geography, and Lifestyle. *Cell*, 176(3):649–662, 1 2019. ISSN 10974172. doi: 10.1016/j.cell.2019.01.001. URL https://linkinghub.elsevier.com/retrieve/pii/S0092867419300017.

[186] M. J. Phillips, F. Delsuc, and D. Penny. Genome-scale phylogeny and the detection of systematic biases. *Molecular Biology and Evolution*, 2004. ISSN 07374038. doi: 10.1093/molbev/msh137.

[187] M. N. Price, P. S. Dehal, and A. P. Arkin. FastTree-2 – Approximately Maximum-Likelihood Trees for Large Alignments. *PLoS ONE*, 5(3):e9490, 3 2010. ISSN 1932-6203. doi: 10.1371/journal.pone.0009490. URL http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=2835736&tool=pmcentrez&rendertype=abstract.

[188] M. N. Price, P. S. Dehal, and A. P. Arkin. FastTree 2 - Approximately maximum-likelihood trees for large alignments. *PLoS ONE*, 5(3), 2010. ISSN 19326203. doi: 10.1371/journal.pone.0009490.

[189] L. Pritchard, R. H. Glover, S. Humphris, J. G. Elphinstone, and I. K. Toth. Genomics and taxonomy in diagnostics for food security: soft-rotting enterobacterial plant pathogens. *Analytical Methods*, 8(1):12–24, 2016. ISSN 1759-9660. doi: 10.1039/C5AY02550H. URL http://xlink.rsc.org/?DOI=C5AY02550H.

[190] C. Quast, E. Pruesse, P. Yilmaz, J. Gerken, T. Schweer, P. Yarza, J. Peplies, and F. O. Glöckner. The SILVA ribosomal RNA gene database project: improved data processing and web-based tools. *Nucleic acids research*, 41(D1):gks1219, 11 2012. ISSN 0305-1048. doi: 10.1093/nar/gks1219. URL http://academic.oup.com/nar/article/41/D1/D590/1069277/The-SILVA-ribosomal-RNA-gene-database-project.

[191] M. Rabiee and S. Mirarab. INSTRAL: Discordance-aware Phylogenetic Placement using Quartet Scores. *bioRxiv*, 432906(2):384–391, 8 2018. ISSN 1063-5157. doi: 10.1101/432906. URL http://biorxiv.org/content/early/2018/10/02/432906.abstract.

[192] M. Rabiee and S. Mirarab. Forcing external constraints on tree inference using ASTRAL. *BMC Genomics*, 2020. ISSN 14712164. doi: 10.1186/s12864-020-6607-z.

[193] E. Rachtman, M. Balaban, V. Bafna, and S. Mirarab. On the impact of contaminants

on the accuracy of genome skimming and the effectiveness of exclusion read filters. *Molecular Ecology Resources*, n/a(n/a), 2020. doi: 10.1111/1755-0998.13135. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/1755-0998.13135.

[194] E. Rachtman, S. Sarmashghi, V. Bafna, and S. Mirarab. Uncertainty Quantification Using Subsampling for Assembly-Free Estimates of Genomic Distance and Phylogenetic Relationships. *SSRN Electronic Journal*, 2021. ISSN 1556-5068. doi: 10.2139/ssrn.3986497. URL https://www.ssrn.com/abstract=3986497.

[195] M. Ragonnet-Cronin, E. Hodcroft, S. Hué, E. Fearnhill, V. Delpech, A. J. L. Brown, and S. Lycett. Automated analysis of phylogenetic clusters. *BMC bioinformatics*, 14(1):317, 2013.

[196] M. Ragonnet-Cronin, E. Hodcroft, S. Hué, E. Fearnhill, V. C. Delpech, A. J. L. Brown, S. Lycett, and S. Hue. Automated analysis of phylogenetic clusters. *BMC bioinformatics*, 14:317, 2013. ISSN 1471-2105. doi: 10.1186/1471-2105-14-317. URL http://www.scopus.com/inward/record.url?eid=2-s2.0-84887072241&partnerID=tZOtx3y1.

[197] A. Rahman, R. Chikhi, and P. Medvedev. Disk compression of k-mer sets. *Algorithms for Molecular Biology*, 16(1):10, 12 2021. ISSN 1748-7188. doi: 10.1186/s13015-021-00192-7. URL https://almob.biomedcentral.com/articles/10.1186/s13015-021-00192-7.

[198] G. Reinert, D. Chew, F. Sun, and M. S. Waterman. Alignment-free sequence comparison (I): statistics and power. *Journal of computational biology : a journal of computational molecular cell biology*, 16(12):1615–34, 12 2009. ISSN 1557-8666. doi: 10.1089/cmb.2009.0198. URL http://www.ncbi.nlm.nih.gov/pubmed/20001252http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC2818754.

[199] J. Ren, X. Bai, Y. Y. Lu, K. Tang, Y. Wang, G. Reinert, and F. Sun. Alignment-Free Sequence Analysis and Applications. *Annual Review of Biomedical Data Science*, 1(1):93–114, 7 2018. ISSN 2574-3414. doi: 10.1146/annurev-biodatasci-080917-013431. URL https://pubmed.ncbi.nlm.nih.gov/31828235https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6905628/https://www.annualreviews.org/doi/10.1146/annurev-biodatasci-080917-013431.

[200] P. Rice, I. Longden, and A. Bleasby. EMBOSS: The European Molecular Biology Open Software Suite. *Trends in Genetics*, 16(6):276–277, 6 2000. ISSN 01689525. doi: 10.1016/S0168-9525(00)02024-2. URL https://linkinghub.elsevier.com/retrieve/pii/S0168952500020242.

[201] D. Robinson and L. Foulds. Comparison of phylogenetic trees. *Mathematical Biosciences*, 53(1-2):131–147, 7 1981. ISSN 1557-8666. doi: 10.1089/cmb.2010.0185. URL http://www.sciencedirect.com/science/article/pii/0025556481900432.

[202] S. Röhling, T. Dencker, and B. Morgenstern. The number of k-mer matches between two DNA sequences as a function of k. *bioRxiv*, page 527515, 2019. doi: 10.1101/527515. URL https://www.biorxiv.org/content/10.1101/527515v2.

[203] G. L. Rosen, E. R. Reichenberger, and A. M. Rosenfeld. NBC: The naive Bayes classification tool webserver for taxonomic classification of metagenomic reads. *Bioinformatics*, 27(1):127–129, 2011. ISSN 13674803. doi: 10.1093/bioinformatics/btq619.

[204] T. Roychowdhury, A. Vishnoi, and A. Bhattacharya. Next-Generation Anchor Based Phylogeny (NexABP): Constructing phylogeny from Next-generation sequencing data. *Scientific Reports*, 3(1):2634, 12 2013. ISSN 2045-2322. doi: 10.1038/srep02634. URL http://www.nature.com/articles/srep02634.

[205] A. Rzhetsky and M. Nei. A Simple Method for Estimating and Testing Minimum-Evolution Trees. *Molecular Biology and Evolution*, 9(5):945–945, 09 1992. ISSN 0737-4038. doi: 10.1093/oxfordjournals.molbev.a040771. URL https://doi.org/10.1093/oxfordjournals.molbev.a040771.

[206] P. Sagulenko, V. Puller, and R. A. Neher. TreeTime: Maximum-likelihood phylodynamic analysis. *Virus Evolution*, 4(1):1–9, jan 2018. ISSN 2057-1577. doi: 10.1093/ve/vex042. URL http://academic.oup.com/ve/article/doi/10.1093/vex042/4794731.

[207] N. Saitou and M. Nei. The neighbour-joining method: a new method for reconstructing phylogenetic trees. *Molecular biology and evolution*, 4(4):406–425, 1987.

[208] A. Sand, M. Holt, J. Johansen, R. Fagerberg, G. Brodal, C. Pedersen, and T. Mailund. Algorithms for Computing the Triplet and Quartet Distances for Binary and General Trees. *Biology*, 2(4):1189–1209, 9 2013. ISSN 2079-7737. doi: 10.3390/biology2041189. URL http://www.mdpi.com/2079-7737/2/4/1189.

[209] S. Sarmashghi, K. Bohmann, M. T. P. Gilbert, V. Bafna, and S. Mirarab. Skmer: assembly-free and alignment-free sample identification using genome skims. *Genome Biology*, 20(1):34, 12 2019. ISSN 1474-760X. doi: 10.1186/s13059-019-1632-4. URL https://genomebiology.biomedcentral.com/articles/10.1186/s13059-019-1632-4.

[210] E. Sayyari, J. B. Whitfield, and S. Mirarab. Fragmentary Gene Sequences Negatively Impact Gene Tree and Species Tree Reconstruction. *Molecular Biology and Evolution*, 34(12):3279–3291, 12 2017. ISSN 0737-4038. doi: 10.1093/molbev/msx261. URL http://dx.doi.org/10.1093/molbev/msx261http://academic.oup.com/mbe/article/doi/10.1093/molbev/msx261/4344836/Fragmentary-gene-sequences-negatively-impact-genehttps://academic.oup.com/mbe/article/doi/10.1093/molbev/msx261/4344836/Fragmentary-gene-sequenc.

[211] P. D. Schloss and J. Handelsman. Introducing DOTUR, a Computer Program for

Defining Operational Taxonomic Units and Estimating Species Richness. *Applied and Environmental Microbiology*, 71(3):1501–1506, 3 2005. ISSN 0099-2240. doi: 10.1128/AEM.71.3.1501-1506.2005. URL https://aem.asm.org/content/71/3/1501.

[212] P. D. Schloss and S. L. Westcott. Assessing and Improving Methods Used in Operational Taxonomic Unit-Based Approaches for 16S rRNA Gene Sequence Analysis. *Applied and Environmental Microbiology*, 77(10):3219–3226, may 2011. ISSN 0099-2240. doi: 10.1128/AEM.02810-10. URL http://aem.asm.org/lookup/doi/10.1128/AEM.02810-10.

[213] A. Sczyrba, P. Hofmann, P. Belmann, D. Koslicki, S. Janssen, J. Droege, I. Gregor, S. Majda, J. Fiedler, and E. Dahms. Critical Assessment of Metagenome Interpretation−a benchmark of computational metagenomics software. *bioRxiv*, 14(11):99127, 11 2017. ISSN 1548-7091. doi: 10.1038/nmeth.4458. URL http://www.nature.com/articles/nmeth.4458.

[214] N. Segata, L. Waldron, A. Ballarini, V. Narasimhan, O. Jousson, and C. Huttenhower. Metagenomic microbial community profiling using unique clade-specific marker genes. *Nature Methods*, 9(8):811–814, 2012. ISSN 1548-7091. doi: 10.1038/nmeth.2066.

[215] N. Segata, D. Börnigen, X. C. Morgan, and C. Huttenhower. PhyloPhlAn is a new method for improved phylogenetic and taxonomic placement of microbes. *Nature Communications*, 2013. ISSN 20411723. doi: 10.1038/ncomms3304.

[216] W. Shen, S. Le, Y. Li, and F. Hu. SeqKit: A cross-platform and ultrafast toolkit for FASTA/Q file manipulation. *PLoS ONE*, 11(10):1–10, 12 2016. ISSN 19326203. doi: 10.1371/journal.pone.0163962. URL http://g3journal.org/lookup/doi/10.1534/g3.116.034744.

[217] F. Sievers, D. Dineen, A. Wilm, and D. G. Higgins. Making automated multiple alignments of very large numbers of protein sequences. *Bioinformatics*, 29(8):989–995, 2013.

[218] E. L. Sonnhammer and V. Hollich. Scoredist: A simple and robust protein sequence distance estimator. *BMC Bioinformatics*, 6:1–8, 2005. ISSN 14712105. doi: 10.1186/1471-2105-6-108.

[219] M. S. Springer and J. Gatesy. On the importance of homology in the age of phylogenomics. *Systematics and Biodiversity*, 16(3):1–19, 12 2017. ISSN 1477-2000. doi: 10.1080/14772000.2017.1401016. URL https://www.tandfonline.com/doi/full/10.1080/14772000.2017.1401016.

[220] A. Stamatakis. RAxML version 8: A tool for phylogenetic analysis and post-analysis of large phylogenies. *Bioinformatics*, 30(9):1312–1313, 5 2014. ISSN 14602059. doi: 10.1093/bioinformatics/btu033. URL https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btu033.

[221] M. Stark, S. A. Berger, A. Stamatakis, and C. von Mering. MLTreeMap–accurate Maximum Likelihood placement of environmental DNA sequences into taxonomic and functional reference phylogenies. *BMC genomics*, 11(1):461, 1 2010. ISSN 1471-2164. doi: 10.1186/1471-2164-11-461. URL http://www.biomedcentral.com/1471-2164/11/461.

[222] M. Steel. Recovering a tree from the leaf colourations it generates under a Markov model. *Applied Mathematics Letters*, 1994. ISSN 08939659. doi: 10.1016/0893-9659(94)90024-8.

[223] M. Steinegger and J. Söding. MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nature Biotechnology*, page 104034, oct 2017. ISSN 1087-0156. doi: 10.1038/nbt.3988. URL http://www.nature.com/doifinder/10.1038/nbt.3988.

[224] N. Sueoka. Intrastrand parity rules of DNA base composition and usage biases of synonymous codons. *Journal of Molecular Evolution*, 40(3):318–325, 1995. ISSN 14321432. doi: 10.1007/BF00163236.

[225] P. Sulo, D. Szabóová, P. Bielik, S. Poláková, K. Šoltys, K. Jatzová, and T. Szemes. The evolutionary history of Saccharomyces species inferred from completed mitochondrial genomes and revision in the 'yeast mitochondrial genetic code'. *DNA Research*, 24 (6):571–583, 12 2017. ISSN 1340-2838. doi: 10.1093/dnares/dsx026. URL http://academic.oup.com/dnaresearch/article/24/6/571/3868562.

[226] S. Sunagawa, D. R. Mende, G. Zeller, F. Izquierdo-Carrasco, S. a. Berger, J. R. Kultima, L. P. Coelho, M. Arumugam, J. Tap, H. B. Nielsen, S. Rasmussen, S. Brunak, O. Pedersen, F. Guarner, W. M. de Vos, J. Wang, J. Li, J. Dore, S. D. Ehrlich, a. Stamatakis, and P. Bork. Metagenomic species profiling using universal phylogenetic marker genes. *Nature methods*, 10(12):1196–1199, 12 2013. ISSN 1548-7105. doi: 10.1038/nmeth.2693. URL http://www.ncbi.nlm.nih.gov/pubmed/24141494.

[227] N. Takahata and M. Kimura. A model of evolutionary base substitutions and its application with special reference to rapid change of pseudogenes. *Genetics*, 1981. ISSN 00166731. doi: 10.1093/genetics/98.3.641.

[228] K. Tamura and M. Nei. Estimation of the Number of Nucleotide Substitutions in the Control Region of Mitochondrial-DNA in Humans and Chimpanzees. *Molecular biology and evolution*, 10(3):512–526, 1993. ISSN 0737-4038.

[229] K. Tamura, M. Nei, and S. Kumar. Prospects for inferring very large phylogenies by using the neighbor-joining method. *Proceedings of the National Academy of Sciences*, 101(30):11030–11035, 7 2004. ISSN 0027-8424. doi: 10.1073/pnas.0404206101. URL http://www.pnas.org/cgi/doi/10.1073/pnas.0404206101.

[230] K. Tang, J. Ren, and F. Sun. Afann: bias adjustment for alignment-free sequence comparison based on sequencing data using neural network regression. *Genome Biology*, 20(1):266, 12 2019. ISSN 1474-760X. doi: 10.1186/s13059-019-1872-3. URL https://genomebiology.biomedcentral.com/articles/10.1186/s13059-019-1872-3.

[231] S. Tavaré. Line-of-descent and genealogical processes, and their applications in population genetics models. *Theoretical Population Biology*, 26(2):119–164, 10 1984. ISSN 00405809. doi: 10.1016/0040-5809(84)90027-3. URL https://linkinghub.elsevier.com/retrieve/pii/0040580984900273.

[232] S. Tavaré. Some Probabilistic and Statistical Problems in the Analysis of DNA Sequences. *Lectures on Mathematics in the Life Sciences*, 17:57–86, 1986.

[233] L. R. Thompson, J. G. Sanders, D. McDonald, A. Amir, J. Ladau, K. J. Locey, R. J. Prill, A. Tripathi, S. M. Gibbons, G. Ackermann, J. A. Navas-Molina, S. Janssen, E. Kopylova, Y. Vázquez-Baeza, A. González, J. T. Morton, S. Mirarab, Z. Zech Xu, L. Jiang, M. F. Haroon, J. Kanbar, Q. Zhu, S. Jin Song, T. Kosciolek, N. A. Bokulich, J. Lefler, C. J. Brislawn, G. Humphrey, S. M. Owens, J. Hampton-Marcell, D. Berg-Lyons, V. McKenzie, N. Fierer, J. A. Fuhrman, A. Clauset, R. L. Stevens, A. Shade, K. S. Pollard, K. D. Goodwin, J. K. Jansson, J. A. Gilbert, R. Knight, J. L. A. Rivera, L. Al-Moosawi, J. Alverdy, K. R. Amato, J. Andras, L. T. Angenent, D. A. Antonopoulos, A. Apprill, D. Armitage, K. Ballantine, J. Bárta, J. K. Baum, A. Berry, A. Bhatnagar, M. Bhatnagar, J. F. Biddle, L. Bittner, B. Boldgiv, E. Bottos, D. M. Boyer, J. Braun, W. Brazelton, F. Q. Brearley, A. H. Campbell, J. G. Caporaso, C. Cardona, J. Carroll, S. C. Cary, B. B. Casper, T. C. Charles, H. Chu, D. C. Claar, R. G. Clark, J. B. Clayton, J. C. Clemente, A. Cochran, M. L. Coleman, G. Collins, R. R. Colwell, M. Contreras, B. B. Crary, S. Creer, D. A. Cristol, B. C. Crump, D. Cui, S. E. Daly, L. Davalos, R. D. Dawson, J. Defazio, F. Delsuc, H. M. Dionisi, M. G. Dominguez-Bello, R. Dowell, E. A. Dubinsky, P. O. Dunn, D. Ercolini, R. E. Espinoza, V. Ezenwa, N. Fenner, H. S. Findlay, I. D. Fleming, V. Fogliano, A. Forsman, C. Freeman, E. S. Friedman, G. Galindo, L. Garcia, M. A. Garcia-Amado, D. Garshelis, R. B. Gasser, G. Gerdts, M. K. Gibson, I. Gifford, R. T. Gill, T. Giray, A. Gittel, P. Golyshin, D. Gong, H.-P. Grossart, K. Guyton, S.-J. Haig, V. Hale, R. S. Hall, S. J. Hallam, K. M. Handley, N. A. Hasan, S. R. Haydon, J. E. Hickman, G. Hidalgo, K. S. Hofmockel, J. Hooker, S. Hulth, J. Hultman, E. Hyde, J. D. Ibáñez-Álamo, J. D. Jastrow, A. R. Jex, L. S. Johnson, E. R. Johnston, S. Joseph, S. D. Jurburg, D. Jurelevicius, A. Karlsson, R. Karlsson, S. Kauppinen, C. T. E. Kellogg, S. J. Kennedy, L. J. Kerkhof, G. M. King, G. W. Kling, A. V. Koehler, M. Krezalek, J. Kueneman, R. Lamendella, E. M. Landon, K. Lane-deGraaf, J. LaRoche, P. Larsen, B. Laverock, S. Lax, M. Lentino, I. I. Levin, P. Liancourt, W. Liang, A. M. Linz, D. A. Lipson, Y. Liu, M. E. Lladser, M. Lozada, C. M. Spirito, W. P. MacCormack, A. MacRae-Crerar, M. Magris, A. M. Martín-Platero, M. Martín-Vivaldi, L. M. Martínez, M. Martínez-Bueno, E. M. Marzinelli, O. U. Mason, G. D. Mayer, J. M. McDevitt-Irwin, J. E. McDonald, K. L. McGuire, K. D. McMahon, R. McMinds, M. Medina, J. R. Mendelson, J. L. Metcalf, F. Meyer, F. Michelangeli, K. Miller, D. A. Mills, J. Minich, S. Mocali, L. Moitinho-

Silva, A. Moore, R. M. Morgan-Kiss, P. Munroe, D. Myrold, J. D. Neufeld, Y. Ni, G. W. Nicol, S. Nielsen, J. I. Nissimov, K. Niu, M. J. Nolan, K. Noyce, S. L. O'Brien, N. Okamoto, L. Orlando, Y. O. Castellano, O. Osuolale, W. Oswald, J. Parnell, J. M. Peralta-Sánchez, P. Petraitis, C. Pfister, E. Pilon-Smits, P. Piombino, S. B. Pointing, F. J. Pollock, C. Potter, B. Prithiviraj, C. Quince, A. Rani, R. Ranjan, S. Rao, A. P. Rees, M. Richardson, U. Riebesell, C. Robinson, K. J. Rockne, S. M. Rodriguezl, F. Rohwer, W. Roundstone, R. J. Safran, N. Sangwan, V. Sanz, M. Schrenk, M. D. Schrenzel, N. M. Scott, R. L. Seger, A. Seguin-Orlando, L. Seldin, L. M. Seyler, B. Shakhsheer, G. M. Sheets, C. Shen, Y. Shi, H. Shin, B. D. Shogan, D. Shutler, J. Siegel, S. Simmons, S. Sjöling, D. P. Smith, J. J. Soler, M. Sperling, P. D. Steinberg, B. Stephens, M. A. Stevens, S. Taghavi, V. Tai, K. Tait, C. L. Tan, N. Tas¸, D. L. Taylor, T. Thomas, I. Timling, B. L. Turner, T. Urich, L. K. Ursell, D. van der Lelie, W. Van Treuren, L. van Zwieten, D. Vargas-Robles, R. V. Thurber, P. Vitaglione, D. A. Walker, W. A. Walters, S. Wang, T. Wang, T. Weaver, N. S. Webster, B. Wehrle, P. Weisenhorn, S. Weiss, J. J. Werner, K. West, A. Whitehead, S. R. Whitehead, L. A. Whittingham, E. Willerslev, A. E. Williams, S. A. Wood, D. C. Woodhams, Y. Yang, J. Zaneveld, I. Zarraonaindia, Q. Zhang, and H. Zhao. A communal catalogue reveals Earth's multiscale microbial diversity. *Nature*, 551(7681):457–463, 11 2017. ISSN 0028-0836. doi: 10.1038/nature24621. URL http://www.nature.com/doifinder/10.1038/nature24621.

[234] Y. Turakhia, B. Thornlow, A. S. Hinrichs, N. De Maio, L. Gozashti, R. Lanfear, D. Haussler, and R. Corbett-Detig. Ultrafast Sample Placement on Existing Trees (UShER) Empowers Real-Time Phylogenetics for the SARS-CoV-2 Pandemic. *bioRxiv*, 53(6):809–816, 6 2020. ISSN 1061-4036. doi: 10.1101/2020.09.26.314971. URL https://www.biorxiv.org/content/early/2020/09/28/2020.09.26.314971.

[235] I. Ulitsky, D. Burstein, T. Tuller, and B. Chor. The Average Common Substring Approach to Phylogenomic Reconstruction. *Journal of Computational Biology*, 13(2):336–350, 3 2006. ISSN 1066-5277. doi: 10.1089/cmb.2006.13.336. URL http://www.liebertpub.com/doi/10.1089/cmb.2006.13.336.

[236] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. Jarrod Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. Carey, İ. Polat, Y. Feng, E. W. Moore, J. Vand erPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and S. . . Contributors. SciPy 1.0–Fundamental Algorithms for Scientific Computing in Python. *arXiv e-prints*, art. arXiv:1907.10121, Jul 2019.

[237] C. von Mering, P. Hugenholtz, J. Raes, S. G. Tringe, T. Doerks, L. J. Jensen, N. Ward, and P. Bork. Quantitative Phylogenetic Assessment of Microbial Communities in Diverse Environments. *Science*, 315(5815):1126–1130, 2 2007. ISSN 0036-8075. doi: 10.1126/science.1133420. URL http://www.sciencemag.org/cgi/doi/10.1126/science.1133420.

[238] P. J. Waddell and M. Steel. General Time-Reversible Distances with Unequal Rates across Sites: Mixing Γ and Inverse Gaussian Distributions with Invariant Sites. *Molecular Phylogenetics and Evolution*, 8(3):398–414, 12 1997. ISSN 10557903. doi: 10.1006/mpev.1997.0452. URL http://linkinghub.elsevier.com/retrieve/pii/S1055790397904528.

[239] T. Warnow. *Computational phylogenetics: An introduction to designing methods for phylogeny estimation*. Cambridge University Press, 2017. ISBN 1316886921.

[240] T. Warnow. *Divide-and-Conquer Tree Estimation: Opportunities and Challenges*, pages 121–150. Springer International Publishing, Cham, 2019. ISBN 978-3-030-10837-3. doi: 10.1007/978-3-030-10837-3_6. URL https://doi.org/10.1007/978-3-030-10837-3_6.

[241] J. O. Wertheim, B. Murrell, S. R. Mehta, L. A. Forgione, S. L. Kosakovsky Pond, D. M. Smith, and L. V. Torian. Growth of HIV-1 Molecular Transmission Clusters in New York City. *The Journal of Infectious Diseases*, 2018. ISSN 0022-1899. doi: 10.1093/infdis/jiy431.

[242] M. V. Westbury, K. F. Thompson, M. Louis, A. A. Cabrera, M. Skovrind, J. A. S. Castruita, R. Constantine, J. R. Stevens, and E. D. Lorenzen. Ocean-wide genomic variation in Gray's beaked whales, Mesoplodon grayi. *Royal Society Open Science*, 8(3):rsos.201788, 3 2021. ISSN 2054-5703. doi: 10.1098/rsos.201788. URL https://royalsocietypublishing.org/doi/10.1098/rsos.201788.

[243] T. J. Wheeler. Large-scale neighbor-joining with NINJA. In *Algorithms in Bioinformatics*, pages 375–389. Springer, 2009. ISBN 3642042406.

[244] T. J. Wheeler and J. D. Kececioglu. Multiple alignment by aligning alignments. *Bioinformatics*, 23(13):559–568, 7 2007. doi: 10.1093/bioinformatics/btm226. URL http://bioinformatics.oxfordjournals.org/cgi/content/abstract/23/13/i559.

[245] S. Whelan and N. Goldman. A General Empirical Model of Protein Evolution Derived from Multiple Protein Families Using a Maximum-Likelihood Approach. *Molecular Biology and Evolution*, 18(5):691–699, 5 2001. ISSN 1537-1719. doi: 10.1093/oxfordjournals.molbev.a003851. URL https://academic.oup.com/mbe/article-lookup/doi/10.1093/oxfordjournals.molbev.a003851.

[246] J. Whitfield. Mathematics of Evolution and Phylogeny. * Edited by Olivier Gascuel. *Briefings in Bioinformatics*, 2008. ISSN 1467-5463. doi: 10.1093/bib/bbn036.

[247] D. D. Womble. GCG: The Wisconsin Package of sequence analysis programs. *Methods in molecular biology (Clifton, N.J.)*, 2000. ISSN 10643745. doi: 10.1385/1-59259-192-2:3.

[248] D. E. Wood and S. L. Salzberg. Kraken: ultrafast metagenomic sequence classification using exact alignments. *Genome Biology*, 15(3):R46, 2014. ISSN 1465-6906. doi:

10.1186/gb-2014-15-3-r46. URL http://genomebiology.biomedcentral.com/articles/10.1186/gb-2014-15-3-r46.

[249] G. A. Wu, S.-R. Jun, G. E. Sims, and S.-H. Kim. Whole-proteome phylogeny of large dsDNA virus families by an alignment-free method. *Proceedings of the National Academy of Sciences*, 106(31):12826–12831, 2009. ISSN 0027-8424. doi: 10.1073/pnas.0905115106. URL http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=2722272&tool=pmcentrez&rendertype=abstract%5Cnhttp://www.pnas.org/cgi/doi/10.1073/pnas.0905115106.

[250] X. Xia. Information-theoretic indices and an approximate significance test for testing the molecular clock hypothesis with genetic distances. *Molecular Phylogenetics and Evolution*, 52(3):665–676, 9 2009. ISSN 10557903. doi: 10.1016/j.ympev.2009.04.017. URL https://linkinghub.elsevier.com/retrieve/pii/S1055790309001572.

[251] X. Xia. DAMBE7: New and Improved Tools for Data Analysis in Molecular Biology and Evolution. *Molecular Biology and Evolution*, 35(6):1550–1552, 6 2018. ISSN 0737-4038. doi: 10.1093/molbev/msy073. URL https://academic.oup.com/mbe/article/35/6/1550/4970565.

[252] K. Yang and L. Zhang. Performance comparison between k-tuple distance and four model-based distances in phylogenetic tree reconstruction. *Nucleic Acids Research*, 36(5):e33–e33, 1 2008. ISSN 0305-1048. doi: 10.1093/nar/gkn075. URL https://academic.oup.com/nar/article-lookup/doi/10.1093/nar/gkn075.

[253] Z. Yang and A. D. Yoder. Estimation of the Transition/Transversion Rate Bias and Species Sampling. *Journal of Molecular Evolution*, 48(3):274–283, 3 1999. ISSN 0022-2844. doi: 10.1007/PL00006470. URL http://link.springer.com/10.1007/PL00006470.

[254] S. H. Ye, K. J. Siddle, D. J. Park, and P. C. Sabeti. Benchmarking Metagenomics Tools for Taxonomic Classification. *Cell*, 178(4):779–794, 8 2019. ISSN 1097-4172 (Electronic). doi: 10.1016/j.cell.2019.07.010.

[255] H. Yi and L. Jin. Co-phylog: an assembly-free phylogenomic approach for closely related organisms. *Nucleic Acids Research*, 41(7):e75–e75, 4 2013. ISSN 1362-4962. doi: 10.1093/nar/gkt003. URL https://academic.oup.com/nar/article/41/7/e75/1067663.

[256] C. Yin, G. Shen, D. Guo, S. Wang, X. Ma, H. Xiao, J. Liu, Z. Zhang, Y. Liu, Y. Zhang, K. Yu, S. Huang, and F. Li. InsectBase: a resource for insect genomes and transcriptomes. *Nucleic Acids Research*, 44(D1):D801–D807, 1 2016. ISSN 0305-1048. doi: 10.1093/nar/gkv1204. URL http://www.ncbi.nlm.nih.gov/pubmed/26578584http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC4702856https://academic.oup.com/nar/article-lookup/doi/10.1093/nar/gkv1204.

[257] O. Zagordi and J. R. Lobry. Forcing reversibility in the no-strand-bias substitution model allows for the theoretical and practical identifiability of its 5 parameters from pairwise DNA sequence comparisons. *Gene*, 347(2 SPEC. ISS.):175–182, 2005. ISSN 03781119. doi: 10.1016/j.gene.2004.12.019.

[258] C. Zhang, M. Rabiee, E. Sayyari, and S. Mirarab. ASTRAL-III: polynomial time species tree reconstruction from partially resolved gene trees. *BMC Bioinformatics*, 19(S6):153, 5 2018. ISSN 1471-2105. doi: 10.1186/s12859-018-2129-y. URL https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-018-2129-y.

[259] C. Zhang, Y. Zhao, E. L. Braun, and S. Mirarab. TAPER: Pinpointing errors in multiple sequence alignments despite varying rates of evolution. *Methods in Ecology and Evolution*, 2021. ISSN 2041210X. doi: 10.1111/2041-210X.13696.

[260] J. Zhang, K. Kobert, T. Flouri, and A. Stamatakis. PEAR: A fast and accurate Illumina Paired-End reAd mergeR. *Bioinformatics*, 2014. ISSN 13674803. doi: 10.1093/bioinformatics/btt593.

[261] Q. Zheng, C. Bartow-McKenney, J. S. Meisel, and E. A. Grice. HmmUFOtu: An HMM and phylogenetic placement based ultra-fast taxonomic assignment and OTU picking tool for microbiome amplicon sequencing studies. *Genome Biology*, 19(1):82, 12 2018. ISSN 1474-760X. doi: 10.1186/s13059-018-1450-0. URL https://genomebiology.biomedcentral.com/articles/10.1186/s13059-018-1450-0.

[262] Q. Zhu, C. L. Dupont, M. B. Jones, K. M. Pham, Z.-D. Jiang, H. L. DuPont, and S. K. Highlander. Visualization-assisted binning of metagenome assemblies reveals potential new pathogenic profiles in idiopathic travelers' diarrhea. *Microbiome*, 6(1):201, 12 2018. ISSN 2049-2618. doi: 10.1186/s40168-018-0579-0. URL https://microbiomejournal.biomedcentral.com/articles/10.1186/s40168-018-0579-0.

[263] Q. Zhu, U. Mai, W. Pfeiffer, S. Janssen, F. Asnicar, J. G. Sanders, P. Belda-Ferre, G. A. Al-Ghalith, E. Kopylova, D. McDonald, T. Kosciolek, J. B. Yin, S. Huang, N. Salam, J.-Y. Jiao, Z. Wu, Z. Z. Xu, K. Cantrell, Y. Yang, E. Sayyari, M. Rabiee, J. T. Morton, S. Podell, D. Knights, W.-J. Li, C. Huttenhower, N. Segata, L. Smarr, S. Mirarab, and R. Knight. Phylogenomics of 10,575 genomes reveals evolutionary proximity between domains Bacteria and Archaea. *Nature Communications*, 10(1):5477, 12 2019. ISSN 2041-1723. doi: 10.1038/s41467-019-13443-4. URL http://dx.doi.org/10.1038/s41467-019-13443-4http://www.nature.com/articles/s41467-019-13443-4.

[264] A. Zielezinski, S. Vinga, J. Almeida, and W. M. Karlowski. Alignment-free sequence comparison: Benefits, applications, and tools, 2017. ISSN 1474760X.

[265] A. Zielezinski, H. Z. Girgis, G. Bernard, C.-A. Leimeister, K. Tang, T. Dencker, A. K. Lau, S. Röhling, J. J. Choi, M. S. Waterman, M. Comin, S.-H. Kim, S. Vinga, J. S.

Almeida, C. X. Chan, B. T. James, F. Sun, B. Morgenstern, and W. M. Karlowski. Benchmarking of alignment-free sequence comparison methods. *Genome Biology*, 20 (1):1–44, 12 2019. ISSN 1474-760X. doi: 10.1186/s13059-019-1755-7. URL https://genomebiology.biomedcentral.com/articles/10.1186/s13059-019-1755-7.

[266] D. J. Zwickl and D. M. Hillis. Increased taxon sampling greatly reduces phylogenetic error. *Systematic biology*, 51(4):588–98, 8 2002. ISSN 1063-5157. doi: 10.1080/10635150290102339. URL http://sysbio.oxfordjournals.org/cgi/content/abstract/51/4/588.