ABSTRACT

Title of dissertation: ACCURATE AND SCALABLE PHYLOGENY ESTIMATION VIA GRAPH CUTS

Yunheng Han, Doctor of Philosophy, 2025

Dissertation directed by: Prof. Erin Molloy, Assistant Professor
Department of Computer Science, University of Maryland

Reconstructing evolutionary relationships among populations or species (called a species tree) is an important precursor of many biological studies, with applications in agriculture, medicine, and conservation. A major obstacle to species tree reconstruction is that the evolutionary histories of species can vary across the genome due to incomplete lineage sorting, a biological process modeled by the Multi-Species Coalescent. Many of the leading methods developed to date, for example, the ASTRAL family of methods, reconstruct the species tree from gene trees (i.e., a tree built from the related regions of the genomes across different species). ASTRAL address heterogeneity by evaluating evolutionary relationships on four species at a time, as these quartets have favorable statistical properties under the Multi-Species Coalescent. Despite significant advances, species tree reconstruction continues to be challenged by the complexity of evolutionary processes, the hardness of related optimization problems, and the quality of data; thus, new algorithms are needed. This dissertation presents advanced methods based on graph cuts, yielding improvements in scalability, accuracy, and robustness.

First, I introduce TREE-QMC, a heuristic for the NP-hard Maximum Quartet Support Species Tree problem. TREE-QMC is based on the divide-and-conquer approach proposed by Snir and

Rao (2010). My main contribution is showing how to build an object called the quartet graph directly from gene trees, without explicitly enumerating all quartets. This result enables TREE-QMC to be cubic time in the number of species, making it the first method to break the quartic time barrier without down-sampling the input quartets since the divide-and-conquer framework was proposed. Second, I introduce the notion of a normalized quartet graph and show how to efficiently integrate normalization into graph construction. Together, these contributions enable TREE-QMC to achieve greater accuracy and scalability than ASTRAL-III, the leading method at the time, on data sets with large numbers of taxa (500-1000) and other challenging conditions.

Second, I reformulate the TREE-QMC algorithm to weight quartets based on gene tree branch lengths and support values, as proposed by Zhang and Mirarab (2022). Although the weighting scheme improves robustness of TREE-QMC to poor quality inputs (i.e., gene trees with missing species and/or estimation error), it comes with a small increase in time complexity compared to the unweighted algorithm. Fortunately, the increase in running time is small in practice, behaving more like a constant factor. Moreover, weighted TREE-QMC is highly competitive with weighted ASTRAL-IV, the leading method at the time, again producing more accurate species trees on data sets with large numbers of taxa (500-1000) and other challenging conditions.

Third, I demonstrate the utility of TREE-QMC for evolutionary scenarios, like hybridization, where the species history is a network rather than a tree. Recent research shows that reconstructing the tree-like aspects of a network, called the tree of blobs, is important for scalable network reconstruction via divide-and-conquer. An obstacle here is that the leading method for tree of blob reconstruction, TINNiK, does not scale to large numbers of species. To address this issue, I propose to build a tree with TREE-QMC and then contract edges in it based on statistical testing. This approach enables greater scalability and accuracy than TINNiK, especially on data sets with high amounts of incomplete lineage sorting.

Overall, the algorithms presented in this dissertation advance species trees and tree of blob reconstruction and highlight avenues for future research.

ACCURATE AND SCALABLE PHYLOGENY ESTIMATION VIA GRAPH CUTS

by

Yunheng Han

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2025

Advisory Committee:
Prof. Erin Molloy, Assistant Professor, Chair/Advisor
Prof. Charles Delwiche, Professor, Dean's Representative
Prof. Mihai Pop, Professor
Prof. Robert Patro, Associate Professor
Prof. Laxman Dhulipala, Assistant Professor

Preface

The content of this thesis has been published in peer-reviewed journals or is under preparation for submission.

- **Chapter 2:** Han, Yunheng and Erin K. Molloy. (2023). Improving quartet graph construction for scalable and accurate species tree estimation from gene trees. *Genome Research* 33(7): 1042–1052.

- **Chapter 3:** Han, Yunheng and Erin K. Molloy. (2025). Improved robustness to gene tree incompleteness, estimation errors, and systematic homology errors with weighted TREE-QMC. *Systematic Biology*: syaf009.

Dedication

To my beloved Xiao, who has always been my source of love, strength, and support.

Table of Contents

Chapter 1: Introduction

A central tenant of evolution is that all organisms on Earth trace their origins back to a common ancestor. Going forward in time, the diversity of life observed today is the result of small and large genomic changes inherited across generations, with the frequency of these genomic changes in populations being shaped by mechanisms such as natural selection, genetic drift, and gene flow [34, 44]. Reconstructing the evolutionary relationships among populations or species (also called taxa) is an important precursor of many biological studies, with applications in agriculture, medicine, and conservation (e.g., studies of trait evolution [32]). Advances in sequencing technology over the last decade have dramatically increased the quantity of information available to reconstruct evolutionary relationships. However, there are still evolutionary relationships that are difficult to resolve (e.g., [38, 39, 58, 80, 117, 118, 121, 129, 130]). A major challenge to resolving these relationships is that the evolutionary history can vary across the genome. In particular, a tree built from the related regions of the genomes across different species (referred to as a gene tree) can differ from the species or population tree due to biological processes [80] or errors arising during gene tree estimation and earlier data processing steps [121]. One of the most promising approaches for addressing gene tree heterogeneity due to biological processes involves evaluating evolutionary relationships on four species at a time (called quartets); indeed, many of the leading methods, most notably the ASTRAL family of methods [77, 158–160], are based on quartets. However, even with the significant algorithmic and statistical advances over the last decade, there are still challenges with accuracy and scalability. This dissertation investigates quartet-based methods. We hypothesize that quartet-based methods based on the divide-and-conquer framework of Snir and

Rao [124] can enable more accurate reconstruction of large species trees and networks than the leading methods if challenges in scalability are overcome. In the remainder of this section, we review key concepts from species phylogenetics and then present an overview of this dissertation and its contributions.

1.1 Background

1.1.1 Gene trees vs. Species Trees

Although the concept of a species is difficult to define, for the purposes of this dissertation, a species can be viewed as a population with the property that individuals inherit genetic material from those in the previous generation. Populations splitting are represented as branching events (also called speciation events) and populations mixing are represented by reticulation events. Species trees include only branching events, whereas species networks also allow reticulation events.

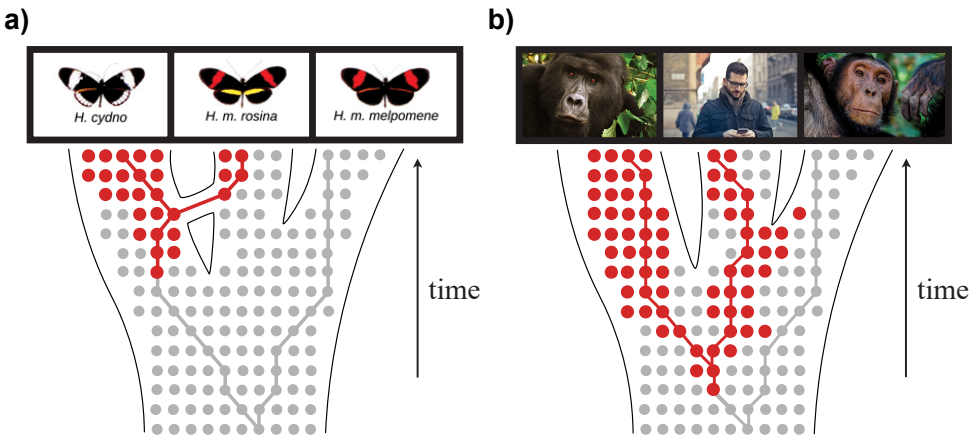


Figure 1.1: **Gene trees evolve within a) species network and b) species tree.** Each dot represents an individual (chromosome) in the population, and each row of dots represents the population of individuals at a specific generation in time (generations do not overlap). Lines indicate the transmission of a gene (e.g., a DNA sequence) from parent to offspring; thus, the lines taken together form the genealogy of the gene, also referred to as the gene tree.

In contrast, a gene tree describes the transmission of genetic material (i.e., a gene) from parents to offspring. Gene trees evolve within species trees and networks, which govern the pool(s) of potential ancestors, as shown in Fig. 1.1. In subfigure Fig. 1.1a, the gene tree evolves with **gene flow** because the *H. m. rosina* lineage can trace its ancestry back through two different ancestral populations: one shared with *H. cydno* and the other shared with *H. m. melpomene*. In subfigure Fig. 1.1b, the chimpanzee and human lineages do not coalesce (i.e., trace back to a common ancestor) in their most recent ancestral population, a phenomenon called **incomplete lineage sorting (ILS)**; this allows the gene tree to disagree with the species tree. ILS is more likely to occur when the species tree contains many short branches close together [28] (e.g., a rapid radiation of speciation events, as occurred for modern birds [54]). Because organisms do not typically inherit their entire genome from a single parent (consider sexually reproducing species), gene trees can vary across the genome, a phenomenon called **gene tree discordance**.

These ideas were brought to the forefront of phylogenetics by Maddison's seminal paper "Gene Trees in Species Trees" [68]. It is now well-established that both ILS and gene flow impact the evolution of major groups, including *whales* [139], *bats* [98], *birds* [54, 90], *butterflies* [27].

Models of Evolution. The **Multi-Species Coalescent (MSC)** models the evolution of gene trees within a species tree, allowing for ILS [22, 91, 112]. A similar model for species networks, called the **Network Multi-Species Coalescent (NMSC)** [151, 152], allows for both ILS and gene flow. Although these models generate gene trees, gene trees are not directly observed, rather the inherited DNA sequences are observed in present-day (extant) individuals or fossil records. Thus, evolution is modeled hierarchically. First, gene trees evolve within a species tree or network according to the MSC or NMSC, respectively. Second, a molecular sequence evolves down each gene tree (Fig. 1.2), producing molecular sequences for each of the species at the leaves. Models of molecular sequence evolution typically allow single nucleotide substitutions, like the Generalized Time Reversible (GTR) model [138], and sometimes small insertions and deletions [104].

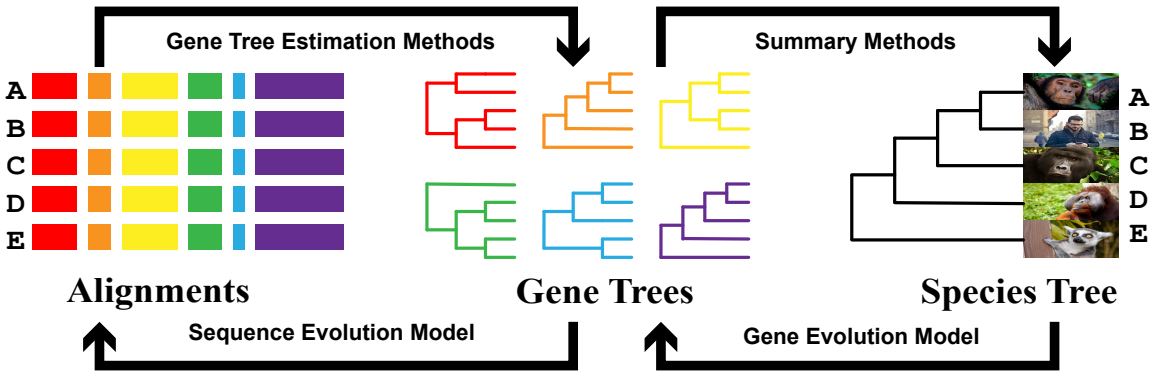


Figure 1.2: **Hierarchical models of evolution.** First, gene trees evolve within species tree. Second, a DNA sequence evolves down each gene tree, resulting in a DNA sequences for each gene tree. Human and non-human primate images are from Pexels [93–97].

1.1.2 Phylogeny Estimation

Existing phylogenetic methods differ widely in terms of the types of input data they require, the underlying biological models they assume, and the computational approaches they employ.

Traditional Phylogenetic Tree Estimation. Traditionally, phylogenetic methods were introduced in the context of reconstructing an evolutionary tree from sequences or characters. One of the earliest approaches, called Maximum Parsimony, seeks a tree that provides the simplest explanation of the data; for example minimizing the total number of substitutions [29, 35]. Maximum Parsimony is NP-hard [37] and can be **statistically inconsistent** under the standard models of molecular sequence evolution (e.g, the GTR model) [30], meaning the optimal tree under the maximum parsimony criterion will be different from the true (i.e. model) tree with high probability given a sufficiently large amount of data. In contrast, statistical methods, such as maximum likelihood (ML) and Bayesian inference, estimate the phylogeny under models of molecular sequence evolution. ML is typically consistent under the same model used for likelihood calculations with some modest assumptions; however, ML is NP-hard [108]. As a result, popular ML methods, such as RAxML [132] and IQ-TREE [87], apply search heuristics. Lastly, distance methods, such as

Neighbor Joining (NJ) [113] and UPGMA [154], reconstruct an evolutionary tree based on pairwise distances computed between sequences (e.g., the expected number of substitutions). Although most distance methods are fast, typically polynomial time, their accuracy is heavily dependent on the quality of the distance matrix.

Of these three approaches, ML is the most popular. It is worth noting that ML requires **aligned** sequences as input, meaning they are organized into a matrix, called a **multiple sequence alignment (MSA)**, such that each row corresponds to a sequence and each column corresponds to a site (i.e., nucleotides that evolved from a common ancestor). Along with the tree topology, ML methods estimate **branch lengths**, typically as the expected number of substitutions per site. Popular software packages also implement various ways of estimating **support** for a branch, a complex issue and active area of research [122].

Multi-locus Phylogenetic Tree Estimation. Traditional phylogenetic methods were designed for data sets with molecular sequences from one or a few genes. Today, phylogenetic methods are designed for data sets with many thousands of genes, referred to multi-locus data sets (Fig. 1.3). A popular approach concatenates the aligned DNA sequences for each gene together and then applies a traditional phylogenetic method (e.g., ML under the GTR model) to the supermatrix. This approach, referred to as **concatenation**, assumes all genes share the same evolutionary history. This assumption is violated in many biological systems that evolved with ILS or gene flow. Moreover, concatenation can be statistically inconsistent under the MSC [110].

The limitation of concatenation has led to the development of new methods that account for gene tree heterogeneity (e.g., [18, 63, 66, 67]). This dissertation focuses on a particular class of methods, called **summary methods**, which estimate species trees in a two-stage pipeline: first, a gene tree is estimated from the (aligned) DNA sequences for each gene, typically using ML, and second, the species tree is estimated from the gene trees using a summary method that accounts for gene tree heterogeneity, for example due to ILS.

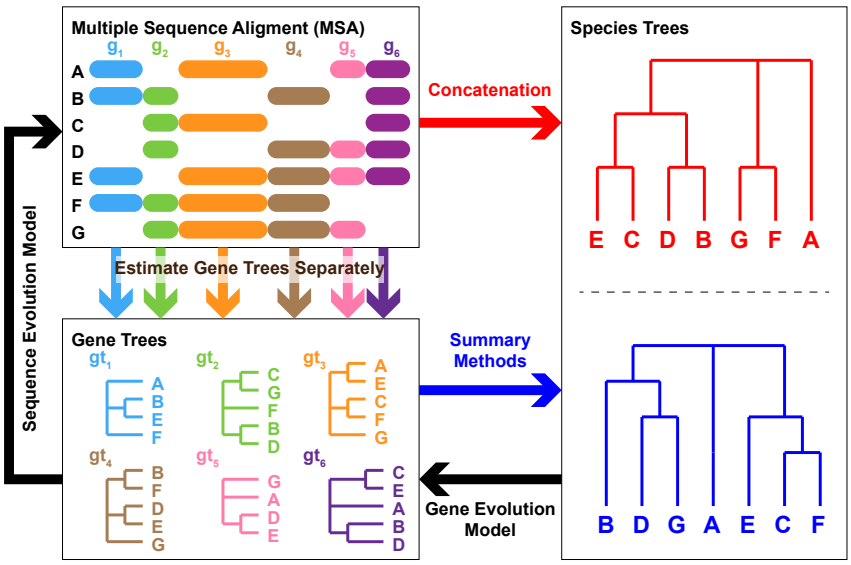


Figure 1.3: **Concatenation vs. summary methods.** The multi-locus data sets includes aligned DNA sequences for six genes (g_1 to g_6). In the concatenation approach, they are combined into a supermatrix from which a species tree is estimated. In the summary method approach, gene trees (gt_1 to gt_6) are estimated individually under a molecular sequence evolution model. These gene trees represent the evolutionary histories of individual regions of the genome (called loci), which may differ due to ILS. From the set of estimated gene trees, species trees are inferred using summary methods. This two step pipeline reverses the hierarchical model of evolution, which combines gene and sequence evolutionary models.

1.1.3 Quartet-based Summary Methods

The ASTRAL family of methods [77, 78, 157, 158], perhaps the most popular summary methods developed to date, reconstruct **unrooted** species tree from unrooted gene trees. Unrooting a phylogenetic tree (i.e., removing the directionality of edges) preserves evolutionary relationships but loses ancestor / descendant information (Fig. 1.4a–b). Unrooted phylogenetic trees with four leaves, called **quartets**, can display one of three possible evolutionary relationships among species (Fig. 1.4c). If there are more than four species, the unrooted tree can be decomposed into the set of quartets it displays (Fig. 1.4d). Broadly speaking, quartet-based summary methods reconstruct species trees from the quartets displayed by the input gene trees.

ASTRAL, in particular, is a heuristic for the **Maximum Quartet Support Species Tree (MQSST)**

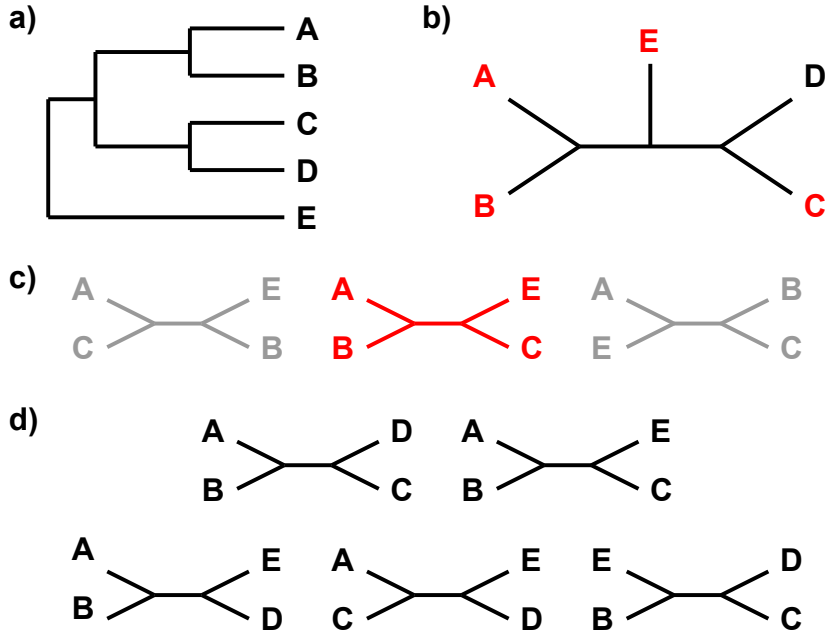


Figure 1.4: **Quartets. a)** A rooted phylogenetic tree on species set $\{A,B,C,D,E\}$. **b)** An unrooted version of the tree from subfigure (a). **c)** Quartets (unrooted trees with four species) have one of three possible topologies. The three quartets on species set $\{A,B,C,E\}$ are denoted $A,B|C,E$, $A,C|B,E$, $A,E|B,C$ going from left to right. The middle quartet (highlighted in red) is displayed by the unrooted tree in subfigure (b); the other two quartets are not. **d)** The complete set of $\binom{5}{4} = 5$ quartets displayed the unrooted tree in subfigure (b).

problem [77], which can be framed as weighting quartets (four-leaf trees) by their frequencies in input gene trees and then looking for a species tree T that maximizes the total weight of the quartets displayed by T. The optimal solution to MQSST is a statistically consistent estimator of the (unrooted) species tree under the MSC model [78]; however, MQSST is NP-hard [62].

A variety of heuristics have been developed for MQSST [6, 77, 103, 124, 156]. ASTRAL executes an exact (dynamic programming) algorithm for MQSST within a constrained version of the solution space constructed from the input gene trees [77]. The third version of ASTRAL (called ASTRAL-III) has a time complexity of $O((nk)^{1.726}x)$, where n is the number of species (also called taxa), k is the number of gene trees and $x = O(nk)$ is the size of the constrained solution space [157]. Because x depends on the amount of gene tree heterogeneity, a recent FASTRAL method [23] runs

ASTRAL-III in an aggressively constrained solution space to speed up species tree estimation. The latest version of ASTRAL, called ASTRAL-IV or ASTER [156], seeks to improve efficiency by introducing new techniques based on dynamic programming and phylogenetic placement, which seeks to add a single species to a tree, typically in an optimal fashion.

Perhaps the first widely used quartet-based method was **Quartet Max Cut (QMC)**, developed by Snir and Rao [7, 124, 125]. QMC is based on a top-down, divide-and-conquer framework. Each step in the divide phase determines a non-terminal branch in the output species tree. A branch, defined by the two subsets of taxa it separates, is found by (1) building a graph from the quartets displayed by the input gene trees, referred to as the **quartet graph**, and then (2) seeking an optimal cut under the objective function. The algorithm continues by recursion on each subset of taxa, terminating when there are three or fewer taxa, as there is only one possible tree that can be returned. These trees are joined together during the conquer phase. Quartet Fiduccia and Mattheyses (QFM) [69, 103] employs the same divide-and-conquer framework as QMC but applies a different technique for graph partitioning.

Importantly, QMC and QFM take quartets as input, so running them as summary methods requires quartets to be extracted from the gene trees. This preprocessing step has a time complexity $\Omega(n^4k)$, limiting scalability. On the other hand, a benefit of the QMC framework compared to ASTRAL-III is that the branches appearing in the output species tree are not restricted to those from the constraint set, which may be difficult to construct with high accuracy when gene trees are missing species, for example [78, 86]. Moreover, Mahbub et al. [69] found QFM to be more accurate than ASTRAL-III under challenging model conditions characterized by high levels of ILS as well as gene tree estimation error. This finding is central to our hypothesis that the QMC framework of Snir and Rao [124] can enable more accurate reconstruction of large species trees and networks than the leading methods if challenges in scalability are overcome.

1.1.4 Gene Tree Estimation Error and Other Practical Challenges

A major selling point of summary methods is that many are statistically consistent under the MSC, unlike concatenation. However, most proofs of consistency typically assume that the input gene trees are complete (i.e., not missing taxa) and error-free [111]. These conditions are unlikely to hold in practice [81, 129].

Three well-known issues that impact gene tree quality are estimation error, homology error, and incompleteness. **Gene tree estimation error (GTEE)** refers to inaccuracies arising during phylogeny estimation, for example due to model misspecification or low phylogenetic signal. **Homology errors** refer to mistakes in determining evolutionary relatedness [121, 130], for example the inclusion of unrelated sequences or errors in aligning DNA sequences. **Incompleteness** refers to gene trees missing one or more species, typically due to data processing issues (although this can also occur because of gene loss). In simulations, summary methods typically decrease in accuracy as GTEE or incompleteness increases [78, 81, 88, 147]. Moreover, a growing number of systematic studies show that homology errors at the site or sequence level can impact species tree estimates produced by different summary methods [121, 130].

The dominant approach to combating poor quality gene trees is gene tree filtering, in which entire gene trees are removed from the data set, typically based on some threshold for the proportion of missing taxa [21, 51] and/or proxies for GTEE [120]. However, simulation studies have suggested that filtering can reduce the accuracy of summary methods and that only gene trees with very high GTEE should be completely removed [81]. This finding motivates the use of less aggressive approaches, for example, contracting very low support branches in gene trees [119, 157] and/or removing specific taxa from individual gene trees. Taxa are typically removed based on the fraction of gaps in the MSA of the gene (referred to as fragmentary missing data by [115] and [89]) or based on (long) outlier branch lengths in the gene tree [70, 89]. Critically, these approaches require substantial effort and decision-making on the part of researchers, who must choose how to

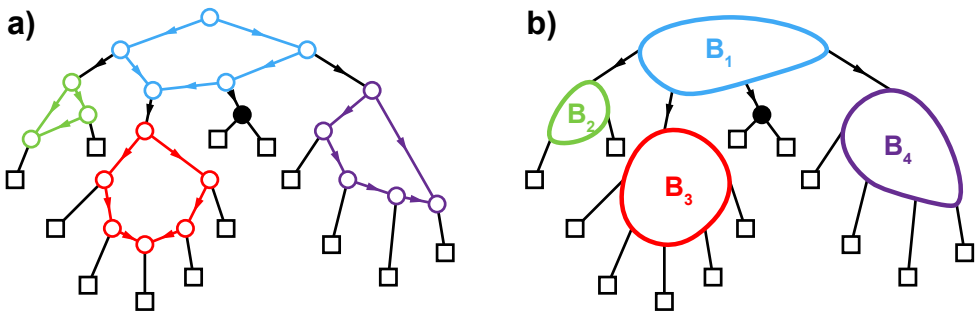


Figure 1.5: **Tree of blobs of a network. a)** Example of a level-1 species network. Four maximal bridgeless subgraphs, called "blobs", are highlighted in different colors (note that all other blobs in the network do not contain any edges). **b)** Tree of blobs for network shown in subfigure (a) built by contracting all edges in each blob into a single vertex.

quantify error, as well as select thresholds for filtering gene trees, contracting branches, and/or removing individual taxa. Often a range of parameter settings is explored, leading to many different estimates of the species tree (e.g., [51]).

To improve robustness to GTEE, Zhang and Mirarab [156] recently proposed to leverage gene tree branch lengths and support values to weight quartets within the popular summary method ASTRAL. However, these quartet weighting schemes make it challenging to design efficient algorithms, ultimately motivating the authors to development new search heuristics, specifically ASTRAL-IV discussed in Section 1.1.3.

1.1.5 Species Networks and Trees of Blobs

Similar concepts and challenges arise in species network reconstruction. One of the most popular methods, called SNaQ, takes gene trees as input and then reconstructs a species network under the NMSC via maximum pseudo-likelihood [126]. Like many phylogenetic network methods, the output of SNaQ belongs to a restricted class of networks, referred to as level-1 networks, with the property that no vertex belongs to more than one cycle, after undirecting the edges of the network.

Despite these limits on network complexity, SNaQ can only run on a small number of species

(around 30 [59]). There are two main challenges to scalability: first, heuristic search of network space, which is even larger than the tree space, and second, SNaQ's pseudo-likelihood function is computed over all quartets and thus is computationally intensive, although it is still more efficient to compute than the complete likelihood function [153]. Regardless, statistical methods are often favored over faster methods (e.g., distance-based methods [15] or split-based methods [25]) when gene trees can differ from each other for reasons other than gene flow (e.g., ILS and GTEE). To address scalability, Kolbow *et al.* [59] recently presented InPhyNet, a divide-and-conquer algorithm for reconstructing level-1 species networks, similar to the TreeMerge approach [82, 83] for divide-and-conquer species trees. Central to InPhyNet's approach is the decomposition of the species set into smaller subsets on which SNaQ can be run. Sampling species according to the **tree of blobs**, which represents the tree-like parts of a network [43], enables InPhyNet to be statistically consistent for level-1 networks. Unfortunately, trees of blobs, like the species networks, are challenging to reconstruct.

To our knowledge, the only method for reconstructing a tree of blobs under the NMSC is TINNiK [4]. TINNiK is statistically consistent and operates in two phases: first, hypothesis tests are applied to subsets of four species, and second, the results are used to reconstruct the tree of blobs using a distance-based approach. The first phase of TINNiK's algorithm appears to be $\Omega(n^4k)$ time, and the second phase appears to be $\Omega(n^5)$ time. Critically, TINNiK cannot scale to large numbers of taxa, requiring Kolbow et al. [59] to apply an alternative approach to subset decomposition that does not require a tree of blobs on the complete species set when implementing InPhyNet. It is also worth noting that an approach similar to TINNiK can be used for reconstructing species networks [2]. Overall, accurate and scalable network reconstruction remains a major open challenge in phylogenetics.

1.1.6 Phylogenetic Method Evaluation

Simulations are the gold standard for phylogenetic method evaluation. The idea is to simulate data under a model of evolution so that the true tree is known and can be compared to trees estimated from the simulated data. Benchmarking studies often focus on branch error [77, 78, 81, 88] (recall that branches in unrooted trees are defined by the two subsets of taxa they separate, called a **bipartition**). Branches in the true tree that are missing from the estimated tree are false negatives, whereas branches in the estimated tree that are missing from the true tree are false positives. The sum of these false positives and false negatives gives the Robinson-Foulds distance between two trees [107]. Often, these branch error metrics are normalized so that 0 represents no error (i.e., all branches are reconstructed correctly) and 1 indicates all error (i.e., no branches are reconstructed correctly). Evaluating differences between two phylogenetic networks is more complicated.

1.2 Overview of Dissertation

This dissertation is motivated by (1) the need for species tree and network methods that are both accurate and scalable and (2) our hypothesis that the QMC framework of [124] can enable more accurate reconstruction of large species trees and networks than the leading methods if challenges in scalability are overcome. The remainder of the dissertation is organized as follows (Fig. 1.6).

In Chapter 2, we present two algorithmic advancements to the QMC framework. First, we present an algorithm to build the quartet graph directly from gene trees without extracting all $\Omega(n^4k)$ quartets, where n is the number of species and k is the number of gene trees. Second, we introduce the notion of a normalized quartet graph and show how to efficiently integrate normalization into the graph construction algorithm. Together, these contributions form the basis of a new summary method, called **TREE-QMC**, which has time complexity $O(n^3k)$, with some assumptions on subproblem decomposition. In an evaluation study, normalization improved accuracy and

direct graph construction improved scalability, enabling TREE-QMC to outperform ASTRAL-III, the leading method at the time, on data sets with large numbers of taxa (500–1009) and other challenging conditions.

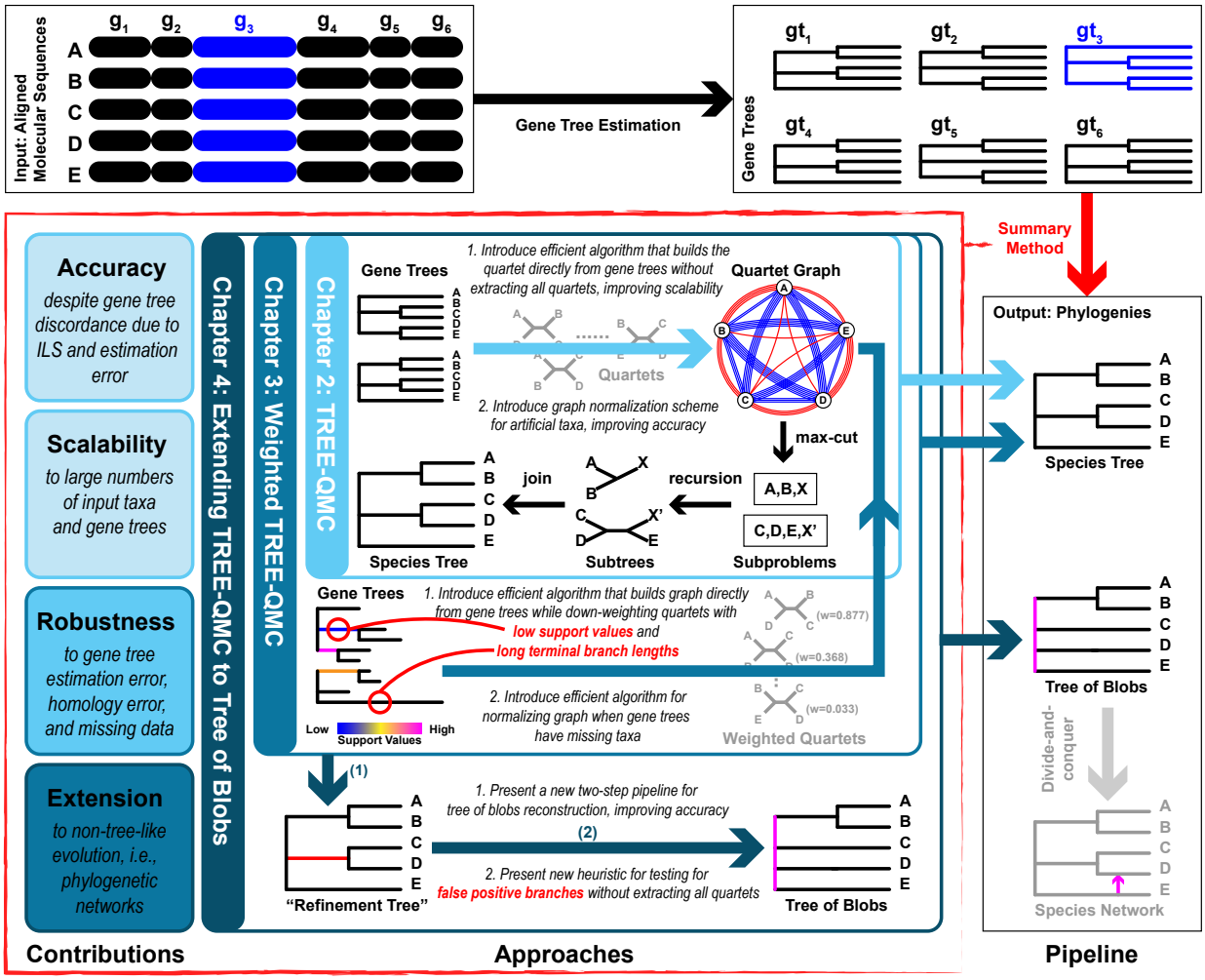


Figure 1.6: **Overview of the dissertation and its contributions.**

In Chapter 3, we further improve upon the QMC framework and the TREE-QMC method. First, we show that gene tree incompleteness complicates quartet graph normalization and provide an efficient algorithm to address this issue. Second, we introduce a new algorithm to compute the quartet graph directly from gene trees, while weighting quartets based on gene tree branch lengths and support values, as proposed by Zhang and Mirarab [156]. Perhaps surprisingly, the algorithm

has only a small increase in time complexity and no increase in storage complexity compared to unweighted TREE-QMC. In an evaluation study, **weighted TREE-QMC** was more robust to gene tree incompleteness, estimation error, and homology error, than the unweighted method. Moreover, weighted TREE-QMC was highly competitive with weighted ASTRAL-IV, the leading method at the time, producing more accurate species trees on data sets with large numbers of taxa (500–1000) and other challenging conditions.

In Chapter 4, we turn to the problem of network reconstruction, focusing on the tree of blobs. Motivated by the success of graph cuts for species tree estimation, we propose a new approach, tentatively called **TOB-QMC**, that operates in two steps: first, a **refinement tree** is built by running a quartet-based summary method given the input gene trees, and second, branches in the refinement tree are contracted based on hypothesis testing. For the first step, we use TREE-QMC, and for the second step, we use the same statistical tests as TINNiK [3]. An naive implementation of step two requires testing all subsets of four species "around" each branch, which is computationally intensive. We therefore propose a heuristic algorithm to search for the subset of four species that achieves the minimum p-value. By setting an iteration limit of $O(n)$ in the heuristic search, TOB-QMC reconstructs a tree of blobs in $O(n^3k)$ time, while being very close in accuracy to exhaustive search. In an evaluation study, TOB-QMC (with the heuristic search) was not only more computationally efficient than TINNiK but also more accurate, especially on data sets with high ILS.

Overall, this dissertation presents new algorithms that advance the scalability, accuracy, and robustness of the QMC framework, positioning it as a leading summary method for species tree reconstruction. Moreover, it demonstrates that the utility of the QMC framework for species network reconstruction. We conclude with limitations of this research and future directions in Chapter 5.

Chapter 2: TREE-QMC: Improving Quartet Graph Construction for Scalable and Accurate Species Tree Estimation from Gene Trees

This chapter has been published in the Genome Research *special issue for RECOMB 2023 [45]. Supplementary materials referenced in this chapter are freely available on* Zenodo *(*`https://doi.org/10.5281/zenodo.7478483`*) and reproduced in Appendix A.*

2.1 Introduction

Estimating the evolutionary history for a collection of species is a fundamental problem in evolutionary biology. Increasingly, species trees are estimated from multi-locus data sets, with molecular sequences partitioned into (recombination-free) regions of the genome (referred to as loci or genes). A popular approach to species tree estimation involves concatenating the alignments for individual loci together and then estimating a phylogeny under standard models of molecular sequence evolution, like the Generalized Time Reversible (GTR) model [138].

Such models assume the genes have a shared evolutionary history; however, this is not necessarily the case. The evolutionary histories of individual genes (referred to as gene trees) can differ from each other due to biological processes [68]. Incomplete lineage sorting (ILS), one of the most well-studied sources of gene tree discordance, is an outcome of genes evolving within populations of individuals, as modeled by the multi-species coalescent (MSC) [22, 91, 112]. Concatenation-based approaches to species tree estimation can be statistically inconsistent under the MSC [110]. Moreover, simulation studies have shown concatenation can perform poorly when the amount of

ILS is high (e.g. 61). ILS is expected to impact many major groups, including birds [54], placental mammals [71], and land plants [143]. Thus, species tree estimation methods that account for ILS, either explicitly or implicitly, are of interest.

An alternative to concatenation involves estimating gene trees (typically one per locus) and then applying a summary method. The most popular summary method to date, ASTRAL [77], is a heuristic for the NP-hard Maximum Quartet Support Species Tree (MQSST) problem [62], which can be framed as weighting quartets (four-leaf trees) by their frequencies in the input gene trees and then seeking a species tree T that maximizes the total weight of the quartets displayed by T. The optimal solution to MQSST is a statistically consistent estimator of the (unrooted) species tree under the MSC model [78], which is why heuristics for this problem are widely used in the context of multi-locus species tree estimation. Proofs of consistency typically assume the input gene trees are error-free [111]; however, this is unlikely in practice. An analysis of gene trees published for several recent systematic studies found low bootstrap support values on average (Table 1 in 81), suggesting that gene tree estimation error (GTEE) may be pervasive across modern phylogenomics data sets. GTEE can negatively impact the accuracy of summary methods, as demonstrated by simulation (e.g. 147) and systematic studies (e.g. 73). Overall, GTEE and ILS present significant challenges to species tree estimation.

A third challenge is scalability. ASTRAL executes an exact (dynamic programming) algorithm for MQSST within a constrained version of the solution space constructed from the input gene trees. There have been many improvements to ASTRAL, with the latest version ASTRAL-III [157] has a time complexity of $O((nk)^{1.726}x)$, where n is the number of species (also called taxa), k is the number of gene trees, and $x = O(nk)$ is the size of the constrained solution space. Because x depends on the amount of gene tree heterogeneity, a recent method FASTRAL [23] runs ASTRAL-III in an aggressively constrained solution space to speedup species tree estimation.

The other popular quartet methods, wQMC [7] and wQFM [69], take weighted quartets as input and then execute a divide-and-conquer approach to phylogeny reconstruction. A recent study found

wQFM to be more accurate than ASTRAL-III on challenging model conditions characterized by high ILS and high GTEE [69]. In these analyses, wQFM was given $\Theta(n^4)$ quartets as input, with each quartet weighted by the number of gene trees that displayed it. The related input processing step limits scalability of this approach. Here, we enable improved accuracy and scalability by introducing TREE-QMC.

2.2 Results

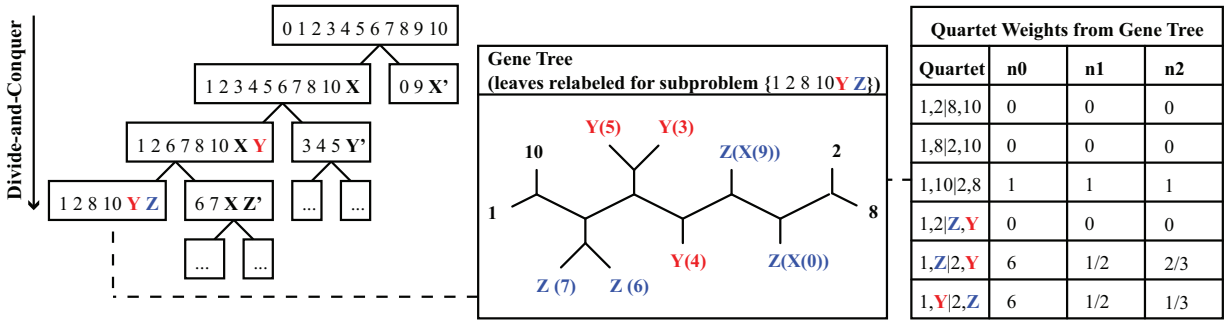
2.2.1 Overview of TREE-QMC Method

TREE-QMC builds upon the first widely-used quartet method, wQMC, which reconstructs the species tree in a divide-and-conquer fashion. At each step in the divide phase, an internal branch in the output species tree is identified; this branch splits the taxa into two disjoint subsets (Figure 2.1). The algorithm continues by recursion on the subproblems implied by the two subsets of taxa."Artificial taxa" are introduced to represent the species on the opposite of the branch so that solutions to subproblems can be combined during the conquer phase. The recursion terminates when the subproblem has three or fewer taxa, as there is only one possible tree that can be returned. At each step in the conquer phase, trees for complementary subproblems are connected at the related artificial taxa, until there is a single tree on the original set of species (Supplemental Figure A.1).

Central to wQMC's approach is a graph built from weighted quartets. This graph is constructed in such a way that its max cut should correspond to a branch in the output species tree [7, 124, 125]. Our observation is that quartets on artificial taxa can have higher weights than quartets on only non-artificial taxa (called singletons) when looking at a single gene tree (Figure 2.1). As we will show, normalizing the quartet weights so that each gene tree gets one vote for every subset of four species improves accuracy. The best performing normalization scheme (n2) weights quartets based on the subproblem decomposition; essentially, quartets are upweighted if their taxa are more

closely related to the current subproblem (note: n1 denotes uniform normalization and n0 denotes no normalization). Moreover, we provide an algorithm to build the (normalized) quartet graph directly from the input gene trees, enabling TREE-QMC to have a time complexity of $O(n^3k)$ if the subproblem decomposition is perfectly balanced (Theorem A.3 in the Supplemental Materials). This analysis is for a highly idealized setting and ignores large constant factors (Theorem A.2 in the Supplemental Materials).

Beyond time complexity, methods can differ from each other in terms of data locality, code optimizations, and other theoretical guarantees (e.g., ASTRAL is guaranteed to find an optimal solution within its constrained solution space, whereas TREE-QMC has no such guarantee). Thus, in the remainder of this paper, we focus on evaluating the empirical performance of TREE-QMC (and its different normalization schemes) against the leading quartet-based summary methods on simulated and biological data.



Quartet Weights from Gene Tree

Quartet	n0	n1	n2
1,2|8,10	0	0	0
1,8|2,10	0	0	0
1,10|2,8	1	1	1
1,2|Z,Y	0	0	0
1,Z|2,Y	6	1/2	2/3
1,Y|2,Z	6	1/2	1/3

Figure 2.1: At each step in the divide phase, taxa are split into two disjoint subsets and then artificial taxa are introduced to represent the species on the other side of the split. To compute the quartet weights for a given subproblem, the leaves of each gene tree are relabeled by the artificial taxa. Without normalization (column n0), quartet $1,2|Y,Z$ gets 0 votes and the alternative quartets get 6 votes each (note: quartet $1,Y|2,Z$ gets 6 votes by taking either species 5, 3, or 4 for label Y and either species 0 or 9 for label Z). With normalization, each gene tree gets one vote for each subset of four labels, although this vote can be split across the three possible quartets. In the uniform normalization scheme (column n1), we simply divide column n0 by the total number of votes cast in the unnormalized case. In the non-uniform normalization scheme (column n2), we leverage that structure implied by the divide phase of the algorithm; the idea is that species should have lesser importance each time they are re-labeled by artificial taxa.

2.2.2 Experimental Evaluation

We now give an overview of our simulation study; details are provided in the Supplemental Materials.

Methods

TREE-QMC is compared against five leading quartet methods: wQMC v1.3, wQFM v3.0, ASTRAL v5.5.7 (denoted ASTRAL-III or ASTRAL3), and FASTRAL. Two of these methods, wQMC and wQFM, which take weighted quartets instead of gene trees as input (the input processing step is performed using the script distributed with wQFM). All methods are run in default mode. The current version of TREE-QMC requires binary gene trees as input so polytomies in the estimated gene trees are refined arbitrarily before running TREE-QMC (the same refinements are used in all runs of TREE-QMC to ensure a fair comparison across the normalization schemes).

Evaluation Criteria

All methods are compared in terms of species tree error, quartet score, and runtime. Species tree error is the percent Robinson-Foulds (RF) error (i.e., normalized RF distance between the true and estimated species trees multiplied by 100). Because the true and estimated species tree are both binary, the RF error rate is equivalent to false negative error rate (i.e., the fraction of internal branches in the estimated species tree that are incorrect and thus missing from the true species tree). Two-sided Wilcoxon signed-rank tests are used to evaluate differences between TREE-QMC-n2 versus FASTRAL as well as TREE-QMC-n2 versus ASTRAL3 (TREE-QMC-n2 is also compared against wQFM when possible).

We report the difference in the quartet scores between the estimated and true species tree (scores are computed using the same set of gene trees). The quartet score is the number of quartets in

the input gene trees that are displayed by the output species tree (divided by the total number of quartets in the gene trees). This quantity is simply the (normalized) MQSST objective function, so higher quartet scores imply a better solution to MQSST.

The runtime is the wall clock time, which for wQFM and wQMC includes the time to weight quartets based on the input gene trees (the fraction of time spent on input processing phase is reported in the Supplemental Materials). All methods are run on the same data set on the same compute node on our cluster; the maximum wall clock time is 18 hours.

Simulated data sets

Our benchmarking study utilizes data simulated in prior studies, specifically the ASTRAL-II simulated data sets [78] as well as the avian and mammalian simulated data sets [79]. These data are generated by (1) taking a model species tree, (2) simulating gene trees within the species tree under the MSC, (3) simulating sequences down each gene tree under the GTR model, and (4) estimating a tree from set of gene sequences. Either the true gene trees from step 2 or the estimated gene trees from step 4 can be given as input to methods. This process is repeated for various parameter settings.

The ASTRAL-II data sets are generated from model species trees simulated under the Yule model given three parameters: species tree height, speciation rate, and number of taxa. The speciation rate is set so that speciation events are clustered near the root (deep) or near the tips (shallow) of the species tree. There are 50 replicates for each model condition (note that a new model species tree is simulated each replicate data set). The avian and mammalian simulated data sets are generated from published species trees estimated for 48 birds [54] and 37 mammals [128], respectively. The species tree branches are scaled to vary the amount of ILS, and the sequence length is changed to vary the amount of GTEE. There are 20 replicates for each model condition.

The data properties (ILS and GTEE levels) are summarized in Supplemental Tables A.1 and

A.2. The ILS level is the percent RF error (between the true species tree and the true gene tree) averaged across all gene trees, and GTEE level is the percent RF error (between the true and estimated gene trees) averaged across all gene trees. Overall, these data sets cover a range of model conditions. The results are presented in four experiments looking at the impact of varying the number of taxa, the species tree scale/height (proxy for ILS), the sequence length (proxy for GTEE), and the number of genes.

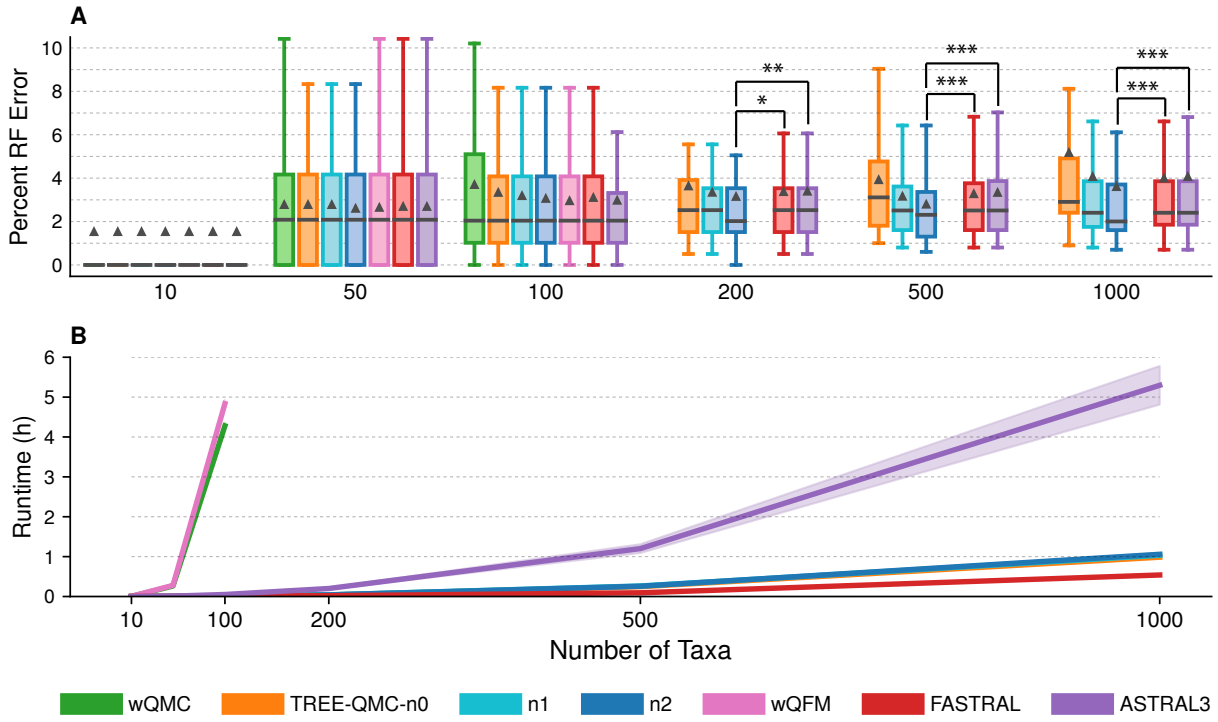


Figure 2.2: Impact of number of taxa. (A) Percent species tree error across replicates (bars represent medians; triangles represent means; outliers are not shown). The symbols *, **, and *** indicate significance at $p < 0.05$, 0.005, and 0.0005, respectively (all but one test survives Bonferroni multiple comparison correction; see Supplemental Table A.4 for details). (B) Mean runtime across replicates (shaded region indicates standard error). All data sets have species tree height 1×, shallow speciation, and 1000 estimated genes trees. The ILS level is 17–35%, and GTEE level is 19–30%. Note: the input processing for wQMC and wQFM does not run within our maximum wall clock time of 18 hours for data sets with 200 or more taxa.

Number of Taxa

Figure 2.2A–B shows the impact of varying the number of taxa. The pipelines that need weighted quartets to be given as input (wQFM and wQMC) run on the order of seconds for 10 taxa, minutes for 50 taxa, and hours for 100 taxa. The runtime of these pipelines is dominated by the time to weight $\Theta(n^4)$ quartets by their frequency in the input gene trees (Supplemental Table A.3). This input processing step does not complete on our compute nodes within our maximum wallclock time of 18 hours for most data sets with 200 taxa. Therefore, we could not run wQMC and wQFM on data sets with 200, 500, or 1000 taxa. In contrast, TREE-QMC implements a similar approach to wQMC but bypasses the input processing step, scaling to 1000 taxa and 1000 genes. For these data sets, FASTRAL, TREE-QMC-n2, and ASTRAL-III complete on average in 32 minutes, 64 minutes, and 5.3 hours, respectively (note: ASTRAL-III fails to complete on 3/50 replicates within our maximum wall clock time of 18 hours). Thus, TREE-QMC-n2 is much faster than ASTRAL-III and is not much slower than FASTRAL. TREE-QMC-n2 is significantly more accurate than either FASTRAL or ASTRAL-III when the number of taxa is 200 or greater. For these same conditions, quartet weight normalization, and especially the non-uniform (n2) scheme, improves TREE-QMC's accuracy. Results for methods given true gene trees as input or only 250 (out of 1000) gene trees as input are shown in Supplemental Figures A.7–A.9).

Incomplete Lineage Sorting (ILS)

ASTRAL-II data (200 taxa, 1000 estimated gene trees). Figure 2.3A–B shows the impact of varying the species tree height and thus the amount of ILS for the ASTRAL-II data sets. TREE-QMC-n2, FASTRAL, and ASTRAL-III produce highly accurate species trees, with median species tree error at or below 6% for all model conditions (note: the input processing for wQMC and wQFM does not run within our maximum wall clock time of 18 hours for these 200-taxon data sets). For some conditions, TREE-QMC-n2 is significantly more accurate than FASTRAL or

ASTRAL-III; otherwise, there are no significant difference between these pairs of methods. Quartet weight normalization improves the accuracy of TREE-QMC; this effect is most pronounced when the amount of ILS was very high (species tree height: 0.5×). On these same conditions, ASTRAL-III is much slower than the other methods, taking taking 73 minutes on average for the highest amount of ILS (species tree height: 0.5×) compared to 5 minutes on average for the lowest amount of ILS (species tree height: 5×). In contrast, both TREE-QMC-n2 and FASTRAL are quite fast, taking on average less than 3 minutes for model conditions with 200 or fewer taxa. Results for methods given true gene trees as input or only 250 (out of 1000) gene trees as input are shown in Supplemental Figures A.10–A.12).

Avian simulated data (48 taxa, 1000 estimated gene trees). Figure 2.4A–C shows the impact of varying the species tree scale and thus the amount of ILS on the avian simulated data sets. wQMC is the least accurate method and is even less accurate than TREE-QMC-n0 (no normalization). Normalization improves the performance of TREE-QMC for these data, enabling TREE-QMC-n2 to be among the most accurate methods when the amount of ILS is high (species tree scales: 0.5× and 1×). Testing for differences between TREE-QMC-n2 versus the other three leading methods (wQFM, FASTRAL, and ASTRAL-III) reveals that either TREE-QMC-n2 is significantly better or else there are no significant differences between these pairs of methods. All methods finish quickly: wQMC and wQFM complete in less than 13 minutes on average, ASTRAL-III completes in less than 4 minutes on average, and the other methods finish in less than 1 minute on average.

Figure 2.4D–F shows the difference in quartet score between estimated and true species trees. We find that most methods typically recover species trees with higher quartet scores than the true species tree, indicating that the true species tree is not the optimal solution to MQSST. Moreover, the relative performance of methods for quartet score is different than the relative performance of methods for species tree error for many model conditions. These two trends are especially pronounced when gene trees are estimated (mean error: 60–62%).

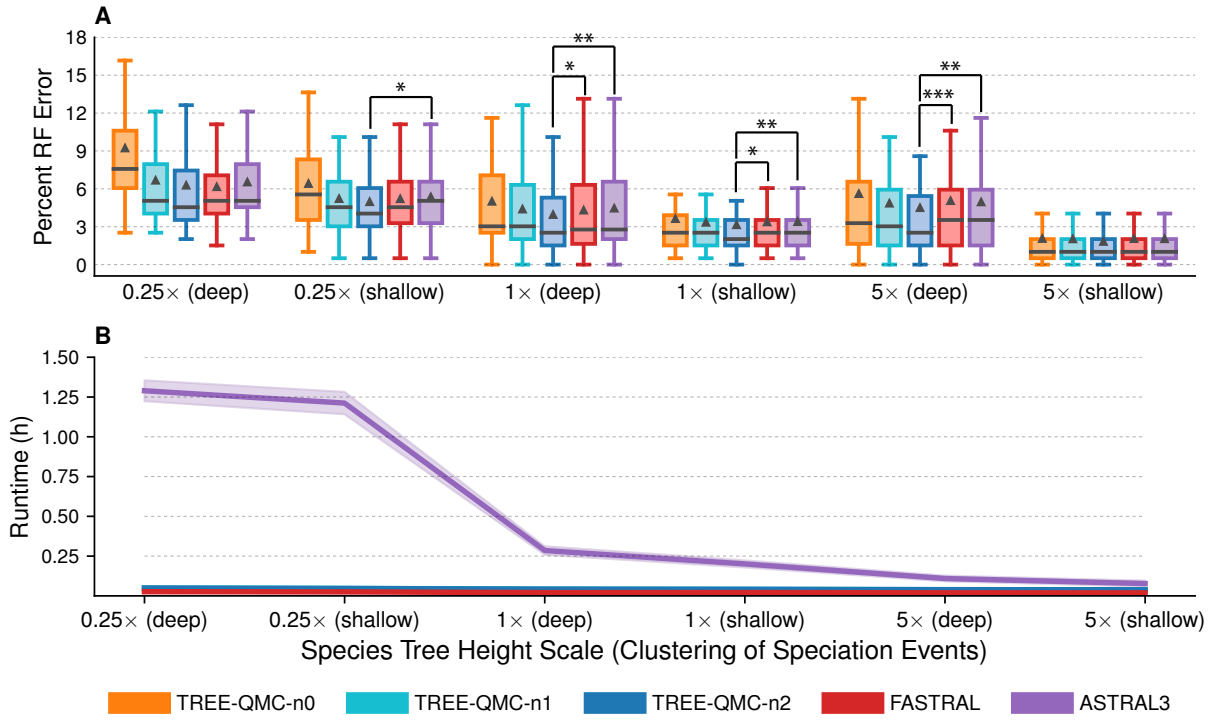


Figure 2.3: Impact of the amount of ILS. (A) Percent species tree error across replicates (bars represent medians; triangles represent means; outliers are not shown). The symbols *, **, and *** indicate significance at $p < 0.05$, 0.005, and 0.0005, respectively (three tests survive multiple comparison corrections; see Supplemental Table A.5 for details). (B) Mean runtime across replicates (shaded region indicates standard error). All data sets have 200 taxa and 1000 estimated gene trees. One model condition with species tree height 1× and shallow speciation is repeated from Figure 2.2. For species tree heights 0.5×, 1×, and 5×, the ILS level is 68–69% , 34%, and 9–21%, respectively, and the GTEE level is 44%, 27%–34%, and 21–28%, respectively.

Mammalian simulated data (37 taxa, 200 estimated gene trees). All methods have similar performance for the mammalian data, although these data sets represent easier model conditions in terms of ILS and GTEE levels (Supplemental Figures A.3 and Supplemental Table A.7).

Gene Tree Estimation Error (GTEE)

Avian simulated data (48 taxa, 1000 gene trees). Figure 2.4A–C also shows the impact of GTEE for each species tree scale (ILS level). For each ILS levels, methods are given true gene trees or estimated gene trees (mean error: 60-62%). The trends for estimated gene trees are discussed

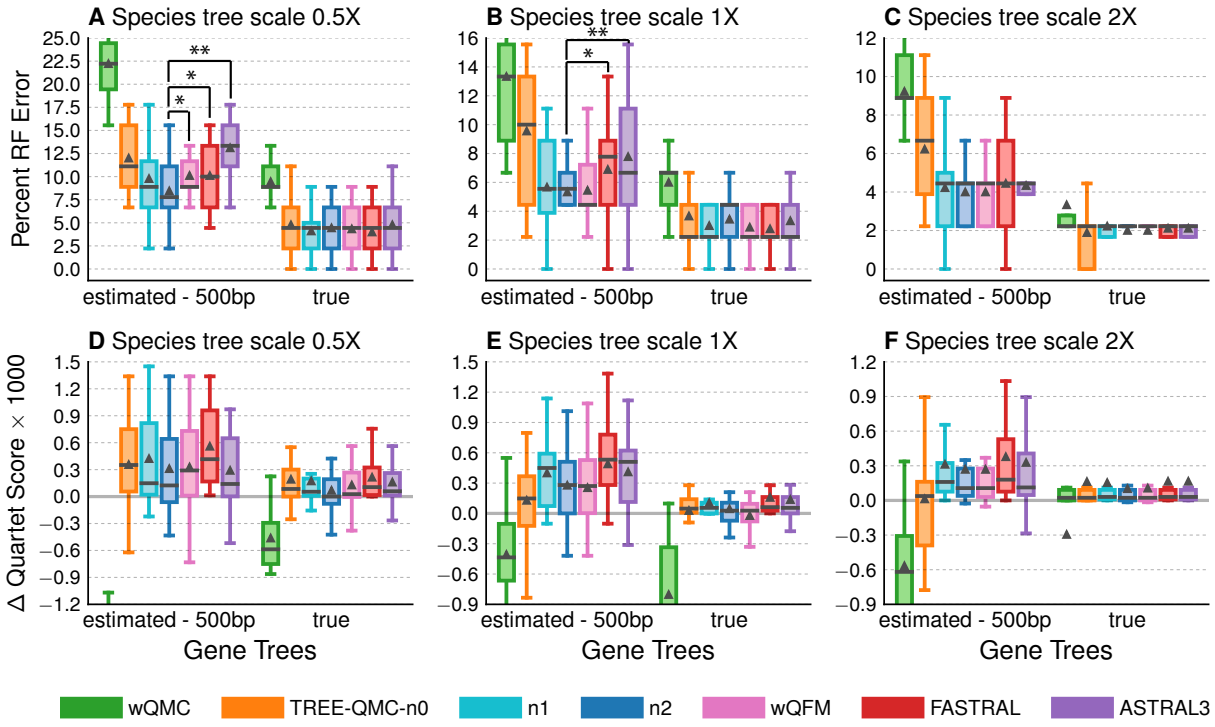


Figure 2.4: Impact of ILS and GTEE. (A), (B), and (C) Percent species tree error for the avian data set with 1000 estimated or true gene trees and species tree scales 0.5×, 1×, and 2×, respectively. Two-sided Wilcoxon-signed ranked tests were used to evaluate differences between TREE-QMC-n2 versus wQFM, FASTRAL, and ASTRAL3 (9 tests per subfigure). The symbols *, **, and *** indicate significance at $p < 0.05$, 0.005, and 0.0005, respectively (for 0.5× species tree scale with estimated gene trees, the difference between TREE-QMC-n2 and ASTRAL-II survives Bonferroni multiple comparison correction; see Supplemental Table A.6 for details). (D), (E), and (F) show the difference in quartet score between the estimated and true species tree times 1000 for species tree scales 0.5×, 1×, and 2×, respectively (positive values indicate the estimated tree is a better solution to MQSST than the true tree). For species tree heights 0.5×, 1×, and 2×, the ILS level is 60% , 47%, and 35%, respectively, and the GTEE level is 60%, 60%, and 62%, respectively. Results for wQMC are cut off because otherwise the trends cannot be observed (see Supplemental Figure A.2 for full y-axes).

above. For true gene trees, there are no significant differences between TREE-QMC-n2 versus the other leading methods (wQFM, FASTRAL, and ASTRAL-III), and all versions of TREE-QMC perform similarly so the utility of normalization is diminished. Moreover, these methods find species trees with similar quartet scores to the true species tree, unlike the case of estimated gene trees. Lastly, the performance of wQMC is in line with the other methods when there is very little gene tree heterogeneity due to ILS or GTEE (Figure 2.4C).

Mammalian simulated data (37 taxa, 200 gene trees). Similar trends between methods are observed for mammalian simulated data sets when varying the sequence lengths (Supplemental Figure A.4 and Supplemental Table A.8). TREE-QMC is significantly more accurate than FASTRAL and ASTRAL-III for the shortest sequence length (250 bp; GTEE level 43%); there are no differences in accuracy between these pairs of methods otherwise.

Number of Genes

Similar trends between methods are observed when varying the number of genes (e.g., Supplemental Figure A.5 and Supplemental Tables A.9–A.10).

2.2.3 Avian phylogenomics data set

We also re-analyze the avian data set from [54] with 3,679 ultraconserved elements (UCEs). This data set includes the best maximum likelihood tree and the set of 100 bootstrapped trees for each UCE. Although the true species tree is unknown, we discuss the presence and absence of strongly corroborated clades, such as Passerea and six of the magnificent seven clades excluding clade IV [13]. We also compare methods to the published concatenation tree estimated by running RAxML [133] on UCEs only [54]; thus the comparison between concatenation and the quartet-based summary methods is on the same data set. Branch support is computed for the estimated species trees using ASTRAL-III's local posterior probability [114] as well as using multi-locus bootstrapping (MLBS) [116]. We repeat this analysis (except MLBS) on the TENT data (14,446 gene trees), which includes gene trees estimated on UCEs as well as exons and introns. In this case, methods are compared to the published TENT concatenation tree estimated by running ExaML [60].

UCE data

For the UCE data (48 taxa, 3679 gene trees), ASTRAL-III complete in 65 minutes, making it the most time consuming method. All other methods run in less than a minute; however, the preprocessing step to weight quartets for wQFM takes 41 minutes.

Both FASTRAL and ASTRAL-III produce the same species tree (Figure 2.5C), and both TREE-QMC-n2 and wQFM produce the same species tree (Figure 2.5A). We compare these two trees to the published concatenation tree for UCEs (Figure 2.5B). There are many similarities between these three trees, as all contain the magnificent seven clades. The TREE-QMC-n2 and FASTRAL trees differ from the concatenation tree by 7 and 9 branches, respectively, putting the TREE-QMC-n2 tree slightly closer to the concatenation tree than the FASTRAL tree. The TREE-QMC-n2 tree recovers Passerea and Afroaves and fails to recover Columbea, like the concatenation tree and unlike the ASTRAL-III tree (note that Passerea was considered to be strongly corroborated, after accounting for data type effects, by 13). Overall, there are only five branches that differ between the TREE-QMC-n2 tree and the FASTRAL tree; all of these branches have nearly equal quartet support for their alternative resolutions so that both trees represent reasonable hypotheses.

TENT data

For the TENT data (48 taxa, 14446 gene trees), TREE-QMC-n2 and FASTRAL complete in less than 3 minutes, whereas it takes 2.35 hours to weight quartets. wQFM completes in less than a minute after this preprocessing phase. We do not run ASTRAL-III as this analysis was reported to take over 30 hours [23].

All three methods produce a different tree, which is compared to the published concatenation tree for TENT data (Supplemental Figure A.6). None of the trees recover Passera, and only the concatenation and wQFM trees recover Afroaves, although this branch has very local support (local posterior probability of 0.0) in the wQFM tree. Once again, the TREE-QMC-n2 and wQFM

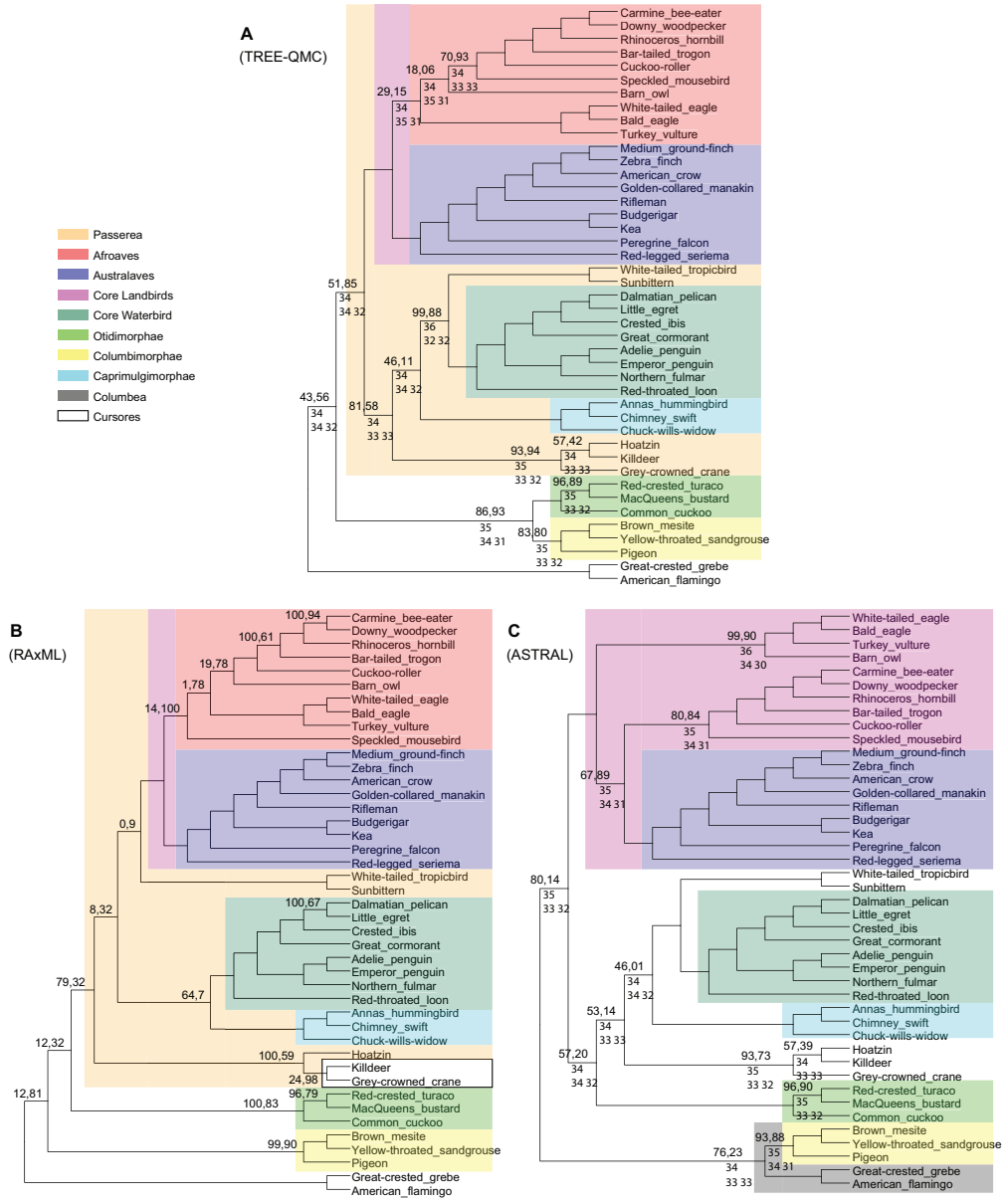


Figure 2.5: Avian UCE data. (A) Tree estimated from UCE gene trees using TREE-QMC-n2/wQFM. (B) Tree estimated from concatenated UCE alignment using RAxML. (C) Tree estimated from UCE gene trees using ASTRAL-III/FASTRAL. Above the branch, we show support values X, Y, where X is estimated using ASTRAL's local posterior probability (multiplied by 100) and Y is computed using RAxML's bootstrap support for subfigure B and using MLBS for subfigures A and C. Support values are only shown when X is less than 100. Below the branch, we show the quartet support (the two values below it correspond to quartet support for the two alternative resolutions of the branch). Taxa outside of Neoaves are not shown as all methods recovered the same topology outside of Neoaves.

trees are closest to the concatenation tree, with the TREE-QMC-n2, wQFM, and FASTRAL trees differing from it by 8, 8, and 10 branches, respectively. There are 5 branches that differ between the wQFM tree and the TREE-QMC-n2 tree (two of these branches in the wQFM have very low support: local posterior probability of 0.03 and 0.0). There are only 3 branches that differ between the TREE-QMC-n2 tree and the FASTRAL tree; as with the UCE data, these branches are reasonable based on quartet support for their alternative resolutions.

2.3 Discussion

Our method TREE-QMC builds upon the algorithmic framework of wQMC [7] by introducing the *normalized* quartet graph and showing that it can be computed directly from gene trees. These contributions together enable our new method TREE-QMC to be highly competitive with the leading quartet-based summary methods in terms of species tree accuracy and empirical runtime, even outperforming them on some simulated data sets. Specifically, TREE-QMC (with non-uniform normalization) is competitive with wQFM in terms of species tree accuracy but scales to much larger data sets. Moreover, TREE-QMC is at least as accurate, and often more accurate, than the dominant method ASTRAL-III (and its improvement FASTRAL), while being highly competitive in terms of empirical runtime.

The model conditions where TREE-QMC outperforms ASTRAL-III are characterized by large numbers of species or high amounts of gene tree heterogeneity due to ILS and GTEE. For the latter scenario, the true species tree was typically not the optimal solution to MQSST (note: this observation is not out of line with the statistical theory because the proof of consistency assumes infinite error-free gene trees). Therefore, better heuristics to MQSST may not translate to more accurate species trees when GTEE is high.

A major goal is to develop summary methods that are robust to GTEE. One approach is weighting quartets not just by their frequency in the input gene trees. A new version of ASTRAL, dubbed

weighted ASTRAL [156], which was published during our study, adjusts quartet weights based on the branch support and branch lengths in the estimated gene trees. TREE-QMC's non-uniform normalization scheme adjusts the quartet weights based on the subproblem division (i.e., quartets are upweighted if they are on species in more closely related subproblems, which ideally reflects closeness in the true species tree). In the future, it would be interesting to compare TREE-QMC to weighted ASTRAL as well as to implement other quartet weighting schemes within TREE-QMC.

There are several other opportunities for future work worth mentioning. First, the version of TREE-QMC presented here requires binary gene trees as input. Thus, TREE-QMC was given gene trees that are randomly refined in our experimental study, whereas all other methods were given gene trees with polytomies. This did not have a negative impact on TREE-QMC's performance relative to the other methods; however, it would be worth exploring this issue further. Ultimately, this inherent limitation of TREE-QMC could be addressed by devising an efficient algorithm for computing the " edges" in the quartet graph (see Methods section), although this would come at the cost of increased runtime. Second, the experimental study presented here only evaluates TREE-QMC in the context of multi-locus species tree estimation where gene tree can be discordant with the species tree due to ILS and/or GTEE. Our study does not address the use of TREE-QMC as a more general quartet-based supertree method, and future work should explore whether quartet weight normalization is beneficial in this context. Lastly, TREE-QMC's algorithm operates on gene trees that are multi-labeled due to artificial taxa, so the algorithms presented here can be applied to gene trees that are multi-labeled due to other causes, such as multiple individuals being sampled per species [101] or genes evolving via duplications [64, 123, 148, 158]. Future work should explore the effectiveness of TREE-QMC under these conditions as well those characterized by missing data due to gene loss or other causes [88].

2.4 Methods

We now present the TREE-QMC method. To begin, we some terminology for phylogenetic trees and the notation used through this section and the Supplemental Materials.

2.4.1 Terminology and Notation

A *phylogenetic tree* T is a triplet $(g, \mathscr{L}, \phi)$, where g is a connected acyclic graph, $\mathscr{L}$ is a set of labels (species), and ϕ maps leaves in g to labels in $\mathscr{L}$. If ϕ is a bijection, we say that T is *singly-labeled*; otherwise, we say T is *multi-labeled*. Trees may be either *unrooted* or *rooted*. Edges in an unrooted tree are undirected, whereas edges in a rooted tree are directed away from the root, a special vertex with in-degree 0 (all other vertices have in-degree 1). To transform an unrooted tree T into a rooted tree T_r, we select an edge in T, sub-divide it with a new vertex r (the root), and then orient the edges of T away from the root. Conversely, we transform a rooted tree T_r into an unrooted tree T by undirecting its edges and then suppressing any vertex with degree 2.

For a tree T, we denote its edge set as $E(T)$, its internal vertex set as $V(T)$, and its leaf set as $L(T)$. Sometimes we consider a phylogenetic tree T *restricted* to a subset of its leaves $R \subseteq L(T)$. Such a tree, denoted $T|_R$, is created by deleting leaves in $L(T) \setminus R$ and suppressing any vertex with degree 2 (while updating branch lengths in the natural way). Henceforth, all trees are *binary*, meaning that non-leaf, non-root vertices (referred to as *internal* vertices) have degree 3.

To present TREE-QMC, we need two additional concepts: *bipartitions* and *quartets*. A bipartition splits a set $\mathscr{L}$ of labels into two disjoint sets: $\mathscr{E}$ and $\mathscr{F} = \mathscr{L} \setminus \mathscr{E}$. Each edge in a (singly-labeled, unrooted) tree T induces a bipartition because deleting an edge creates two rooted subtrees whose leaf labels form the bipartition $\pi(e) = \mathscr{E}|\mathscr{F}$. A given bipartition is displayed by T if it is in the set $\{\pi(e) : e \in E(T)\}$. The bipartition is trivial if $|\mathscr{E}|$ or $|\mathscr{F}|$ is 1; otherwise, it is non-trivial.

A quartet q is an unrooted, binary tree with four leaves a, b, c, d labeled by A, B, C, D, respec-

tively. It is easy to see that there are three possible quartet trees given by their one non-trivial bipartition: $a,b|c,d$, $a,c|b,d$, and $a,d|b,c$ (note that we typically use lower case letters to denote leaf vertices and capital letters to denote leaf labels, although this distinction is only necessary when trees are multi-labeled). A set of quartets can be defined by a unrooted tree T by restricting T to every possible subset of four leaves in $L(T)$; the resulting set $Q(T)$ is referred to as the quartet encoding of T. If T is multi-labeled, then some of the quartets in $Q(T)$ will have multiple leaves labeled by the same label. Lastly, we say that T displays a quartet q if $q \in Q(T)$.

2.4.2 Review of wQMC

As previously mentioned, our new method TREE-QMC builds upon the divide-and-conquer method wQMC [7]. To produce a bipartition on $\mathscr{X}$, wQMC constructs a graph from $\mathscr{Q}$, referred to as the **quartet graph**, and then seeks its maximum cut [7, 124, 125]. The quartet graph is formed from two complete graphs, $\mathbb{B}$ and $\mathbb{G}$, both on vertex set V (i.e., there exists a bijection between V and $\mathscr{X}$). All edges in $\mathbb{B}$ and $\mathbb{G}$ are initialized to weight zero. Then, each quartet $q = A,B|C,D \in \mathscr{Q}_{\mathscr{X}}$ contributes its weight $w_{\mathscr{T}}(q)$ to two "bad" edges in $\mathbb{B}$ and four "good" edges in $\mathbb{G}$, where $w_{\mathscr{T}}(q)$ corresponds to the number of gene trees in the input set $\mathscr{T}$ that display q. The bad edges are based on sibling pairs: (A,B) and (C,D). The good edges are based on non-sibling pairs: (A,C), (A,D), (B,C), and (B,D). We do not want to cut bad edges because siblings should be on the same side of the bipartition; conversely, we want to cut good edges because non-siblings should be on different sides of the bipartition. Ultimately, we seek a cut $\mathscr{C}$ to maximize $\sum_{(X,Y)\in\mathscr{C}}(\mathbb{G}[X,Y] - \alpha\mathbb{B}[X,Y])$, where $\alpha > 0$ is a hyperparameter that can be optimized using binary search. Although MaxCut is NP-complete [56], fast and accurate heuristics have been developed [26]. The cut gives a bipartition in the output species tree and the wQMC method proceeds by recursion on the two subsets of species on each side of the bipartition. Artificial taxa are introduced to represent the species on the other side of the bipartition.

2.4.3 TREE-QMC: Quartet Weight Normalization

Our key observation in developing TREE-QMC is that artifical taxa change the quartet weights so that a single gene tree will vote multiple times for quartets on artificial taxa and only once for quartets on only non-artificial taxa (called singletons). As shown in Figure 2.1, the weight of quartet $M,N|O,P$ is

$$f_0(M,N|O,P) = \sum_{m\in\mathbf{M}}\sum_{n\in\mathbf{N}}\sum_{o\in\mathbf{O}}\sum_{p\in\mathbf{P}} w_{\mathscr{T}}(m,n|o,p) \tag{2.1}$$

where $\mathbf{M} \subset \mathscr{L}$ denotes the set of leaves (i.e., species) in T associated with label M (and similarly for $\mathbf{N},\mathbf{O},\mathbf{P}$). When labels M,N,O,P are all singletons, each gene tree casts exactly one vote for one of the three possible quartets: $M,N|O,P$ or $M,O|N,P$ or $M,P|N,O$ (assuming no missing data). Otherwise, each gene tree casts $|\mathbf{M}|\cdot|\mathbf{N}|\cdot|\mathbf{O}|\cdot|\mathbf{P}|$ votes (again assuming no missing data) and thus can vote for more than one topology.

We propose to normalize the quartet weights so that each gene tree casts one vote for each subset of four labels, although it may split its vote across the possible quartet topologies in the case of artificial taxa. To get this outcome, we can simply divide by the number of votes cast so that the weight of $M,N|O,P$ becomes

$$f_1(M,N|O,P) = \frac{f_0(M,N|O,P)}{|\mathbf{M}|\cdot|\mathbf{N}|\cdot|\mathbf{O}|\cdot|\mathbf{P}|} \tag{2.2}$$

This can be implemented efficiently by assigning an importance value $I(x)$ to each species $x \in \mathscr{S}$ and then compute the weight as

$$f(M,N|O,P) = \sum_{m\in\mathbf{M},n\in\mathbf{N},o\in\mathbf{O},p\in\mathbf{P}} I(m,n,o,p)\cdot w_{\mathscr{T}}(m,n|o,p) \tag{2.3}$$

where $I(m,n,o,p) = I(m)\cdot I(n)\cdot I(o)\cdot I(p)$. Specifically, Equation 2.3 reduces to Equation 2.2 when $I(m) = |\mathbf{M}|^{-1}$ for all $m \in \mathbf{M}$ (and similarly for $\mathbf{N},\mathbf{O},\mathbf{P}$). Because all species with the same

label are assigned the same importance value, we refer to this approach as *uniform normalization (n1)*. More broadly, the quartet weights will be normalized whenever Equation 2.3 corresponds to a weighted average, meaning that

$$\sum_{m\in\mathbf{M}}\sum_{n\in\mathbf{N}}\sum_{o\in\mathbf{O}}\sum_{p\in\mathbf{P}} I(m,n,o,p) = \sum_{m\in\mathbf{M},n\in\mathbf{N},o\in\mathbf{O},p\in\mathbf{P}} I(m,n,o,p) = 1 \tag{2.4}$$

It is easy to see that this will be the case whenever $\sum_{m\in\mathbf{M}} I(m) = 1$ (and similarly for $\mathbf{N},\mathbf{O},\mathbf{P}$). In *unnormalized (n0)* case, we assign all species an importance value of 1 so that Equation 2.3 reduces to Equation 2.1.

We now describe how to normalize quartet weights while leveraging the hierarchical structure implied by artificial taxa by assigning importance values to species with the same label. The idea is that species should have lesser importance each time they are *re-labeled* by an artificial taxon. In Figure 2.1, artificial taxon Z represents species $\mathbf{Z} = \{0,6,7,9\}$ but species 0 and 9 were previously labeled by artificial taxon X. This relationship can be represented as the rooted "phylogenetic" tree T_Z given by newick string: $(6,7,(0,9)X)Z$. We use T_Z to assign importance values to all species $z \in \mathbf{Z}$, specifically

$$I(z) = \prod_{v\in path(T_Z,z)} \frac{1}{outdegree(v)} \tag{2.5}$$

where $outdegree(v)$ is the out-degree of vertex v and $path(T_Z,z)$ contains the vertices on the path in T_Z from the root to the leaf labeled z, excluding the leaf. Continuing the example, $I(6) = I(7) = \frac{1}{3}$ and $I(0) = I(9) = \frac{1}{3}\cdot\frac{1}{2} = \frac{1}{6}$. By construction, $\sum_{z\in\mathbf{Z}} I(z) = 1$ so this approach normalizes the quartet weights. Because different species with the same label can have different weights, we refer to this approach *non-uniform normalization (n2)*. In our simulation study, normalizing the quartet weights in this fashion improved species tree accuracy for challenging model conditions.

2.4.4 TREE-QMC: Efficient Quartet Graph Construction

Another key development in TREE-QMC is an efficient algorithm for constructing the quartet graph directly from k gene trees, each on n species. Our approach breaks down computing the weights for good and bad edges into three cases:

- Case 1: taxa X, Y are both non-artifical taxa (called singletons)
- Case 2: taxa X is a singleton and Y is an artificial taxon (or vice versa)
- Case 3: taxa X, Y are both artificial taxa

For one gene tree, $\mathbb{G}[X,Y]$ and $\mathbb{B}[X,Y]$ can be computed for all pairs X,Y in case 1, case 2, and case 3 in $O(a^2)$ time, $O(abn)$ and $O(b^2n)$ time, respectively, where a is the number of singletons and b is the number of artificial taxa. Thus, for a subproblem with $s = a + b$ taxa, the quartet graph can be constructed in $O(s^2nk)$ time by applying this algorithm to all gene trees (Theorem A.1 in the Supplemental Materials).

After quartet graph construction, we seek a max cut. Our high-level approach is the same as wQMC. However, we differ in our binary search for α (interval and precision) and our max-cut heuristic, instead using the one proposed by [16] and implemented in the open-source package MQLib by [26]. Our approach for cutting the quartet graph runs in $O(s^2)$ time (Theorem A.2 in the Supplemental Materials), so the time complexity for each subproblem is dominated by quartet graph construction. Overall, the divide-and-conquer algorithm runs in $O(n^3k)$ time (Theorem A.3 in the Supplemental Materials) if the division into subproblems is perfectly balanced. We do not expect subproblem division to be perfectly balanced in practice, and moreover the time complexity analysis hides large constant factors (Theorem A.2 in the Supplemental Materials). That being said, we find TREE-QMC is sufficiently fast, at least on the specific inputs in our study.

Idea behind Efficient Quartet Graph Construction

We now provide the idea behind our approach by considering the simplest case where there are no artifical taxa in gene tree T with n leaves (i.e., T is singly-labeled). For TREE-QMC, the weight of bad edges between taxa X and Y, denoted $\mathbb{B}[X,Y]$, is the number of quartets displayed by T with X,Y as siblings and similarly for $\mathbb{G}[X,Y]$ but non-siblings. This means that we can easily compute the weight of the good edges if given the weight of the bad edges by applying $\mathbb{G}[X,Y] = \binom{n-2}{2} - \mathbb{B}[X,Y]$.

To compute $\mathbb{B}$ efficiently, we observe that there is exactly one leaf associated with label X (denoted x) and one leaf associated with label Y (denoted y), so there is a unique path connecting leaves x and y in T (Figure 2.6a). Deleting the edges on this path (and their end points) produces a forest of K rooted subtrees, denoted $\{t_1, t_2, \ldots, t_K\}$. Let w and z be two leaves of subtrees t_i and t_j, respectively. Then, T displays quartet $x,w|z,y$ for $i < j$, quartet $x,y|w,z$ for $i = j$, and quartet $x,z|w,y$ for $i > j$. To summarize, x,y are siblings if and only if leaves w,z are in the same subtree off the path from x to y. It follows that $\mathbb{B}[X,Y]$ can be computed by considering all ways of selecting two other leaves from the same subtree for all subtrees on the path from x to y.

This observation can be used to count the quartets efficiently when gene trees are singly-labeled. However, we need to be more careful when T is multi-labeled, which is typically the case due to artificial taxa. Following our example, suppose that we want to count the number of bad edges between 0 and 17 contributed by the subtree with leaves 4, 5, and 6. However, if leaves 4 and 5 are both re-labeled by artificial taxon M, the quartet on $0,17|4,5$ corresponds to quartet $0,17|M,M$ has no topological information and should not be counted. The other quartets $0,17|4,6$ and $0,17|5,6$ correspond to $0,17|M,6$ and thus should be counted.

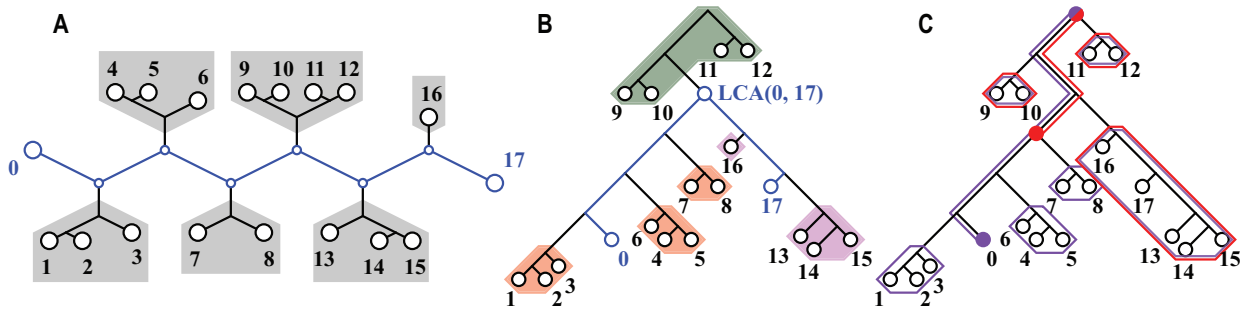


Figure 2.6: To count the quartets induced by T with 0 and 17 as siblings, we consider the path between them (shown in blue in (a)). The deletion of the path produces 6 rooted subtrees (highlighted in grey). Because 0 and 17 are siblings in a quartet if and only if the other two taxa are drawn from the same subtree, the number of bad edges can be computed as $\binom{3}{2}+\binom{3}{2}+\binom{2}{2}+\binom{4}{2}+\binom{3}{2}+\binom{1}{2}=16$. Here we show how to compute the number of quartets induced by T with 0 and 17 as siblings after rooting T arbitrarily. Subfigure (b) shows that we need to consider the number of ways of selecting two taxa from the same subtree for three cases: (1) the subtree above the $lca(0,17)$ (highlighted in green), (2) all subtrees off the path from the $lca(0,17)$ to the left taxon 0 (highlighted in red), and (3) all subtrees off the path from the $lca(0,17)$ to the right taxon 17 (highlighted in pink). Case 1 can be computed in constant time if we know the number of leaves below the LCA, that is, $\mathbb{A}[0,17]=6$ (Eq. 2.8). Cases 2 and 3 can also be computed in constant time as follows. Subfigure (c) shows the prefix of the left child of the $lca(0,17)$, denoted $p[lca(0,17).left]$ is the number of ways of selecting two taxa from the same subtree for all subtrees circled in red, which are off the path from the root to this vertex. Similarly, the the prefix of taxon 0, denoted $p[0]$, is the number of ways of selecting two taxa from the same subtree for all subtrees circled in blue, which are off the path from the root to 0. Therefore, the number of ways of selecting two taxa from all subtrees in case 2 (i.e., subtrees highlighted in red in subfigure (b)) is $\mathbb{L}[0,17]=p[0]-p[lca(0,17).left]=7$ (Eq. 2.9). Case 3 can be computed in a similar fashion as $\mathbb{R}[0,17]=p[17]-p[lca(0,17).right]=3$ (Eq. 2.10). Putting this all together gives $\mathbb{B}[0,17]=16$ (Eq. 2.6).

Algorithm for computing the bad edges given a singly-labeled gene tree

We now present an algorithm for computing the number of bad edges given a singly-labeled gene tree T (later we will extend it to the more general case of a multi-labeled gene tree). After rooting T arbitrarily, we again consider the path between x and y, which now goes through their lowest common ancestor, denoted $lca(x,y)$ (Figure 2.6b). This allows us to break the computation into three parts

$$\mathbb{B}[X,Y]=\mathbb{A}[X,Y]+\mathbb{L}[X,Y]+\mathbb{R}[X,Y] \tag{2.6}$$

where $\mathbb{A}[X,Y]$ is the number of ways of selecting two leaves from the subtree above $lca(x,y)$, $\mathbb{L}[X,Y]$ the number of ways of selecting two leaves from the same subtree for all subtrees off the path from $lca(x,y)$ to leaf in it's *left* subtree (say x), and $\mathbb{R}[X,Y]$ the number of ways of selecting two leaves from the same subtree for all subtrees off the path from $lca(x,y)$ to the leaf in its *right* (say y). As we will show, each of these quantities can be computed in constant time, after an $O(n)$ preprocessing phase, in which we compute two values for each vertex v in T. The first value $c[v]$ is the number taxa below vertex v. The second value $p[v]$, which we refer to as the "prefix" of v, is the number of ways to select two taxa from the same subtree for all subtrees off the path from the root to vertex v (Figure 2.6c). It is easy to see that c can be computed in $O(n)$ time via a post-order traversal. After which, p can be computed in $O(n)$ via a preorder traversal, setting

$$p[v] = p[v.parent] + \binom{c[v.sibling]}{2} \tag{2.7}$$

after initializing $p[root] = 0$. Now we can compute the quantities:

$$\mathbb{A}[X,Y] = \binom{n - c[lca(x,y)]}{2} \tag{2.8}$$

$$\mathbb{L}[X,Y] = p[x] - p[lca(x,y).left] \tag{2.9}$$

$$\mathbb{R}[X,Y] = p[y] - p[lca(x,y).right] \tag{2.10}$$

where $v.left$ denotes the left child of v and $v.right$ denotes the right child of v (see Figure 2.6c). It is possible to access $lca(x,y)$ in constant time after $O(n)$ preprocessing step [42], although we implemented this implicitly by computing the entries of $\mathbb{B}$ during a post-order traversal of T. Thus, we can compute $\mathbb{B}$ in $O(n^2)$ time, provided that T is singly-labeled.

Algorithm for computing the bad edges given a multi-labeled gene tree

We now present an algorithm for computing the number of bad edges $\mathbb{B}[X,Y]$ given a multi-labeled gene tree T. As previously mentioned, this breaks down into three cases. Case 1 (where taxa X,Y are both singletons) is presented below and cases 2 and 3 are presented in the Supplemental Materials.

As before, we focus on the number of ways to select two leaves w,z from a collection of subtrees; however, now that T is multi-labeled, it is possible for two leaves w,z to have the same label. Therefore, we now need to count the number of ways to select two leaves z,w below vertex u so that they are **uniquely labeled** $Z \neq W$ (note that we use capital letters W and Z to denote the current labels of leaves w and z, respectively). This modified binomial is computed by revising the preprocessing phase. We now let $c_0[v]$ denote the number of leaves labeled by singletons below vertex v and let $c_D[v]$ denote the number of leaves labeled by artificial taxon D below vertex v. Thus, for each vertex v, we store a vector $c[v]$ of length $b+1$, where b is the number of artificial taxa in T. As before, we can compute c in $O(bn)$ time via a postorder traversal. However, the number of ways to select two leaves with different labels is now broken into three cases:

1. the number of ways to select two singletons, which equals $\binom{c_0[v]}{2}$,
2. the number of ways to select one singleton and one artificial taxa, which equals $c_0[v] \cdot \sum_{D \in \mathscr{A}(v)} c_D[v]$, where $\mathscr{A}(v)$ is the set of artificial taxa below vertex v, and
3. the number of ways to select two artificial taxa, which equals $\sum_{D \neq E \in \mathscr{A}(v)} c_D[v] \cdot c_E[v]$.

Putting this all together gives the **modified binomial coefficient**:

$$g_0[v] = \binom{c_0[v]}{2} + c_0[v] \cdot G_1[v] + \frac{G_1[v]^2 - G_2[v]}{2} \tag{2.11}$$

where $G_1[v] = \sum_{D \in \mathscr{A}(v)} c_D[v]$ and $G_2[v] = \sum_{D \in \mathscr{A}(v)} c_D[v]^2$. At each vertex, the calculation of $G_1[v]$

and $G_2[v]$ takes $O(b)$ time, after which we can compute $g_0[v]$ in constant time. Thus, g_0 can be computed in $O(bn)$ time. Note that we also need to compute modified binomial coefficient for the subtree "above" vertex v, denoted $g_0[v.above]$. This can be computed in a similar fashion by noting that number of singletons above v is $a - c_0[v]$ and that the number of leaves above v labeled by each artificial taxon D is $|\mathbf{D}| - c_D[v]$.

Using the modified binomial, we can apply our algorithm for singly-labeled trees by redefining prefix sum:

$$p_0[v] = p_0[v.parent] + g_0[v.sibling] \tag{2.12}$$

and then redefining the quantities from which we can compute $B[x,y]$ in constant time, that is, $\mathbb{A}[X,Y] = g_0[lca(x,y).above]$, and $\mathbb{L}[X,Y] = p_0[x] - p_0[lca(x,y).left]$, and $\mathbb{R}[X,Y] = p_0[y] - p_0[lca(x,y).right]$. As there are a^2 pairs of singletons in the subproblem, the total runtime is $O(a^2 + bn)$.

Algorithm for normalizing quartet weights while computing the bad edges

To normalize the quartet weights, $\mathbb{B}[X,Y]$ becomes the *weighted* sum of quartets with X,Y are siblings, where each quartet $x,y|z,w$ is weighted by $I(x,y,z,w) = I(x)I(y)I(z)I(w)$, where $I(x)$ is the importance value assigned to leaf x (which corresponds to a species in the singly-labeled gene tree). When X,Y are singletons,

$$\mathbb{B}[X,Y] = I(x)I(y) \sum_{\substack{w,z \in L(T): Z \neq W \neq X \neq Y, \\ q(x,y,z,y) = x,y|z,y}} I(z)I(w) \tag{2.13}$$

where the importance values of singletons are set to 1 so we know that $I(x) = I(y) = 1$. Note that all of the importance values are set to 1 in the unnormalized case.

To compute the normalized version of $\mathbb{B}[X,Y]$ using the previous algorithm, we set $c_D[v]$ to be the sum of the importance values of the leaves below v that are labeled by D (i.e., $c_D[v] = \sum_{m \in L(v), M = D} I(m)$ where $L(v)$ denotes the set of leaves below v). The proof of correctness follows

from Lemma 2.1, in which we show that the total weight of selecting two uniquely labeled leaves below vertex u equals $g_0[u]$. This is because all other quantities $(p, \mathbb{A}, \mathbb{L}, \mathbb{R})$ are computed from $g_0[u]$.

Lemma 2.1. *The total weight of all taxon pairs in the subtree rooted at internal vertex u*

$$\sum_{\substack{z,w \in L(u): \\ Z \neq W}} I(z)I(w) = g_0[u] \tag{2.14}$$

where $L(u)$ is the set of leaves below vertex u.

See Supplemental Materials for proof.

Lastly, we need to compute the good edges $\mathbb{G}[X,Y]$, which is the total weight of quartets in which X,Y are not siblings. This can be done in constant time, following Lemma 2.2.

Lemma 2.2. *Let T be a multi-labeled gene tree, and let X,Y be singletons. Then,*

$$\mathbb{G}[X,Y] + \mathbb{B}[X,Y] = \binom{c_0[r]-2}{2} + (c_0[r]-2) \cdot G_1[r] + \frac{G_1[r]^2 - G_2[r]}{2} \tag{2.15}$$

where r is the root vertex of T.

See Supplemental Materials for proof.

This concludes our treatment of case 1, in which X,Y are both singletons. In order to compute all entries of $\mathbb{B}$ and $\mathbb{G}$, we also need to consider the other two cases. In case 2, X is a singleton and Y is an artificial taxon or vice versa (Supplemental Figure A.13), and in case 3, both X and Y are artificial taxa (Supplemental Figure A.14). These cases are more complicated because the naive approach would consider all paths in the tree between a leaf labeled X and a leaf labeled Y, which is not efficient. The algorithms and proofs for these cases are provided in the Supplemental Materials.

Acknowledgments

YH and EKM thank the reviewers for feedback that greatly improved this quality of our work.

Funding

This research was funded by the State of Maryland. All computational experiments were performed on the Center for Bioinformatics and Computational Biology (CBCB) compute cluster at the University of Maryland, College Park.

Software and Data Availability

TREE-QMC source code is available on Github: `https://github.com/molloy-lab/TREE-QMC`. Data sets are available on Dryad: `https://doi.org/10.5061/dryad.m0cfxpp6g`.

Chapter 3: Weighted TREE-QMC: Improved Robustness to Gene Tree Incompleteness, Estimation Errors, and Systematic Homology Errors

This chapter has been published in Systematic Biology *[47]. Supplementary materials referenced in this chapter are freely available on* Zenodo *(`https://doi.org/10.5281/zenodo.13852123`) and reproduced in Appendix B.*

3.1 Introduction

Over the last decade summary methods have been widely adopted in species tree estimation pipelines [134, 135]. These pipelines begin with identification of orthologous genomic regions (called loci) from target capture sequencing data and/or assembled genomes. Next, an evolutionary history for each locus (referred to as a gene tree) is reconstructed typically in two steps: first a multiple sequence alignment (MSA) is built and second a phylogeny is estimated from the MSA under some model of molecular sequence evolution (e.g., the GTR model; 138). Finally, the estimated gene trees are given as input to summary methods, many of which are designed to reconstruct the species tree while appropriately accommodating gene tree discordance due to incomplete lineage sorting (ILS), unlike concatenation [110]. Examples of such summary methods include MP-EST [67] and STELAR [53], which are based on triplets implied by rooted gene trees, NJst [66] and ASTRID [140], which are based on pairwise internode distances implied by unrooted gene trees, and ASTRAL [77] and TREE-QMC [45], which are based on quartets implied by unrooted gene trees.

Despite the wide spread use of summary methods, it is increasingly recognized that their accuracy depends on gene tree quality. Three well-established issues are incompleteness, estimation error, and homology errors. A gene tree is incomplete if it is missing one or more species, which may be due to gene loss or data processing issues. Gene tree estimation error (GTEE) refers to inaccuracies arising during phylogeny estimation, for example because of model misspecification or low phylogenetic signal. Homology errors are mistakes in determining shared evolutionary history. They can occur at the gene level due to the inclusion of unrelated sequences or paralogous sequences (when working with single-copy orthologs); they can also occur at the site level due to MSA errors. Both types of homology errors can result in inaccurate gene trees [121, 130]. In simulations, summary methods typically decrease in accuracy as GTEE or incompleteness increases [78, 81, 88, 147]. Moreover, a growing number of systematic studies show that homology errors at the site or gene level can impact species tree estimates produced by different summary methods [121, 130].

The dominant approach for combating poor quality gene trees is *gene tree filtering*, in which entire gene trees are removed from the data set typically based on some threshold for the proportion of missing taxa [21, 51] and/or proxies for GTEE [120]. However, simulation studies have suggested that filtering can reduce the accuracy of summary methods and that only gene trees with very high GTEE should be removed entirely [81]. This finding motivates the use of less aggressive approaches, for example contracting very low support branches in gene trees [119, 157] and/or removing specific taxa from individual gene trees. Taxa are typically removed based on the fraction of gapped sites in the gene MSA (referred to as fragmentary missing data by 115 and the 89) or based on (long) outlier branch lengths in the gene tree [70, 89]. Critically, these approaches require substantial effort and decision-making on the part of researchers, who must choose how to quantify error as well as select thresholds for filtering gene trees, contracting branches, and/or removing individual taxa. Often a range of parameter settings are explored, leading to many different estimates of the species tree (e.g., 51).

To address GTEE, [156] proposed to leverage gene tree branch lengths and support values within the popular summary method ASTRAL. ASTRAL seeks a species tree that maximizes the number of quartets (unrooted four-leaf trees) implied by the input gene trees. This optimization problem is NP-hard [62], so ASTRAL solves the problem within a constrained search space built from the input gene trees [77, 78, 157]. The goal of weighted ASTRAL is to *down-weight quartets with low support on the internal branch* (an indicator that the quartet cannot be confidently resolved due to insufficient phylogenetic signal) and/or *long lengths on the terminal branches* (an indicator that the quartet may be impacted by long branch attraction). The former is referred to as *support weighting*, and the latter is referred to as *length weighting*. When support and length weighting are used together, it is referred to as *hybrid weighting*. These weighting schemes increase the complexity of ASTRAL's original search algorithm. To achieve greater computational efficiency, weighted ASTRAL employs heuristic search based on phylogenetic placement. This new approach, often referred to as ASTRAL-IV or ASTER, has been found to be more robust to incomplete gene trees than the original ASTRAL algorithm [86, 156].

Following the success of weighted ASTRAL-IV, [65] incorporated the length and support weighting schemes (but not the hybrid weighting scheme) into the distance method ASTRID. Overall, leveraging gene tree branch lengths and support values is a promising path forward; however, weighted summary methods are still in their infancy. Comparatively little is known about how they perform on real data, and there are significant challenges to developing efficient algorithms.

In this paper, we introduce weighted TREE-QMC, a new weighted summary method that integrates the length, support, and hybrid weighting schemes of [156] into the Quartet Max Cut framework (QMC) of [124] while also efficiently addressing the issue of quartet weight normalization for incomplete gene trees. A previous challenge in using QMC as a summary method was that it required researchers to explicitly extract quartets from gene trees prior to species tree estimation; we recently addressed this issue, introducing TREE-QMC [45]. Weighting quartets based on gene tree branch lengths and support values presents computational challenges. Perhaps sur-

prisingly, we achieve an algorithm with only a small increase in time complexity and no increase in storage complexity compared to unweighted TREE-QMC. Moreover, our simulation study showed that weighted TREE-QMC was fast and highly competitive with weighted ASTRAL-IV in terms of species tree accuracy, even outperforming weighted ASTRAL-IV on some challenging simulation conditions, such as large numbers of taxa. We also evaluated weighted summary methods on three recently published data sets: one for plants with high rates of missing taxa [86] and two for birds, *Palaeognathae* [19] and *Neognathae* [145], both of which have been found to contain homology errors [121, 130]. Overall, our results suggest that TREE-QMC is efficient, easy-to-use, and improves robustness to gene tree incompleteness, estimation error, and homology errors.

3.2 Materials and Methods I. Weighted TREE-QMC

3.2.1 Overview of unweighted TREE-QMC

To introduce weighted TREE-QMC, we briefly review the original (unweighted) method [45], which leverages the Quartet Max Cut (QMC) framework [7, 124, 125]. The idea behind QMC is to reconstruct the species tree in a divide-and-conquer fashion from a set of input quartets. At each step of the divide phase, a graph is built with vertices representing taxa. The edges between vertices (taxa) are weighted by the relevant input quartets. A cut of the **quartet graph** partitions the taxa into two disjoint subsets, corresponding to a bipartition (associated with an internal branch) in the unrooted species tree. The algorithm continues by recursion on subproblems defined by the taxa on each side of the bipartition. Termination occurs when there are three or fewer taxa in the subproblem as the solution (i.e., a phylogenetic tree on the taxon set) is trivial (Fig. 3.1a). Importantly, **artificial taxa** are added to subproblems at each step in the divide phase to represent the taxa on the other side of the bipartition (Fig. 3.1a). During the conquer phase, subproblem trees are connected at artificial taxa, ultimately producing an unrooted, binary species tree on the complete set of taxa (Fig. S1 in 45).

The (unweighted) TREE-QMC method improves upon QMC in two ways. First, QMC operates on quartets, which must be extracted from gene trees prior to species tree estimation. Given k gene trees on n taxa, the preprocessing phase alone has time complexity $\Omega(n^4k)$ and storage complexity $\Omega(n^4)$ (note that we do not include the storage required for input gene trees). In contrast, TREE-QMC builds the quartet graph directly from the input gene trees, having time complexity $O(n^3k)$, with some assumptions on subproblem decomposition, and storage complexity $O(n^2 + b_{max}^2 n)$, where b_{max} is the maximum number of artificial taxa for any subproblem. To summarize, the time complexity of TREE-QMC is lower than the quartet extraction step required for QMC, and the reduction in storage complexity should further improve efficiency by reducing cache misses. More importantly, our experimental evaluation showed TREE-QMC scaled to large data sets (with 1000 taxa and 1000 genes on which we could not run QMC) and was faster than ASTRAL-III, the leading method at the time (Fig. 2B in 84).

Second, TREE-QMC normalizes quartet weights with respect to artificial taxa so that each gene tree gets one vote for every subset of four taxa present in the tree, although the vote could be split across the three possible quartets when gene trees are multi-labeled by artificial taxa introduced from the QMC framework, multiple gene copies [64], or multiple individuals/alleles per species [101]. TREE-QMC can be run without normalization (**n0**), with uniform normalization (**n1**), or with non-uniform normalization (**n2**), which leverages the structure implied by the introduction of artificial taxa to *downweight quartets that are less relevant for resolving a particular subproblem*, corresponding to an internal branch in the species tree (Fig. 3.1a–d). Normalization was critical for accurate species tree estimation under challenging simulation conditions, such as high mean GTEE, with the n2 normalization scheme yielding the best accuracy (Fig. 4A in 84). Notably, TREE-QMC (with n2 normalization) outperformed ASTRAL-III [157] on challenging model conditions, ranging from large numbers of taxa (200–1000) to very high levels of ILS (60%) and mean GTEE (60%) (Figs. 2 and 4A in 45, respectively).

3.2.2 Normalizing Quartet Graph for Incomplete Gene Trees

Our first improvement to TREE-QMC addresses the issue of incomplete gene trees, specifically their impact on quartet weight normalization. As previously mentioned, we say the weights are normalized if each gene tree votes once for each subset of four taxa present in the tree. We show that the normalization factors derived for the complete taxon set do not constitute a valid normalization scheme for incomplete gene trees (Fig. 3.1d–f). Moreover, **sharing** these normalization factors across all gene trees reduces species tree accuracy in our simulation study. In Section B.1 of the Supplementary Materials, we introduce an efficient algorithm for computing the correct normalization values for each gene tree on the fly. This new approach does not increase the storage or time complexity of quartet graph construction and reduces the number of memory allocations and thus cache misses compared to earlier versions of the TREE-QMC code. Lastly, TREE-QMC now outputs a polytomy when there are no quartets for the subproblem; this can occur when the majority of taxa are missing from gene trees.

3.2.3 Weighting Quartet Graph based on Gene Tree Branch Lengths and Support Values

Our second improvement to TREE-QMC addresses the issue of inaccurate gene trees by leveraging the quartet weighting schemes of [156].

Recall that the vertices in the quartet graph represent taxa from the subproblem and edges between taxa are weighted based on quartets implied by the input gene trees, with leaves mapped to taxa by the function L. Then, each quartet $q = x,y|z,w$ implied by some gene tree T contributes two **bad edges** between each of the sibling pairs (i.e., $(L(x),L(y))$ and $L(z),L(w)$) as well as four **good edges** between each of the non-sibling pairs (i.e., $(L(x),L(z))$, $(L(x),L(w))$, $(L(y),L(z))$ and $(L(y),L(w))$), provided that the leaves are uniquely labeled (i.e., $L(x) \neq L(y) \neq L(z) \neq L(w)$). In

the unweighted algorithm, all good and bad edges have weight one. Using the weighting schemes of [156] (described in the introduction), the **length**, **support**, and **hybrid** weights of the quartet q implied by gene tree T are defined as

$$\mathtt{W_l}(q) = \mathtt{exp}\left(\sum_{\substack{e \in x,y \to u, \\ z,w \to v}} -\mathtt{l}(e) \right), \tag{3.1}$$

$$\mathtt{W_s}(q) = 1 - \prod_{e \in u \to v} (1 - \mathtt{s}(e)), \text{ and} \tag{3.2}$$

$$\mathtt{W_h}(q) = \mathtt{W_l}(q) \cdot \mathtt{W_s}(q) \tag{3.3}$$

respectively, where $\mathtt{l}(e)$ denotes the length of branch e in gene tree T, $\mathtt{s}(e)$ denotes the support value of branch e in T, and u and v denote the **anchor** vertices of quartet $q = x,y|z,w$ in T, with u closer to sibling pair x,y and v closer to sibling pair z,w. Note that $u \to v$ is the set of edges on the path between u and v, which corresponds to the **internal branch** of quartet q implied by T (Fig. 3.2a–b; also see Fig. 1 in 156). Similarly, $x,y \to u$ and $z,w \to v$ is the set of edges on the paths from each of the four leaf vertices to their respective anchor, which corresponds to the four terminal branches of quartet q implied by T.

Figure 3.2 shows the impact of weighting schemes on the quartet graph, taking two gene trees from the Palaeognathae data set [19] as input, one of which is impacted by gene-level homology errors [121]. For simplicity, we show the quartet graph for the Tinamou species, finding that the three weighting schemes successfully down-weight the quartet implied by the gene tree with homology errors, which in turn impacts the best cut(s) and the resulting Tinamous tree.

The naive way to construct the quartet graph is to extract all weighted quartets from the gene trees (Fig. B.1 in the Supplementary Materials); however, this preprocessing step is computationally intensive and requires a large amount of storage even for unweighted quartets, as previously mentioned. The challenge is building the quartet graph efficiently (i.e., directly from gene trees without extracting all possible quartets) while weighting quartets based on gene tree branch

lengths and support values. In Section B.2 of the Supplementary Materials, we introduce efficient algorithms to construct the weighted quartet graph for a subproblem directly from gene trees in $O((a^2b + ab^2n + b^3n)k)$ time and $O(a^2 + ab + b^2n)$ space, where a is the number of non-artificial taxa from the subproblem (referred to as **singletons**), b is the number of artificial taxa from the subproblem, and n is the total number of species, which we assume corresponds to the maximum number of leaves in any gene tree for simplicity (Theorem B.4 in the Supplementary Materials). Thus, weighted graph construction requires an additional factor of $O(b)$ compared to building the unweighted graph. After constructing the quartet graph, the only work remaining for the subproblem is seeking a max cut. The heuristic we use to seek a max cut [26] does not exceed the time complexity of quartet graph construction (Theorem 2 in 45). Lastly, we need to account for the subproblems being solved within the divide-and-conquer framework. This brings the overall time complexity of weighted TREE-QMC to $O(n^4k)$ assuming the subproblems are produced in a perfectly balanced fashion (Theorem B.5 in the Supplementary Materials), although see the discussion for an explanation of why this worst-case analysis does not reflect empirical performance. The overall space complexity of weighted TREE-QMC is $O(n^2 + b_{max}^2 n)$, same as the unweighted method.

3.2.4 New features in weighted TREE-QMC

Lastly, we implement several new features within weighted TREE-QMC.

Bioconda. TREE-QMC is now installable via bioconda.

Multi-labeled and non-binary gene trees. TREE-QMC now extends its framework for handling artificial taxa to accommodate multi-labeled gene trees as well as non-binary gene trees being given as input. Multi-labeled means that the input gene trees can have multiple leaves labeled by the same species (e.g., due to sampling multiple individuals per species or multiple gene copies).

This option is not evaluated in this work.

Characters and "BP" mode. TREE-QMC now allows character data to be given as input; see the `--chars` and `--char2tree` options. TREE-QMC treats each character (i.e., site) in the data set as an unrooted tree with internal branches separating taxa assigned the same state from all other taxa. This functionality is useful because quartet-based summary methods are statistically consistent estimators of species trees for binary (0/1) characters evolving under the neutral Wright-Fisher model [74, 84] even when there is error and missingness provided it is unbiased [46]. [131] refer to this approach as summary methods in "bipartition mode" or "BP mode". If the `--bp` option is used with TREE-QMC instead of the `--chars` option, branch lengths in coalescent units will be computed under the neutral Wright-Fisher model with a fast maximum likelihood estimator, assuming a constant mutation rate [84]; otherwise, branch lengths are not computed for character data. These options are not evaluated in this work.

Quadrapartition Quartet Support (QQS). TREE-QMC now enables users to output Quadrapartition Quartet Support (QQS) for the three quartet topologies "around" each internal branch (quadripartition) for unweighted or weighted quartets [80, 114]; see the `--support` or `--supportonly` options. Additionally, maximum likelihood branch lengths in coalescent units are reported, along with the effective number (**EN**) of gene trees that provide information (i.e., quartets) for resolving the branch. Typically a branch is considered well supported when EN is sufficiently large and discordance follows signatures of ILS (i.e., the QQS value correspond to the branch in the estimated species tree is higher than the QQS values for the two alternative branch resolutions, which should be roughly equal to each other). Note that QQS and EN values are used by ASTRAL to compute branch support as the local posterior probability (**Local PP**) [114].

Partitioned Coalescence Support (PCS). TREE-QMC now enables users to compute Partitioned Coalescence Support (PCS) for a specified focal branch in an input species tree [40, 41].

Specifically, the `--pcsonly` option returns the QQS values for each gene tree for each of the three possible resolutions of the focal branch in the species tree, enabling users to identify outlier gene trees that strongly or weakly favor a particular resolution of the focal branch compared to other gene trees. If applied to characters or gene trees sorted by their position in the genome, the output QQS plots can highlight interesting regions of the genome, for example a region of suppressed recombination in the avian tree of life (Fig. 2 in 80) as well as enrichment for deep coalescence at the MHC locus in primates (Fig. PanGenomeS1 in 150).

3.3 Materials and Methods II. Simulation Study

We now describe our performance study for evaluating the utility of weighted TREE-QMC on data sets simulated in prior studies focusing on gene tree estimation error and incompleteness; specifically, we used the ASTRAL-II [78], ASTRAL-III [157], and Asteroid [86] simulated data sets.

3.3.1 Species Tree Estimation Methods

A large number of summary methods have been developed over the last decade. To make the simulation study manageable, we focused on **weighted summary methods**: weighted ASTRAL-IV and weighted ASTRID. We also included Asteroid [86], as it is similar to (unweighted) ASTRID but with improvements for incomplete gene trees. All summary methods were run in default mode with the parameter settings summarized below. Detailed repository information, version/commit numbers, and software commands are available in the Supplementary Materials.

For the Asteroid data sets, summary methods were run in unweighted mode, as the estimated gene trees did not include branch support and no alignments were available. Additionally, the Asteroid data sets had high rates of missing taxa, so we ran TREE-QMC on simulated data sets with and without updating the (n2) normalization values for incomplete gene trees (Fig. 3.1). The

latter is denoted as **n2 shared** to indicate the same normalization values are shared across all gene trees regardless of their taxon sets.

For the ASTRAL-II/III data sets, summary methods were run with their best weighting option. We ran ASTRAL-IV with hybrid weights (denoted **ASTER-IV-wh**), as this weighting scheme yielded the best accuracy in [156]. Motivated by these prior results, we ran TREE-QMC with hybrid weights (denoted **TREE-QMC-wh** or **TQMC-wh**) and in unweighted mode for comparison. ASTRID does not implement the hybrid weighting scheme; we ran ASTRID with support weights (denoted **ASTRID-ws**), as this weighting scheme yielded the best accuracy in [65]. **Asteroid** does not implement any weighting schemes; thus we ran Asteroid in default (unweighted) mode. For all weighted summary methods, we set the minimum/maximum values for gene tree branch support values to 0.333/1 for data sets with abayes support and 0/100 for data sets with bootstrap support, as indicated in the user manuals. On the smaller simulated data sets (i.e., Asteroid and ASTRAL-III), we ran TREE-QMC with each of its three quartet weight normalization schemes: **n0** (none), **n1** (uniform), and **n2** (non-uniform). The non-uniform (n2) yielded the best accuracy, so we used it for the larger (i.e., ASTRAL-II) simulated data sets and for biological analyses.

3.3.2 Simulated Data Sets

We evaluated summary methods using 3719 data sets simulated in three prior studies. These simulations were conducted by (1) simulating species trees under the Yule model [155], (2) simulating gene trees within the species tree under the Multi-Species Coalescent (MSC) [102], which models ILS, and (3) simulating sequences down gene trees under standard models of molecular sequence evolution (e.g., the GTR model; 138), which produces an MSA because there are no insertions or deletions. Different model conditions were created by varying a one or two model parameters at at time while keeping the others fixed. Multiple replicate data sets were simulated for each model condition. To enable comparisons of model conditions across studies, we empiri-

cally evaluated ILS, GTEE, and missingness (MISS) for each data set, reporting the average values across data sets with the same model condition. **ILS** was evaluated as the normalized Robinson-Foulds (RF) distance between the true species tree and true gene trees, averaged across all gene trees. **GTEE** was evaluated as the normalized RF distance between each true and estimated gene tree, averaged across all gene trees. **MISS** was evaluated as the fraction of missing taxa from each gene tree, averaged across all gene trees. We summarize the model conditions and empirical properties below.

Asteroid simulated data sets. The Asteroid data sets were simulated by [86] to evaluate Asteroid in the context of missing data. After deleting sequences from gene MSAs, they estimated gene trees with maximum likelihood (ML) under the GTR+GAMMA4 model [149] using ParGenes [85]. The empirical properties of the Asteroid data sets are given in Table B.1 in the Supplementary Materials. Overall, the data sets were characterized by very high missingness (0.76–0.84), very low ILS (0.05–0.08), and medium GTEE (0.30–0.45), with some exceptions depending on the model parameter being varied. Model conditions (50 replicates each) were simulated by varying the

1. **effective population size** and thus ILS level: 10 (ILS: 0), **50 million (default; ILS: 0.06)**, 100 million (ILS: 0.11), 500 million (ILS: 0.39), and 1 billion (ILS: 0.56)
2. **number of taxa**: 25, **50 (default)**, 75, 100, 125, 150
3. **number of genes**: 250, 500, **1000 (default)**, and 2000
4. **sequence length** and thus GTEE level: 50 bp (GTEE: 46), **100 bp (GTEE: 0.35; default)**, 200 bp (GTEE: 0.25), and 500 bp (GTEE: 0.16)
5. **gene tree branch length scalar** and thus GTEE level: 0.05 (GTEE: 0.55), 0.1 (GTEE: 0.47), **1 (GTEE: 0.35; default)**, 10 (GTEE: 0.42), 100 (GTEE: 0.70), and 200 (GTEE: 0.77)

6. **missingness parameter**: 0.5 (MISS: 0.74), 0.55 (MISS: 0.78), **0.6 (MISS: 0.81; default)**, 0.65 (MISS: 0.84), 0.7 (MISS: 0.86), and 0.75 (MISS: 0.88)

where default indicates the value used in all other model conditions. There are 26 unique model conditions, after removing duplicates from the default parameters, and 1300 replicate data sets in total.

ASTRAL-III (S100) simulated data sets. The ASTRAL-III datasets with 100 taxa were simulated by [157] to evaluate ASTRAL-III in the context of GTEE. They estimated gene trees with maximum likelihood under the GTR+GAMMA model using FastTree-2 [100]. After which, branch support was estimated via bootstrapping with 100 replicates; later [156] estimated abayes branch support [5] with IQ-TREE-2 [76]. The empirical properties of the ASTRAL-III (S100) data sets are given in Table B.2 in the Supplementary Materials. Overall, the data sets were characterized by medium-high ILS (0.46). Four model conditions (50 replicates each) were generated by varying the sequence length and thus the GTEE level: 200 bp (GTEE: 0.56), 400 bp (GTEE: 0.42), 800 bp (GTEE: 0.31), and 1600 bp (GTEE: 0.23). The number of genes given to methods as input varied from 50, 200, 500, and 1000, yielding 16 model conditions and 800 replicate data sets in total.

ASTRAL-II simulated data sets. The ASTER-II data sets (no missingness) were simulated by [78]. We estimated abayes branch support for these data sets (see Supplementary Materials for software command), following the recommendation by [156]. The empirical properties of the ASTRAL-III data sets are in Table S3 of the Supplementary Materials. Model conditions (50 replicates each) were simulated by varying the species tree height and speciation rate with a fixed number of taxa (200), the latter, in particular, varied ILS and GTEE. At the shortest species tree height (called 0.5×), the ILS/GTEE were 0.68/0.69 and 0.44/0.44, respectively. At the longest species tree height (called 5×), the ILS/GTEE were 0.09/0.21 and 0.28/0.21, respectively. Con-

versely, the speciation rate and species tree height was fixed and the number of taxa was varied: 50, 100, 200, 500, 1000 (ILS: 0.31–0.35; GTEE: 0.26–0.30). The number of genes given to methods as input varied from 50, 200, and 1000 genes, yielding 33 model conditions and 1640 replicate data sets (note that 33 data sets were excluded based on criteria described in the Supplementary Materials).

3.3.3 Evaluation Metrics

We compared summary methods in terms of species tree error, runtime, quartet score, and mean branch support.

Species tree error. We reported the normalized Robinson-Foulds (RF) error rate [107], which is equivalent to the false negative (FN) or false positive (FP) error rates when both the true and estimated species trees were binary (note that the FP/FN error rate are defined as the number of branches in the estimated/true species tree that are missing from the true/estimated species tree divided by the number of branches in the estimated/true species tree). All methods returned binary trees, except TREE-QMC, which returned non-binary trees for some of the Asteroid data sets with high missingness. Polytomies were refined arbitrarily before computing RF error (as well as quartet score; see below) to ensure comparisons across methods were interpretable (polytomies were not refined prior to computing FN and FP error).

For each model condition (with replicate data sets simulated under the same conditions), we tested for significant differences between some pairs of methods using two-sided, paired Wilcoxon signed-rank tests, as implemented in the R coin library [52]. To be conservative, we corrected for ties between methods in two different ways: the Wilcoxon method [144] and the Pratt method [99], taking the higher of the two p-values. Lastly, we corrected for multiple comparisons based on Bonferroni correction, dividing the p-value by the number of tests performed for the experiment (note that we treated data sets simulated for the ASTRAL-II, ASTRAL-III (S100), and Asteroid studies

as three different experiments). If the p-value was less than 0.05 after Bonferroni correction, we reported a **significant difference** between two methods for the model condition.

Runtime. We recorded the wall clock time (i.e., the total time for the method to run on an input data set from start to finish). When collecting runtime data, we gave all methods exclusive access to the compute node (architecture: AMD EPYC-7313 with 32 CPUs; maximum RAM: 64 GB). ASTRAL-IV did not complete within our maximum wall clock time of 20 hours for three data sets when given a single-thread, so we also ran it with 16 threads. Otherwise, methods were run with a single thread. Branch support calculations were turned off to enable fair runtime comparisons.

Quartet Score and Branch Support Values. For the quartet-based methods (i.e., TREE-QMC and ASTRAL-IV), we reported the total number of quartets in the input gene trees that were satisfied by each estimated species tree as well as mean branch support. Branch support was computed using ASTRAL's Local PP [114], which is based on QQS values from either unweighted or weighted quartets. Both total quartet score and mean branch support were computed with the same quartet weighting scheme as was used for species tree estimation.

3.4 Results on Simulated Data Sets

We now report the results of benchmarking (weighted) summary methods on three collections of data sets simulated for prior studies.

Asteroid Data Sets. The Asteroid data sets were used to evaluate unweighted summary methods under very high levels of missing taxa (see Figure 3.3a–d as well as Figures B.3–B.8 and Tables B.4–B.11 in the Supplementary Materials). Updating normalization factors to account for missing taxa across gene trees improved species tree (RF) error; specifically, it increased the accuracy of TREE-QMC (n2) in 86% of 1300 data sets compared to sharing factors across all gene trees

(Fig. 2a). Moreover, TREE-QMC (n2) achieved significantly lower FN and FP error compared to no normalization (n0) for 17 and 16 of the 26 model conditions, respectively (Table S11). There were no significant differences between non-uniform (n2) and uniform (n1) normalization factors (Table S10); thus, we compared TREE-QMC (n2) against the other summary methods.

In comparison to ASTRID and ASTRAL-IV, TREE-QMC (n2) outperformed or tied with them in terms of RF error on more than 75% of data sets (better on 74% and 62% of data sets and tied on 9% and 14% of data sets, respectively) (Fig. 3.3b–c). TREE-QMC (n2) achieved significantly lower FN and FP error compared to ASTRID for 18 of the 26 model conditions (Table S6–S8). Likewise, TREE-QMC (n2) achieved significantly lower FP error than ASTRAL-IV on 17 of the 26 model conditions (Table S6–S8); this dropped to just 3 model conditions when considering FN error. Although ASTRAL-IV was less accurate than TREE-QMC (n2), ASTRAL-IV achieved a higher total quartet score on 97% of data sets. The two methods performed similarly in terms of mean branch support (Fig. 3.3d).

Asteroid was the only method that outperformed TREE-QMC. In terms of RF error, Asteroid was better on 46% of data sets, TREE-QMC (n2) was better on 34%, and the two methods tied on the remaining 20% (Fig. B.2 in the Supplementary Materials). Asteroid also achieved significantly lower FN error than TREE-QMC (n2) for 3 of the 26 model conditions (Table S6); Asteroid was not significantly better than TREE-QMC for any model conditions when considering FP error.

3.4.1 ASTRAL-III (S100) data sets

The ASTRAL-III (S100) data sets were used to evaluate weighted summary methods under varying levels of GTEE (see Figure 3.3e–h as well as Figure S9 and Tables B.12–B.15 in the Supplementary Materials). We compared weighted methods given gene trees with abayes support values, as bootstrap support values resulted in worse species tree accuracy for all methods, as previously observed by [156]. The hybrid weighting scheme improved the accuracy of TREE-

QMC. Compared to unweighted method, TREE-QMC-wh (n2) lowered and tied for species tree (RF) error on 57% and 27% of 800 data sets, respectively (Fig. 3.3e). Moreover, TREE-QMC-wh (n2) typically achieved the lowest mean species tree error per model condition compared to using the n0 or n1 normalization schemes—as well all other summary methods (Fig. S9 and Table S12).

In comparison to ASTRID-ws and ASTRAL-IV-wh, TREE-QMC-wh (n2) achieved significantly lower RF error in 4 and 3 of the 16 model conditions, respectively (Tables S13–S14); in contrast, ASTRID-ws and ASTRAL-IV-wh were not significantly more accurate than TREE-QMC-wh (n2) for any model conditions. Looking at individual data sets, TREE-QMC-wh (n2) outperformed or tied with ASTRID-ws and ASTRAL-IV-wh on more than 75% (better on 53% and 44% of data sets and tied on 26% and 35% of data sets, respectively) (Fig. 3.3f–g). However, TREE-QMC-wh (n2) never achieved a higher quartet score than ASTRAL-IV-wh and had better mean branch support on just 5% of data sets (Fig. 3.3h). Note Asteroid should perform the same as (unweighted) ASTRID, as there are no missing taxa and that ASTRID performed worse than ASTRID-ws on these data sets in the study by [65].

3.4.2 ASTRAL-II data sets

The ASTRAL-II data sets were used to evaluate weighted summary methods under varying levels of ILS and numbers of taxa (see Figure 3.3i–l as well as Figures B.10–B.11 and Tables B.16–B.23 in the Supplementary Materials). The trends for these data sets were similar to the ASTRAL-III (S100) data sets. Compared to the unweighted method, TREE-QMC-wh (n2) lowered and tied for species tree (RF) error in 58% and 24% of 1607 data sets, respectively (Fig. 3.3i). TREE-QMC-wh (n2) also outperformed or tied with ASTRID-ws and ASTRAL-IV-wh on more than 75% of data sets (better on 54% and 46% of data sets and tied on 30% and 37%, respectively) (Fig. 3.3j–k). Moreover, TREE-QMC (n2) achieved significantly lower RF error than ASTRID-ws and ASTRAL-IV-wh on 16 and 10 of the 33 model conditions, respectively (Tables S19–S20 and

S22–S23). The difference between these pairs of methods was the most pronounced for model conditions with 500 and 1000 taxa, where TREE-QMC had a clear advantage. However, TREE-QMC-wh (n2) only achieved a higher quartet score than ASTRAL-IV-wh on just 1 of the 1619 data sets and had better mean branch support on just 5% of data sets (Fig. 3.3l).

[78] also published the concatenation (CA-ML) trees (except the 1000-taxon, 1000-gene model condition). In comparison to CA-ML TREE-QMC (n2) was significantly better in terms of RF error on 19 of the 33 model conditions (Tables S18 and S21). The only condition CA-ML had a notable advantage over TREE-QMC (n2) was when the species tree height was longer (in generations) and speciation occurred closer towards the root (Table S18), although it is worth noting that the ILS level for these data sets was only 9%.

3.4.3 Runtime

Lastly, we evaluated the runtime of summary methods on the ASTRAL-II data sets with varying numbers of taxa and 1000 gene trees. As the number of taxa increased from 100 to 1000 (factor of 10), the ratio between the runtime of hybrid weighted and unweighted TREE-QMC (n2) only increased from 2.9 to 3.4 (factor of 1.17) (Fig. B.16 in the Supplementary Materials). At 1000 taxa, the fastest method was ASTRID-ws, followed by ASTRAL-IV-wh (16 threads), TREE-QMC (n2), TREE-QMC-wh (n2), and then ASTRAL-IV-wh (1 thread) at 0.0, 0.35, 0.36, 1.24, and 8.00 hours, respectively. Thus, although TREE-QMC-wh (n2) runtime was slower than the original (i.e., unweighted) algorithm, it still scaled to large numbers of taxa, where it had the greatest advantage in terms of species tree accuracy compared to ASTRID-ws and ASTRAL-IV-wh.

3.5 Biological Analyses

We now describe our re-analysis of three published biological data sets using summary methods Asteroid, ASTRID, TREE-QMC, and ASTRAL-IV. Software commands are similar to those used

in the simulation study.

3.5.1 Plant data set from Morel *et al.,* (2023)

We first re-analyzed the 81-taxa, 6176-gene plant data set curated by [86] to evaluate their method Asteroid. This data set was characterized by extreme missing data, with the majority of gene trees have 9 or fewer taxa. Specifically, the number of gene trees with 30–35 taxa (MISS: 57–63%), 20-29 taxa (MISS: 64–75%), 10-19 taxa (MISS: 77–88%), and 4–9 taxa (MISS: 89–95%) was 4, 95, 490, and 5587, respectively. Overall, the mean percentage of missing taxa across the 6176 gene trees was 95%. Because of the extreme levels of missing data, we used the plant data set to evaluate the impact of gene tree incompleteness on unweighted summary methods, similar to the Asteroid simulated data sets. After species tree estimation, we estimated branch support using ASTRAL's Local PP [114].

We compared the estimated species trees to the (non-binary) reference tree created by combining the relationships among major clades from [89] with the relationships among angiosperms from [136] and then resolving subfamilies and genera (Fig. B.13 in the Supplementary Materials). In comparison to the reference, the (binary) concatenation tree from [86] was missing 14 (26%) internal branches. The ASTRID, ASTRAL-IV, Asteroid, and TREE-QMC trees failed to recover 28 (52%), 23 (43%), 14 (26%), and 14 (26%) of internal branches from the reference (Fig. 3.4). Asteroid and TREE-QMC were closest to the concatenation tree differing by 14 and 15 branches, respectively, whereas ASTRAL-IV and ASTRID were farthest differing by 20 and 29 branches, respectively. All 14 branches missing from the TREE-QMC tree were missing in the other estimated trees with two exceptions: (1) concatenation and ASTRAL-IV recovered Chlorophyta and (2) concatenation recovered the clade uniting Monocots and Eudicots (Fig S13). ASTRID and ASTRAL-IV missed several groups recovered by the remaining methods, including Angiosperms, Monocots, Rosids, and lower order relationships, like the order Poales. It is also worth noting that

ASTRAL-IV separated Chlorophyta from Rhodophyta by all of Monocots, which formed a highly imbalanced (caterpillar-like) topology. This branching order for Monocots differed from all other methods and was highly supported (Local PP); however, the effective number (EN) of gene trees with information around some of these branch was quite low (e.g., 1–49), in which case ASTRAL recommends ignoring them. Lastly, the ASTRAL-IV tree satisfied 3790, 4815, and 26141 more quartets than TREE-QMC, Asteroid, and ASTRID trees (note that the fraction of quartets satisfied was above 0.89 for all methods).

3.5.2 Avian (Neognathae+Palaeognathae) data set from Wu *et al.,* (2024)

Next, we re-analyzed the avian data set curated by [145], which included 5756 coding sequences (CDs), 4871 introns, and 2384 intergenic segments for 124 avian species. Homology errors in CDs were identified by [130], although these errors did not appear biased towards any particular set of taxa (Fig. 1B in 130 and Fig. 1A in 146). We used IQ-TREE-2 (version 2.3.5) to compute abayes support on the best maximum likelihood (ML) gene trees estimated by [145] and then estimated species trees using Asteroid, ASTRID, TREE-QMC, and ASTRAL-IV (with branch support calculations were turned off to enable fair runtime comparisons). Species tree estimation was restricted to the 10,627 CDs and introns, as these markers may be more robust to orthology errors than intergenic regions [130]. All three weighted summary methods were run with and without their different weighting schemes. For unweighted analyses only, we also filtered the input gene trees by removing outlier taxa with TreeShrink version 1.3.9 (options: -m “per-species” -q 0.05) [70]. We confirmed that TreeShrink successfully removed the homology errors from the CDs highlighted in Figure 1B in [129]. After species tree estimation, we estimated branch support as Local PP using the same quartet weighting scheme that was used for species tree estimation [114]. For weighted methods, we used the original gene trees when computing Local PP and QQS values; for unweighted methods only, we used gene trees with outlier taxa removed.

Comparison of unweighted versus weighted summary methods. We first evaluated the stability of trees produced by the same summary method but with different parameter settings or inputs. TREE-QMC produced two species trees: one for hybrid and length weighting (Fig. 3.5c) and the other for all other ways of running TREE-QMC (Fig. B.14b in Supplementary Materials). The two TREE-QMC trees differed on just three branches. ASTRID also produced two species trees: one tree for length weighting and one tree for all other ways of running ASTRID (Fig. 3.5b). The two ASTRID trees differed on 18 internal branches, with the length weighting tree being much farther from the NJst tree (thus we did not consider the length weighting ASTRID tree in further comparisons). Asteroid recovered the same tree as ASTRID if taxon filtering was used; otherwise, the Asteroid tree differed from the ASTRID tree by four branches. Lastly, ASTRAL-IV produced four species trees. Support weighting and no weighting minus taxon filtering produced the same tree. Length weighting differed from no weighting plus taxon filtering by just one internal branch (Fig. B.14d in Supplementary Materials). The two ASTRAL-IV trees farthest apart differed by six branches (hybrid weighting versus support weighting / no weighting minus taxon filtering).

Comparison to published species trees. [145] published two main species trees: one from applying NJst [66] (Fig. 3.5a and Fig. 2 in 145) and one from applying RAxML [133]. The NJst and RAxML trees, which differed by 13 internal branches, were given different inputs than our methods; they were given all three marker types (CDs, introns, and intergenetic regions) after removing 30% outlier genes (i.e., genes with the highest quartet distance between the best ML gene tree and the estimated species tree were removed). In comparison to the NJst (and RAxML) trees, ASTRID/Asteroid, TREE-QMC-wh (n2), and weighted ASTRAL-IV-wh by 8 (16), 8 (13), and 16 (18) branches, respectively. Thus, the TREE-QMC tree was closer to NJst and RAxML trees than the ASTRAL-IV tree by 8 and 5 branches, respectively.

When considering major clades, all methods tested in our study recovered *Palaeognathae/-Neognathae*, *Galloanserae/Neoaves*, *Aequorlitornithes*, and *Hieraves*. Unlike ASTRID and TREE-

QMC, ASTRAL-IV did not recover *Aquaterraves* or *Telluraves* (because of its placement of the two pigeons) or *Columbaves*. No methods recovered *Listusilvae*, *Coraciiorphae*, or *Australaves*, but the ASTRID and TREE-QMC trees required fewer edit moves (either contractions/refinement moves or subtree prune and regraft moves) to achieve monophyly for these groups than the ASTRAL-IV tree. On the other hand, ASTRAL-IV always achieved the highest quartet score and its branches that disagreed with NJst or RAxML had high branch support (Local PP), although the QQS values around these branches were all close to one third, which is consistent with a rapid radiation for Neoaves, as demonstrated by many prior studies [54, 137, 145]. Lastly, ASTRAL-IV recovered a highly imbalanced (i.e., caterpillar-like) branching pattern for *Aquaterraves* minus the two pigeons, which was striking compared to the other estimated species trees.

Runtime. ASTRID completed in just a few seconds, weighted TREE-QMC completed in 11 minutes, weighted ASTRAL-IV-wh completed in 2 hours, and ASTRAL-IV completed in 2.6 hours (both with and without taxon filtering the input using TreeShrink). The wall clock time of ASTRAL-IV (with or without weighting) dropped to 5–8 minutes when using 16 threads instead of one thread.

3.5.3 Avian (Palaeognathae) data set from Cloutier *et al.,* (2019)

Lastly, we re-analyzed the avian (Palaeognathae) data set curated by [19], with 3158 ultra-conserved elements (UCEs) for 15 species. Gene-level homology errors were identified in 105 of the UCEs by [121]; these errors were systematic as they consistently impacted two taxa: White-Throated Tinamou and Chicken, which clustered together (Fig 3.2b). We used IQ-TREE-2 to compute abayes support on the best maximum likelihood (ML) gene trees estimated by [19] and then estimated species trees with the four summary methods, comparing weighting schemes as well as filtering practices for unweighted methods. All analyses produced one of two trees (referred to as "A" and "B") which differed by a single focal branch (quadrapartition). In tree A,

the focal branch split Tinamous plus Kiwis+Cassowary+Emu from Rheas plus Ostrich+Chicken, whereas in tree B, Tinamous swapped with Rheas so they were on the same side of the focal branch as Chicken (Fig. B.15 in Supplementary Materials). The 105 UCEs with homology errors strongly favored tree B, but the hybrid quartet weighting scheme reduced the magnitude of support for tree B, as shown by partitioned coalescent support (PCS) analysis around the focal branch (Fig. B.16 in Supplementary Materials).

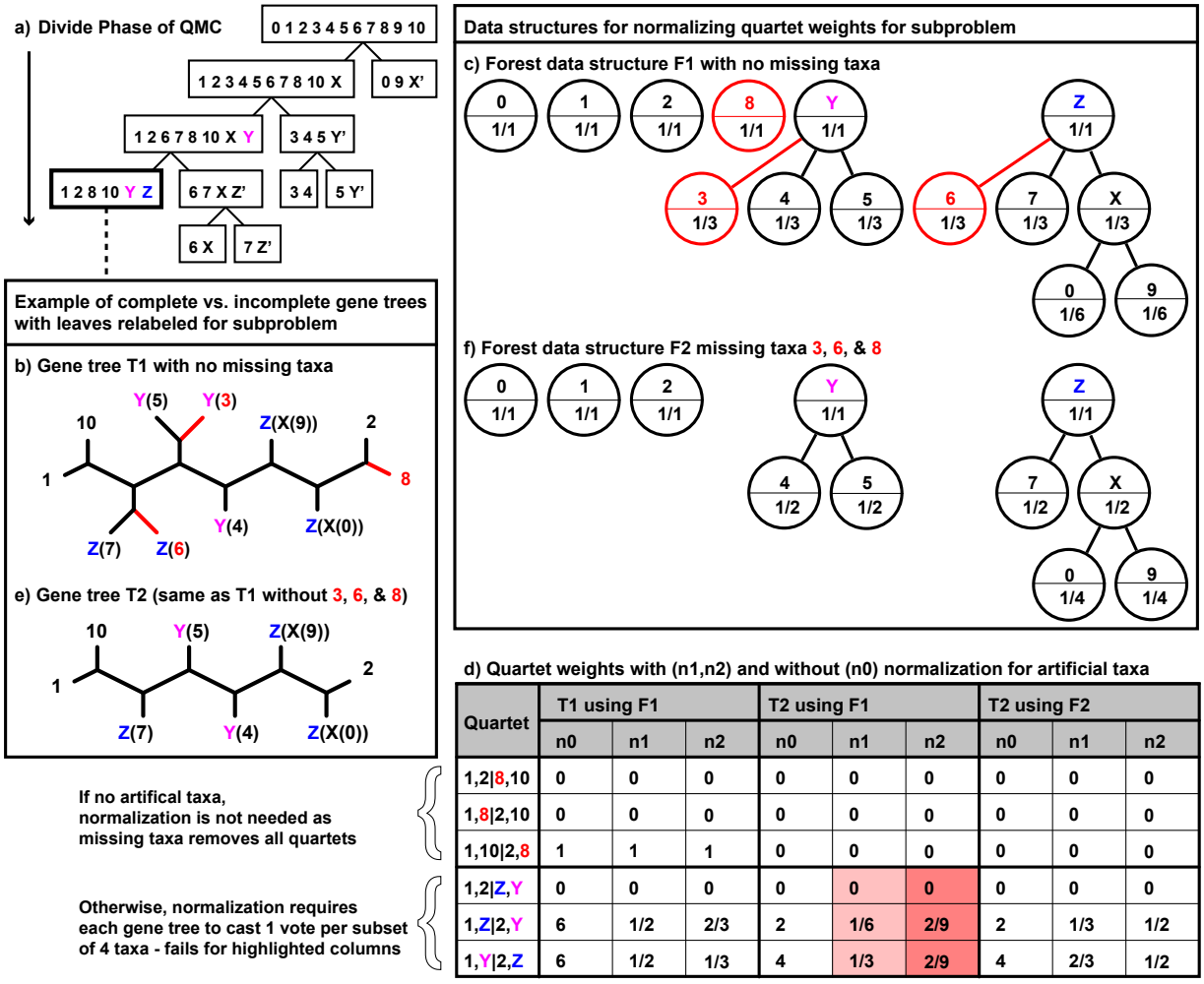
Weighted TREE-QMC and ASTRAL (hybrid and length) returned tree A, as did unweighted TREE-QMC and ASTRAL after filtering the UCE data set (either removing the two impacted taxa from the 105 gene trees with homology errors or removing the gene trees entirely). Otherwise, tree B was returned. Branch support (i.e., Local PP) for the two resolutions of the focal branch was low ($<$0.5; see Fig. S15). Filtering taxa or entire gene trees resulted in tree A satisfying 2507 and 2269 more quartets than tree B respectively. Likewise, tree A satisfied 972.5 more quartets than tree B when using hybrid weights; otherwise it satisfied 1783 fewer quartets. The concatenation trees from UCEs estimated by [19] and [129] (who removed homology errors) resolved the focal branch in the same way as tree A but differed from tree A on one branch, placing Emu and Cassowary as sister to Tinamous instead of Kiwis.

3.6 Discussion

Recently, [156] introduced quartet weighting schemes to improve the robustness of the popular method ASTRAL to gene tree estimation error. Although their quartet weighting schemes are straightforward to describe, they present challenges for developing efficient summary methods. Here, we show that these weighting schemes can be effectively integrated into the Quartet Max Cut framework of [124], introducing weighted TREE-QMC. Our theoretical and empirical study shows that weighted TREE-QMC is a highly promising approach and may even be the leading summary method to date for challenging conditions, like gene tree incompleteness, gene tree estimation

error, and homology errors.

Gene tree incompleteness. Quartet weight normalization was an important algorithmic development from the unweighted TREE-QMC method. Here, we showed that incomplete gene trees require their own normalization factors to be computed, which means that normalization cannot be correctly performed after quartets have been extracted from gene trees, as required by QMC or related algorithms. We introduced a more efficient algorithm for computing the correct normalization factors for each gene tree and found in simulations that using the correct normalization improved the accuracy of TREE-QMC when gene trees were incomplete. For these simulated data sets, TREE-QMC outperformed ASTRID and ASTRAL-IV and was highly competitive with Asteroid. We found similar results for the plant data set, on which TREE-QMC successfully recovered many established clades in the reference tree similar to Asteroid and concatenation, an approach often favored by systematists despite its theoretical limitations when ILS is high. The ASTRAL-IV tree, in contrast, was farther from the reference tree and the concatenation tree, displaying a highly imbalanced (caterpillar-like) branching pattern towards the root. Although this topology was better supported (Local PP) and achieved a higher quartet score, our results on simulated data sets did not suggest that higher mean branch support and/or total quartet score are always indicators of greater species tree accuracy. This may seem unexpected given consistency results for missing data [88]; however, these guarantees make assumptions about the patterns of missing data and hold in the limit of infinite gene trees, whereas our empirical analyses use a finite number.



d) Quartet weights with (n1,n2) and without (n0) normalization for artificial taxa

Quartet	T1 using F1			T2 using F1			T2 using F2		
	n0	n1	n2	n0	n1	n2	n0	n1	n2
1,2|8,10	0	0	0	0	0	0	0	0	0
1,8|2,10	0	0	0	0	0	0	0	0	0
1,10|2,8	1	1	1	0	0	0	0	0	0
1,2|Z,Y	0	0	0	0	0	0	0	0	0
1,Z|2,Y	6	1/2	2/3	2	1/6	2/9	2	1/3	1/2
1,Y|2,Z	6	1/2	1/3	4	1/3	2/9	4	2/3	1/2

If no artifical taxa, normalization is not needed as missing taxa removes all quartets

Otherwise, normalization requires each gene tree to cast 1 vote per subset of 4 taxa - fails for highlighted columns

Figure 3.1: **Normalization for artificial taxa.** **(a)** The divide-and-conquer algorithm starts with a set of input gene trees on 11 taxa. Taxa are split into two disjoint subsets at each subproblem generated during the divide phase. Take the subproblem on taxon set $\{1,2,8,10,Y,Z\}$ as an example, where X, Y, and Z are artificial taxa representing $\{0,9\}$, $\{3,4,5\}$, and $\{6,7,X\}$, respectively. **(b)** A complete gene tree $T1$ with leaves relabeled for the subproblem. **(c)** Forest data structure $F1$ for the subproblem. **(d)** Quartets on $\{1,2,Y,Z\}$ implied by gene tree $T1$ have weights summing to 1 after normalization if using forest data structure $F1$ to compute importance values. **(e)** An incomplete gene tree $T2$ with leaves relabeled for the subproblem (note that $T2$ is the same as $T1$ but missing taxa 3, 6, and 8). Two quartets $(1,7|2,5$ and $1,7|2,4)$ in gene tree $T2$ correspond to $1,Z|2,Y$ after relabeling for the subproblem. If using $F1$ importance values for n2 normalization, $\mathtt{I}(1,7,2,5) = \mathtt{I}(1,7,2,4) = 1 \cdot 1/3 \cdot 1 \cdot 1/3 = 1/9$. For n1 normalization, the importance values are $\mathtt{I}(1,7,2,5) = \mathtt{I}(1,7,2,4) = 1 \cdot 1/4 \cdot 1 \cdot 1/3 = 1/12$. Thus, the total weight of quartet $1,Z|2,Y$ is $1/6$ and $2/9$ after n1 and n2 normalization, respectively. Repeating this process for the other two topologies $1,2|Z,Y$ and $1,Y|2,Z$ does not produce weights that sum to 1 (shaded columns). **(f)** Forest data structure $F2$ for subproblem but updated to exclude taxa 3, 6, and 8. Quartets on $\{1,2,Y,Z\}$ implied by gene tree $T2$ have weights summing to 1 after normalization if using $F2$.

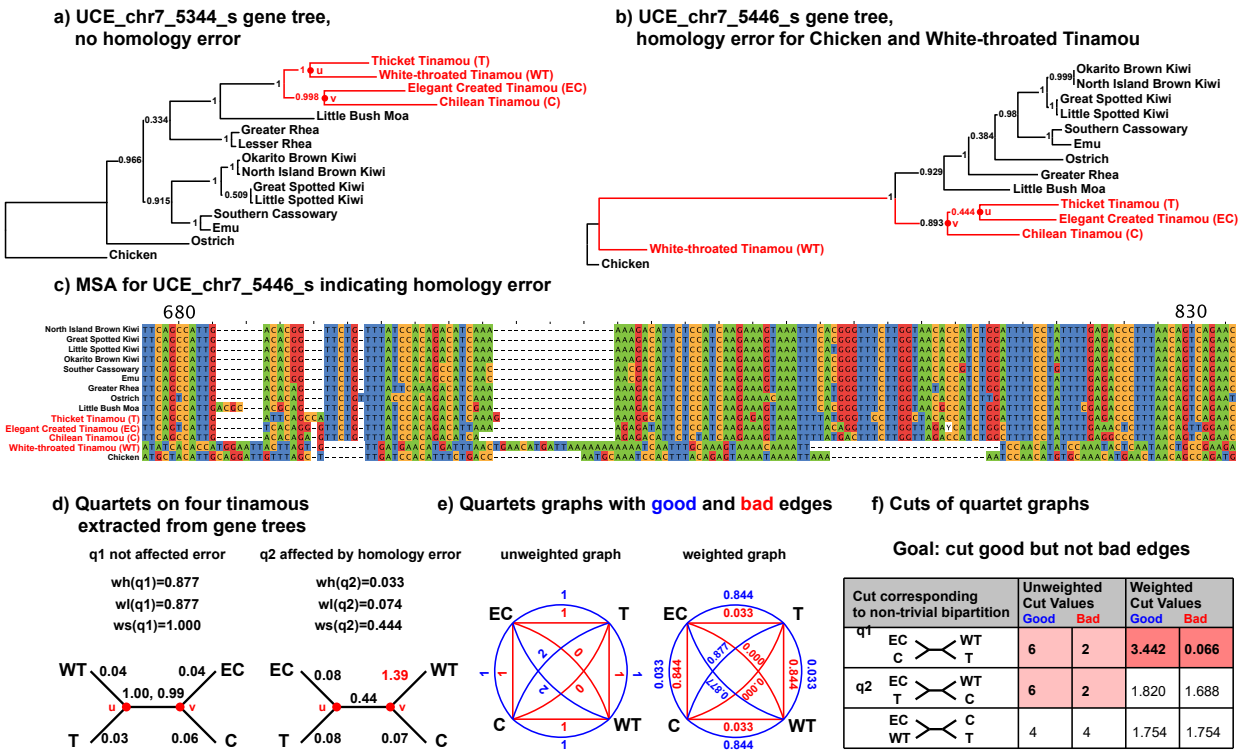


Figure 3.2: **Impact of hybrid weights on quartet graph and its best cut. a)** Gene tree estimated on UCE `chr7_5344_s` by [19], where u and v denote the anchor vertices for the quartet on the four Tinamou species: Thicket Tinamou (T), White-throated Tinamou (WT), Elegant Crested Tinamou (EC), and Chilean Tinamou (C). **b)** Gene tree estimated on UCE `chr7_5446_s` with long branch separating Chicken and White-Throated Tinamou from the remaining species. **c)** MSA for UCE `chr7_5446_s` suggests gene-level homology errors for Chicken and White-Throated Tinamou (visualized with JalView; 142). **d)** Quartets $q1$ and $q2$ on the four Tinamous species extracted from gene trees and `chr7_5344_s` and `chr7_5446_s`, respectively, along with their hybrid (wh), length (wl), and support (ws) weights. **e)** Quartet graphs built from both $q1$ and $q2$ without and with hybrid weighting. **f)** A cut of the quartet graph produces an internal branch in the species tree. Only three cuts yield internal branches. Cut quality is related to the ratio of good and bad edges removed (shown in table). For the unweighted graph, there are two best cuts (shaded). For the weighted graph, there is one best cut corresponding to $q1$ (shaded), which seems desirable because $q2$ is affected by homology errors.

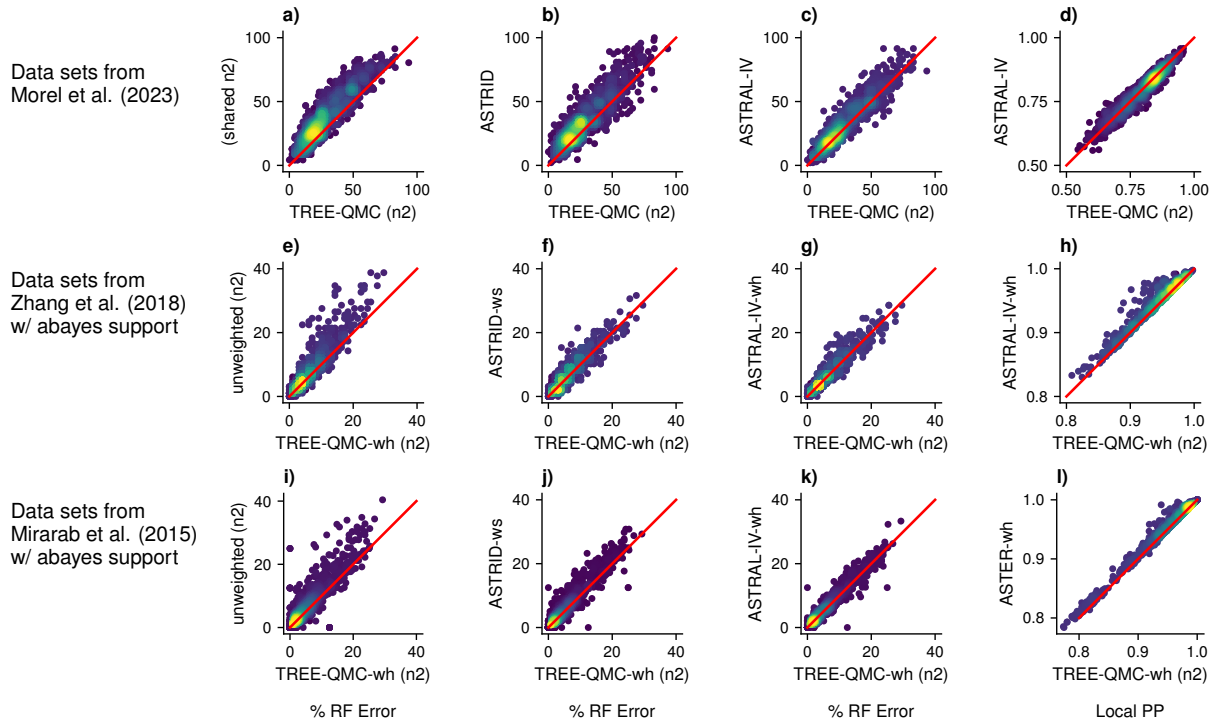


Figure 3.3: **Results on data sets simulated from three prior studies (rows).** Each dot in the scatter plot is a data set, with its position based on the evaluation metric (indicated below columns of subplots) for the method on the x-axis and the method on the y-axis. Lines indicate there is no difference between methods. Data points are colored based on density, with lighter colors indicating higher density and darker colors indicating lower density. For % RF error, higher density of data points above the line indicates that the version of TREE-QMC on the x-axis achieves lower error (and thus is more accurate) than the method on the y-axis.

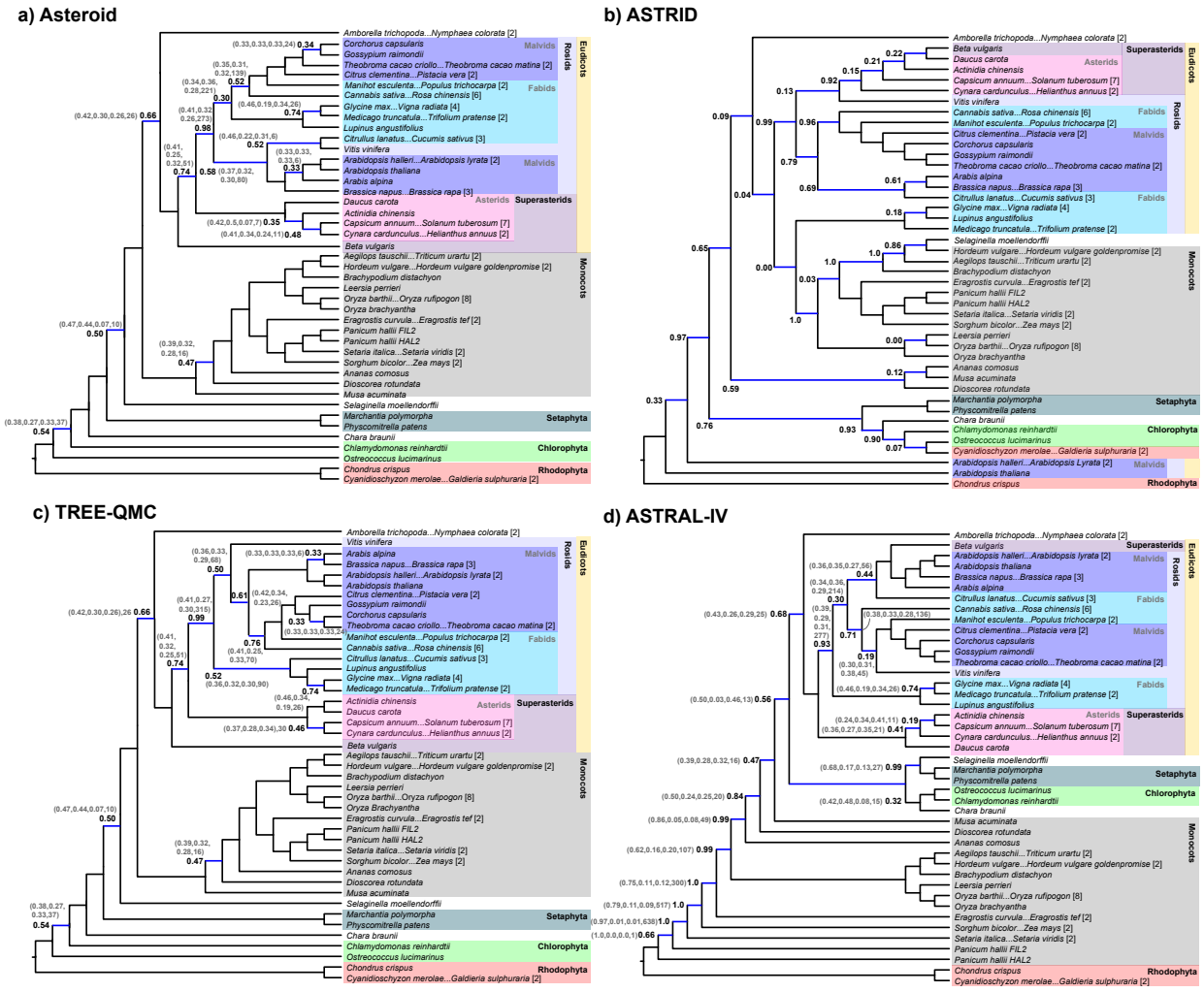


Figure 3.4: **Species trees estimated from data curated by [86].** Subfigures (a), (b), (c), and (d) show species trees we estimated using Asteroid, ASTRID, TREE-QMC, ASTRAL-IV, respectively. Blue branches are not in the RAxML tree from [86]. Branch support (i.e., Local PP) is shown for branches that disagree with the RAxML tree. QQS values and EN are given in parentheses. All species trees are shown on the same reduced the leaf set created based on subtrees in the RAxML that were shared by all estimated species trees (leaf labels have the form $X...Y[N]$ indicating a subtree in the RAxML tree with N taxa, starting taxon X and ending with taxon Y). The reduction in leaf labels from 81 to 43 indicates that the estimated trees are on many of the lower order relationships.

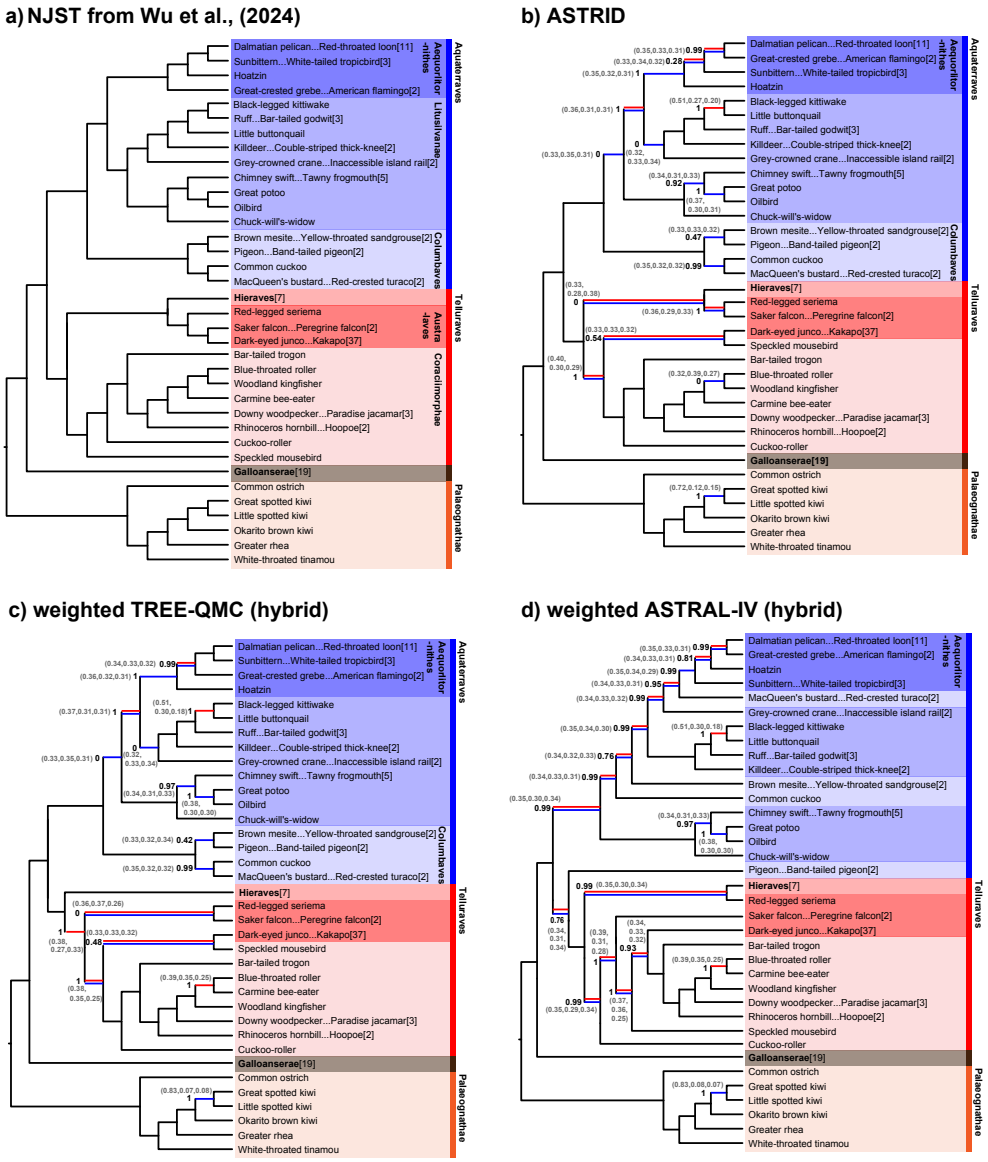


Figure 3.5: **Species trees estimated from data curated by [145].** Subfigure (a) is the NJst tree from Figure 2 in [145]. Subfigures (b), (c), and (d) show species trees we estimated on CDs and introns using ASTRID (no weighting and support weighting scheme), weighted TREE-QMC (hybrid or length weighting scheme), weighted ASTRAL (hybrid weighting scheme), respectively. Red and blue branches are not in the NJst and RAxML trees from [145], respectively. Branch support (i.e., Local PP) is shown for branches that disagree with the NJst and/or RAxML trees. QQS values (in parentheses) are based on unweighted quartets (with taxon filtering) for subfigure (b) and hybrid weighted quartets (without taxon filtering) for subfigures (c) and (d). All species trees are shown on the same reduced leaf set created based on subtrees in the NJst that were shared by all estimated species trees. Leaf labels have the form $X...Y[N]$ indicating the subtree in the NJst tree has N taxa, starting taxon X and ending with taxon Y. The reduction in leaf labels from 124 to 36 indicates that the estimated trees agree on most of the lower order relationships.

Gene tree estimation error. In our simulation study, we found that the hybrid quartet weighting scheme typically improved the accuracy of TREE-QMC compared to the unweighted version. Moreover, TREE-QMC with hybrid weighting outperformed the leading methods: ASTRID with support weighting and ASTRAL-IV with hybrid weighting. A limitation of these simulated data sets is that error is due to insufficient phylogenetic signal or other issues arising during gene tree estimation (they did not encompass errors due to poor quality multiple sequence alignments or model misspecification). Estimation errors *may* follow the "MSC+Error+Support" model, under which weighted ASTRAL-IV is statistically consistent, unlike the unweighted method [156]. Despite this consistency guarantee, we did not always find that greater quartet scores translated to greater accuracy; this could be due to the errors not following the model assumptions and/or the finite number of gene trees.

Homology errors. Our simulation study did not evaluate the impact of paralogy (multi-copy genes) or homology errors on species tree accuracy; however, the latter was evaluated using biological data. The effectiveness of methods on biological data is difficult to discern; however, it is promising that the hybrid and length weighting schemes made TREE-QMC and ASTRAL-IV more robust to *systematic* homology errors in the Palaeognathae data set [19]. We caution that there was still insufficient signal in this data set to resolve this part of the avian evolutionary history (also see 120).

Species tree topological stability and homology errors. We also reanalyzed a recent avian data set curated by [145]. A major finding of [145] was the separation of *Neoaves* into two major clades: *Telluraves* (land birds) and *Aquaterraves* (waterbirds and relatives). [130] later identified homology errors in the data set and showed that ASTRAL-III on the cleaned data set did not recover the *Telluraves* and *Aquaterraves* split. Subsequently, [145] responded that NJst on cleaned data set still recovered the split and suggested that ASTRAL-III was less stable than NJst.

Our reanalysis also found ASTRAL-IV to be less stable than the other methods, as ASTRAL-IV returned different trees when outlier taxa were removed with TreeShrink, unlike ASTRID and TREE-QMC, which returned the same tree regardless of taxon filtering. When considering the five ways of running weighted summary methods (with different inputs and different weighting schemes), ASTRAL-IV produced four different species trees (differing by up to six branches), whereas TREE-QMC and ASTRID each produced two trees (differing by three branches). This may have interesting implications for branch support estimated by bootstrapping procedures.

The trees produced by ASTRAL-IV were distinct from those produced by TREE-QMC and the other methods, including concatenation. Not only did ASTRAL-IV not recover the Telluraves/Aquaterraves split, it also displayed a highly imbalanced (caterpillar-like) branch patterning at the deeper nodes of Aquarterraves minus pigeons. Although the ASTRAL-IV trees were better supported and achieved the highest quartet scores, as previously mentioned, our simulation study did not suggest that higher mean branch support and total quartet score were always indicators of greater species tree accuracy. Interestingly, the TREE-QMC tree was closer to the trees produced by concatenation as well as the distance methods NJst and ASTRID. Although TREE-QMC and ASTRAL-IV are both heuristics for the same optimization problem, they have different approaches. It is possible that distance methods and TREE-QMC's divide-and-conquer framework behave more similarly to each other than ASTRAL-IV's heuristic search (it is worth noting that TREE-QMC was more robust to missing data than ASTRID in our simulation study). Additionally, TREE-QMC downweights quartets with artificial taxa based on subproblem decomposition, which could make the counting of quartets more local or diffuse across the species tree, akin to likelihood scores. Future work should further study ASTRAL-IV and TREE-QMC to better understand the impact of heuristic approaches on species tree accuracy.

Scalability. Compared to the unweighted TREE-QMC algorithm, the introduction of weighting schemes increased the time complexity of quartet graph construction by a factor of b, where

$b = O(n)$ is the number of artificial taxa for the subproblem. Interestingly, the empirical runtime of weighted TREE-QMC did not grow much compared in our computational experiments. We conjecture this difference between our theoretical and empirical study is due to the number b of artificial taxa per subproblem growing sub-linearly in the number n species or behaving more like a constant factor in practice. Under the latter assumption, both weighted and unweighted TREE-QMC have storage complexity $O(n^2)$ and time complexity $O(n^2 \log(n)k)$ with some assumptions on subproblem decomposition (Theorem B.5 in Supplementary Materials). Taken together, these theoretical and empirical results on the scalability of weighted TREE-QMC are promising.

In our computational experiments, weighted TREE-QMC was faster than weighted ASTRAL-IV when both were restricted to one thread; however, weighted ASTRAL-IV was faster when given with 16 threads. Weighted ASTRAL-IV uses multi-threading to speed up the quartet score calculation, parallelizing the computation across gene trees, which contribute independently to the quartet score. Parallelism across gene trees could also be exploited during graph construction by weighted TREE-QMC, and future versions of the software should take advantage of multi-threading and vector instructions available on modern processors.

Acknowledgments

YH and EKM thank Dr. Mark Springer and Dr. John Gatesy for sharing the estimated gene trees used in their reply to [145] and for helpful discussions about the data. YH and EKM thank the editor Dr. Matthew Hahn and the anonymous reviewers for detailed feedback that improved our study.

Funding

This research was funded by the State of Maryland. All computational experiments were performed on the Center for Bioinformatics and Computational Biology (CBCB) compute cluster at

the University of Maryland, College Park.

Software and Data Availability

Weighted TREE-QMC source code is available on Github: `https://github.com/molloy-lab/TREE-QMC`. Data sets are available on Dryad: `https://doi.org/10.5061/dryad.hdr7sqvsx`.

Chapter 4: TOB-QMC: Leveraging TREE-QMC for Species Network Reconstruction via Tree of Blobs

Supplementary materials referenced in this chapter appear in Appendix C.

4.1 Introduction

A tree structure is insufficient to represent the evolutionary history of species when there is gene flow between species or populations. Unlike species trees, species networks include reticulations (i.e., vertices with in-degree greater than one) that represent hybrid species or admixed populations; this leads to a number of challenges. First, the number of possible networks is larger than the number of possible trees, which already increases super-exponentially in the number of leaves [11]. Second, there are multiple sources of gene tree discordance, as gene trees can evolve with both incomplete lineage sorting (ILS) and gene flow, as modeled by the Network Multi-Species Coalescent (NMSC) [151, 152].

Broadly speaking, reconstructing species networks under the NMSC is a major scientific challenge. Earlier methods were based on computing likelihood within a comprehensive framework [153], considering the probability of a gene tree given a species network [152] as well as the probability of sequence data given a gene tree [31]. This calculation is quite computationally intensive, leading to the introduction of methods based on pseudo-likelihood [126, 151]. One of the most popular methods, SNaQ, takes (estimated) gene trees as input and seeks a level-1 (semi-directed) network that maximizes a pseudo-likelihood function based on subsets of four species (also called

a 4-tuple). Although the restriction to level-1 networks reduces complexity, SNaQ is still computationally intensive due to its quartic time pseudo-likelihood calculation, with scalability limited to around 30 species [59]. Similar issues arise for methods based on phylogenetic invariants and hypothesis tests on 4-tuples of species [2, 3, 9, 106].

To address scalability, Kolbow *et al.* [59] recently presented InPhyNet, a divide-and-conquer algorithm for reconstructing level-1 (semi-directed) species networks, similar to the TreeMerge approach [82, 83] for divide-and-conquer species trees. Central to InPhyNet's approach is the decomposition of the species set into smaller subsets on which SNaQ can be run. Sampling species according to the **tree of blobs**, which represents the tree-like parts of a network [43], enables InPhyNet to be statistically consistent. Unfortunately, trees of blobs, like the species networks, are challenging to reconstruct.

To our knowledge, the only method for reconstructing a tree of blobs under the NMSC is TINNiK [4]. TINNiK is statistically consistent and operates in two phases: first, hypothesis tests are applied to 4-tuples of species, and second, the results are used to reconstruct the tree of blobs using a distance-based approach. The first phase of TINNiK's algorithm appears to be $\Omega(n^4k)$ time, and the second phase appears to be $\Omega(n^5)$ time. More practically, TINNiK cannot scale to large numbers of taxa, requiring Kolbow et al. [59] to implement an alternative approach to subset decomposition that does not require a tree of blobs on the complete species set.

Here, we present a new approach, tentatively referred to as TOB-QMC, for reconstructing the tree of blobs under NMSC. TOB-QMC starts by estimating a tree from gene trees and then identifies and contracts the blob edges in the tree based on hypothesis testing around the branch. As we will show, hypothesis testing is computationally intensive, motivating us to introduce heuristics that reduce the overall time complexity of TOB-QMC to $O(n^3k)$. Perhaps surprisingly, TOB-QMC was not only more efficient than TINNiK but also more accurate in our simulation study, especially when the level of ILS is high.

The remainder of the chapter is organized as follows. In Section 4.2, we review terminology

and relevant background. In Section 4.3, we present the TOB method. In Section 4.5, we evaluate the performance of TOB-QMC using simulations. In Section 4.6, we conclude with discussing study limitations and directions of future research.

4.2 Terminology and Background

4.2.1 Definitions and Notations

A **directed phylogenetic network** $\mathscr{N}^+$ is a directed acyclic graph with the property that there exists a path between the root (i.e., a special vertex with in-degree 0) and all other vertices, without any parallel arcs or self-loops. All leaves (i.e., vertices with out-degree zero) are bijectively labeled by a set S of species. An internal node in $\mathscr{N}^+$ can be either a **speciation node** or a **reticulation node**. Speciation nodes are the vertices with in-degree one and out-degree greater than one; reticulation nodes are vertices with in-degree greater than one and out-degree one. Henceforth, networks are assumed to be **binary**, meaning all non-leaf vertices have degree three, except the root, which has degree two. A directed (i.e., rooted) phylogenetic tree is just a directed phylogenetic network without any reticulation nodes. A directed network can be converted into an **undirected** network, simply by suppressing the root and un-directing the edges. A **semi-directed** network is obtained by un-directing only the incoming edges to tree nodes, thus preserving information about reticulation events. We say that a phylogenetic tree is a **display tree** of a directed or semi-directed network if it can be obtained from the network by deleting only incoming edges to reticulation nodes but not their end points (Fig.4.1a,c).

A bridge is an edge whose deletion increases the number of connected components of the graph. The maximal bridgeless subgraphs of an undirected network are called **blobs**. The degree of a blob is given by the number of bridges adjacent to it. We refer to edges in blobs as **blob edges**, and other edges not in blobs (i.e., bridges) as **tree edges**. An undirected network is level-k if it can be simplified to a tree by removing at most k edges from each blob; thus, each vertex in a level-1

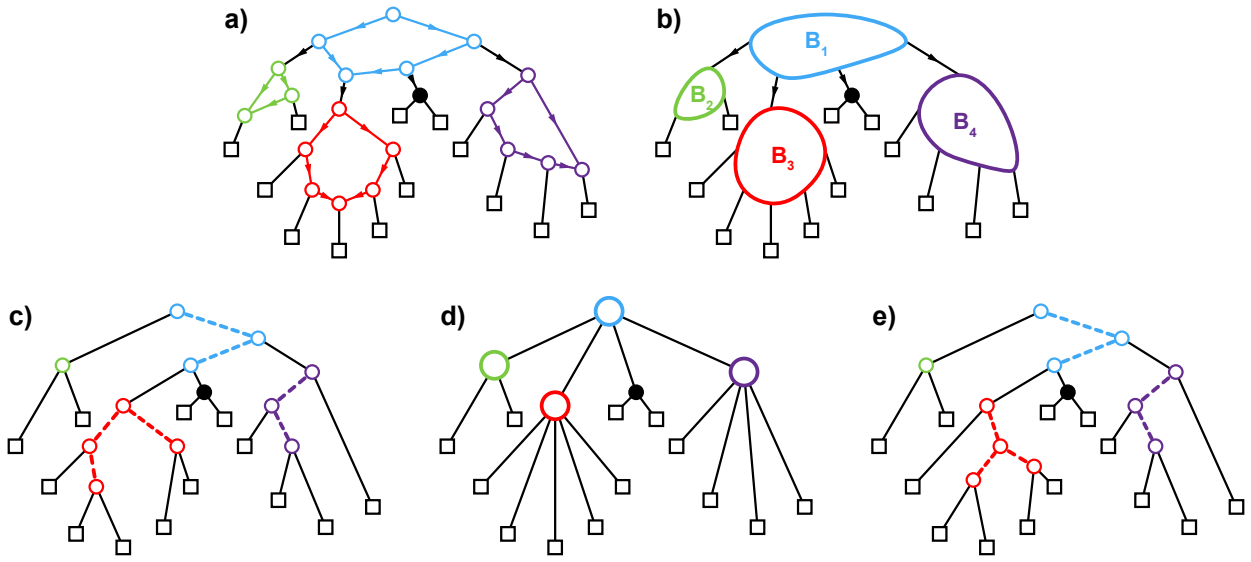


Figure 4.1: **Phylogenetic Networks, Display Trees, Tree of Blobs, and Refinements. a)** A level-1 directed phylogenetic network with four blobs (maximal bridgeless subgraphs) highlighted in different colors (note that all other blobs in the network do not contain any edges). **b)** Tree of blobs for the network in subfigure (a). **c)** A refinement of the tree of blobs subfigure (b) that is also a display tree of the network in subfigure (a). Dashed edges are contracted to obtain the tree of blobs. **d)** Another visualization of the tree of blobs. **e)** A refinement of the tree of blobs in that is not a display tree of the network.

network can participate in at most one cycle. Given an a directed or semi-directed network N^+, a **tree of blobs**, denoted $TOB(N^+)$, is obtained by (1) un-directing N^+, (2) contracting all edges in the same blob into single vertex, and then (3) suppressing all nodes with degree two [43]; this can be computed in linear time [4, 50].

Lastly, we review some terminology for phylogenetic trees. A tree T is a contraction of T' if T can be obtained from T' through a sequence of zero or more edge contractions. A tree T' is a refinement of T if and only if T' is a contraction of T [141]. A **bipartition** divides the taxa into two subsets, denoted $A|B$. Each edge e in a phylogenetic tree induces a bipartition, denoted $Bip(e) = A|B$, because deleting it (but not its enpoints) splits the tree into two connected components whose labeled leaves form the bipartition. Likewise, each non-terminal edge e in a phylogenetic tree induces a **quadrapartition**, denoted $Quad(e) = C|D|E|F$, because deleting it (along with its endpoints) splits the tree into four connected components. Note that for phylogenetic net-

works, bridges (i.e., tree edges) also induce bipartitions and quadrapartitions but blob edges do not. Lastly, a **quartet** is an undirected binary tree with four leaves (an undirected non-binary tree on four leaves is a **star**). Given a four-leaf set $\{a,b,c,d\}$, there are three possible quartets: $a,b|c,d$, $a,c|b,d$, and $a,d|b,c$. A **quartet concordance factor (qCF)** is the proportion of gene trees in which the topology of the quartet is observed among all gene trees resolving the quartet [10] (note that we use `qCFs(q)` to denote a 3-vector containing the qCF for each of the three possible quartets for a given subset q for four species).

4.2.2 TINNiK's Hypothesis Tests

To introduce our method TOB-QMC, we briefly review the two hypothesis tests employed by TINNiK, referred to as the **star-test** and the **tree-test** [3, 4]. Both tests take as input qCFs for a 4-tuple q of species and then return a p-value used to accept or reject the null hypothesis For the star-test, the null hypothesis is that the species phylogeny for q is a star. For the tree-test, the null hypothesis is that the species phylogeny for q is a non-binary tree. If the star-test passes *or* the tree-test fails, TINNiK designates q is a blob quartet (or B-quartet); otherwise, q is designated as a tree-like quartet (or T-quartet), with its topology indicated by the quartet tree with the highest qCF. Note that the star-test is important because the tree-test has a singularity when the three qCFs are equal, in which case there is no phylogenetic signal. The thresholds for the star-test and tree-test are hyperparameters *beta* (β) and *alpha* (α), respectively. Thus, the performance of TINNiK depends on a proper selection of these hyperparameters by users. The user may vary the value of α and β and run TINNiK multiple times to evaluate the robustness of the inferred tree of blobs. Larger β values and/or smaller α will result in more T-quartets, leading to trees of blobs with more internal edges (i.e., more tree-like structure).

4.3 TOB-QMC: Estimating the Tree of Blobs via Refinement Trees and Hypothesis Testing

We now present TOB-QMC, an approach for reconstructing trees of blobs from gene trees. The key idea behind TOB-QMC is to (1) reconstruct a refinement of the tree of blobs $T_B = TOB(N^+)$), denoted T'_B, and then (2) detect and contract branches found in T'_B but not T_B (these false positives are associated with blobs). The challenge is how to perform these steps in an accurate and scalable way.

4.3.1 Reconstruction of Refinement Tree

In step one, our goal is to reconstruct a refinement of the tree of blobs, which we refer to as the **refinement tree**. Consider a set $\mathscr{T}$ of gene trees that have evolved independently and identically distributed under the NMSC given some directed species network N^+. Our approach is motivated by our conjecture that any solution to the Maximum Quartet Support Species Tree (MQSST) problem is a refinement of the tree of blobs for N^+ under the NMSC with high probability given a sufficiently large number of gene trees. MQSST seeks a tree T that maximizes the sum of qCFs for the quartets displayed by T. Although MQSST is NP-hard [62], there are many effective heuristics for MQSST, such as ASTRAL [157] or TREE-QMC [45]. We leverage TREE-QMC, which is based on the Quartet Max Cut (QMC) framework of Snir and Rao [124], which is why we tentatively refer to our approach as TOB-QMC. The time complexity of TREE-QMC is $O(n^3k)$ where n is the number of species and k is the number of gene trees.

It is worth noting that a recent result by Dinh and Baños may appear to contradict our conjecture [24]. Specifically, the authors showed that, for a particular (non-anomalous) semi-directed network, the solution to MQSST tree will *not* be a display tree of N^+ under the NMSC model with high probability given a sufficiently large number of gene trees (Fig. 1 of [24]), although

interestingly, the solution to MQSST is indeed a refinement of the tree of blobs for N^+. It is easy to see that requiring a refinement of a tree of blobs is a much weaker condition than requiring a display tree of the network—just consider that any display tree of a directed or semi-network N^+ is a refinement of $TOB(N^+)$, but the reverse is not true (Fig. 4.1c–e). As we will show, TREE-QMC produces reasonable estimates of the refinement tree in practice, at least for the simulation conditions we study.

4.3.2 Detecting False Positives Edges in the Refinement Tree

After estimating a binary refinement tree in step one, our goal in step two is to identify the edges in it that are missing from the tree of blobs (FPs) and contract them. If the tree estimated in step (1) is indeed a refinement of the tree of blobs, then FP edges either correspond to hard polytomies (vertices with degree greater than 3) or blob edges in the directed network N^+. This motivates using the star-test and tree-test from TINNiK [3, 4]. However, we wish to avoid computing qCFs for all subsets of four taxa, as this would result in a time complexity of $\Omega(n^4k)$, similar to TINNiK, and ideally, the time complexity of contracting branches would be no worse than that of estimating a refinement tree.

Star-Test. First, we address the issue of detecting FP branches from polytomies in the species network by applying the star-test to each non-terminal branch e. Recall that branch e induces a bipartition $Bip(e) = A|B$ as well as a quadrapartition $Quad(e) = A'|A''|B'|B''$ where $A = A' \cup A''$ and $B = B' \cup B''$. In the naive (brute force) approach, we would consider all ways of sampling four taxa with information about the branch (i.e., sampling two taxa from A and two from B). Repeating this for every branch would effectively compute qCFs for all subsets of four species, which has time complexity $\Omega(n^4k)$; a lazy upper bound on the time complexity of star-tests is $O(n^5)$. To avoid these computational issues, we compute the average qCFs around the quadrapartition induced by e (i.e., sampling one taxa from each of A', A'', B', and B''); this can be performed in $O(nk)$ time

using the approach of Sayyari and Mirarab [114]. A star test can be performed for a given 4-tuple in constant time; thus, the entire star-tests computation takes $O(n^2k)$ time. Using the average qCFs around the quadrapartition induced by e is reasonable because, in the case of a star, all 4-tuples sampled will have qCFs that are equal and thus pass the test. It is worth noting that in practice, there could be differences in qCFs from gene tree estimation error, in which case an average could be more robust.

Tree-Test. Second, we address the issue of FP branches from blob edges in the species network by applying the tree-test to each non-terminal branch e. Although it is tempting to apply the tree-test to average qCFs around the quadrapartition induced by e, as in the star-test, issues can arise with this approach. Consider the simple case where the directed network has only one reticulation vertex and the refinement tree is one of the two possible trees displayed by the network. In this case, a blob edge e in the refinement tree can either pass or fail the tree-test depending the 4-tuple sampled around the induced bipartition (Fig. 4.2).

Moreover, the average qCFs around a quadrapartition can incorrectly support tree-like structure. Consider the semi-directed network N^+ with one reticulation vertex shown in Figure 4.3. Let e_{FP} be the branch of interest in the refinement tree, which is one of the two possible trees displayed by N^+ for simplicity. Let $A|B$ be the bipartition induced by e, and let $A'|A''|B'|B''$ be the quadrapartition induced by e, where $A' \cup A'' = A$, $B' \cup B'' = B$. It is easy to see that e_{FP} is a blob edge because the network is still connected after the deletion of e_{FP} due to the hybridization edge $h = u \mapsto v$. Let h' be the other incoming edge of the reticulation node v. Now subdivide each part of the quadrapartition so that $A' = A_i$, $B' = B_j$, $A'' = A_1 \cup A_2 \cup \cdots, A_{i-1}$, and $B'' = B_1 \cup B_2 \cup \cdots, B_{j-1}$, where $A_1, A_2, \cdots, A_i$ and $B_1, B_2, \cdots, B_j$ are subtrees in the forest formed after deleting edges on the paths from u to v in the network.

Now consider sampling four species around the quadraparition induced by e with $a_1 \in A'$, $a_2 \in A''$, $b_1 \in B'$, and $b_2 \in B''$. There are four cases that can occur.

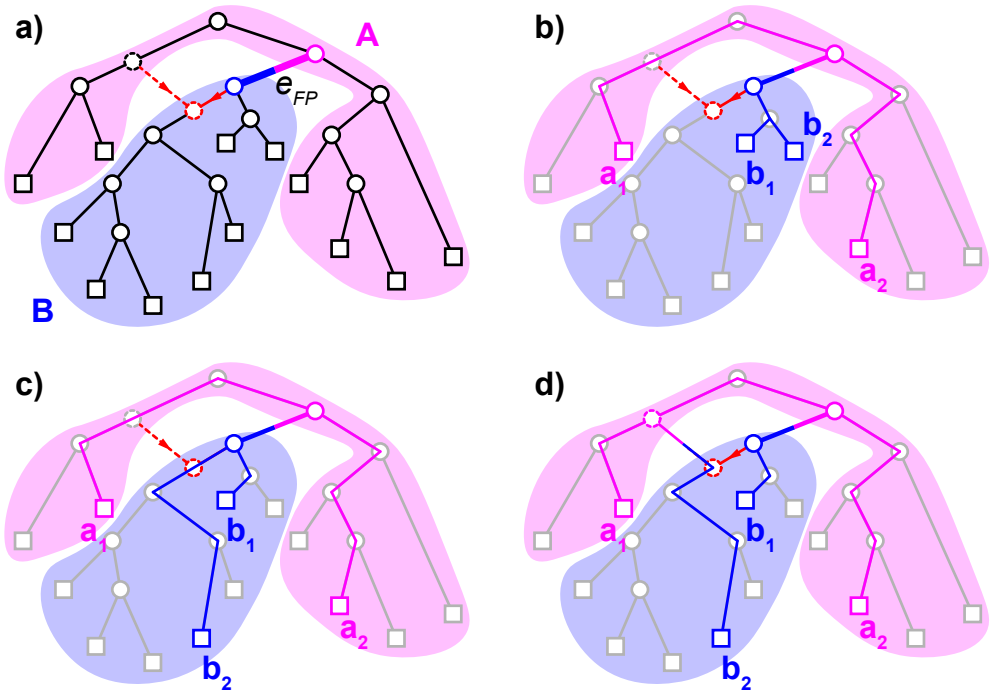


Figure 4.2: **Detecting false positive branches in the refinement tree. a)** Refinement tree is a display tree of a network, created by deleting the dashed red edge but not its endpoints. Consider contracting branch e_{FP} that induces bipartition $A|B$, with A shared in pink and B shared in blue, based on the results of the tree-test. Sample taxa around the bipartition (i.e., sample two taxa a_1, a_2 from A and two taxa b_1, b_2 from B). **b)** The tree-test will pass because deleting all but the four sampled taxa from the network and suppressing vertices of degree two produces a species tree (no signals of gene flow). An alternative selection of four species around the bipartition produces a directed species network, with the two display trees shown in **c)** and **d)**. Thus, the tree-test will fail, indicating that e_{FP} is a blob edge and should be contracted.

Case (1): b_2 is above v (i.e., $b_2 \in B_2, B_3, \cdots, B_{j-1}$) *and* $a_2 \in A_1$. To compute the qCFs, we look at the two display trees, one associated with deleting h' (with inheritance probability $1-\gamma$) and the other associated with deleting h (with inheritance probability γ). In either display tree, the quartet on (a_1, a_2, b_1, b_2) always has topology $a_1, a_2|b_1, b_2$ and internal branch length is x. Thus, the expected qCFs are

$$qCF(a_1, a_2|b_1, b_2) = 1 - \frac{2}{3}e^{-x} \geq qCF(a_1, b_1|b_2, b_2) = qCF(a_1, b_2|a_2, b_1) = \frac{1}{3}e^{-x} \qquad (4.1)$$

which passes the tree-test [9, 24, 75, 126]. *Case (2):* b_2 is above v (i.e., $b_2 \in B_2, B_3, \cdots, B_{j-1}$) *and*

$a_2 \in A_2 \cup A_2 \cup \ldots A_{i-1}$. This is the same as case (1).

Case (3): b_2 is below v (i.e., $b_2 = b_h \in B_1$) *and* $a_2 \in A_1$. In the display tree created by deleting h, the quartet on (a_1, a_2, b_1, b_2) has topology $a_1, a_2|b_1, b_h$ and internal branch length x. Thus, the expected qCFs are the same as Equation 4.1:

$$qCF(a_1, a_2|b_1, b_h) = 1 - \frac{2}{3}e^{-x} \geq qCF(a_1, b_1|b_2, b_h) = qCF(a_1, b_h|a_2, b_1) = \frac{1}{3}e^{-x} \tag{4.2}$$

However, in the display tree created by deleting h', the quartet on (a_1, a_2, b_1, b_2) has topology $a_1, b_1|a_2, b_h$ and internal branch length y. Thus, the expected qCFs are

$$qCF(a_1, b_1|b_2, b_h) = 1 - \frac{2}{3}e^{-y} \geq qCF(a_1, a_2|b_1, b_h) = qCF(a_1, b_h|a_2, b_1) = \frac{1}{3}e^{-y}. \tag{4.3}$$

Combining Equations 4.2 and 4.3 together, the final expected qCFs are

$$\begin{aligned} qCF(a_1, a_2|b_1, b_h) &= (1-\gamma) \cdot (1 - \frac{2}{3}e^{-x}) + \gamma \cdot \frac{1}{3}e^{-y} \\ qCF(a_1, b_1|b_2, b_h) &= (1-\gamma) \cdot \frac{1}{3}e^{-x} + \gamma \cdot (1 - \frac{2}{3}e^{-y}) \\ qCF(a_1, b_h|a_2, b_1) &= (1-\gamma) \cdot \frac{1}{3}e^{-x} + \gamma \cdot \frac{1}{3}e^{-y} \end{aligned} \tag{4.4}$$

which fails the tree test for various settings of γ, x, and y (as it does not preserve the inequality/invariant shown for Equation 4.1). *Case (4):* b_2 is below v (i.e., $b_2 = b_h \in B_1$) *and* $a_2 \in A_2 \cup A_3 \cup \ldots A_{i-1}$. This is similar (3).

To summarize, we show that the qCFs signal the existence of a hybridization edge only when b_2 is sampled below the reticulation node. Simply averaging qCFs around the edge could be indistinguishable, for example, in the case where the subtree below v consists only of very few leaves (and thus the qCFs being averaged are dominated by tree-like signal). Instead of taking an average, we desire the 4-tuple that minimizes the p-value. The challenge is locating the 4-

tuple that strongly signals a blob edge when the network structure and the location of reticulation nodes are unknown. An exhaustive search of all subsets of taxa around a bipartition takes $O(n^5k)$ time, as previously discussed under star-tests. To increase scalability, we propose a hill-climbing heuristic algorithm to search for subset of four species with the lowest p-value (Algorithm 4.1). In each iteration, a random subset of four taxa around the bipartition induced by edge e are selected, referred to as a 4-tuple. After that, it keeps exploring the neighboring 4-tuples, which have exactly one leaf different from the current 4-tuple. If the minimum p-value of the neighbor 4-tuples is smaller than that of the current 4-tuple, the algorithm updates the current 4-tuple. The process continues until a local minimum is reached and then repeats. The final result is given by the minimum p-value found across all iterations.

Our implementation of the search for a minimum p-value computes qCFs for selected 4-tuples on the fly. To do so, we use the well-known binary lifting algorithm [20] to retrieve the LCA of any pair of leaves in a gene tree in $O(1)$ time. The preprocessing time, as well as the storage cost, is $O(nlog(n)k)$ where n is the number of taxa and k is the number of gene trees (this is less than the time complexity of TREE-QMC). After preprocessing, we can calculate qCFs for any 4-tuple in $O(k)$ time and then perform the tree-test in constant time. In our implementation, we store qCFs so that they can be used again later (this does not impact the time complexity). Lastly, we need to factor in the number of tree-tests performed per branch; this is controlled a user-specified iteration limit c). To summarize, each branch can be processed in $O(ck)$ time, and the entire tree-test phase takes $O(nck)$ time.

4.3.3 Implementation

The algorithm for building the tree of blobs is shown in Algorithm 4.2. First, we build a refinement tre with TREE-QMC, which has time complexity $O(n^3k)$, as previously discussed. Next, we compute the p-values for each branch based on the the star-test and the tree-test. The

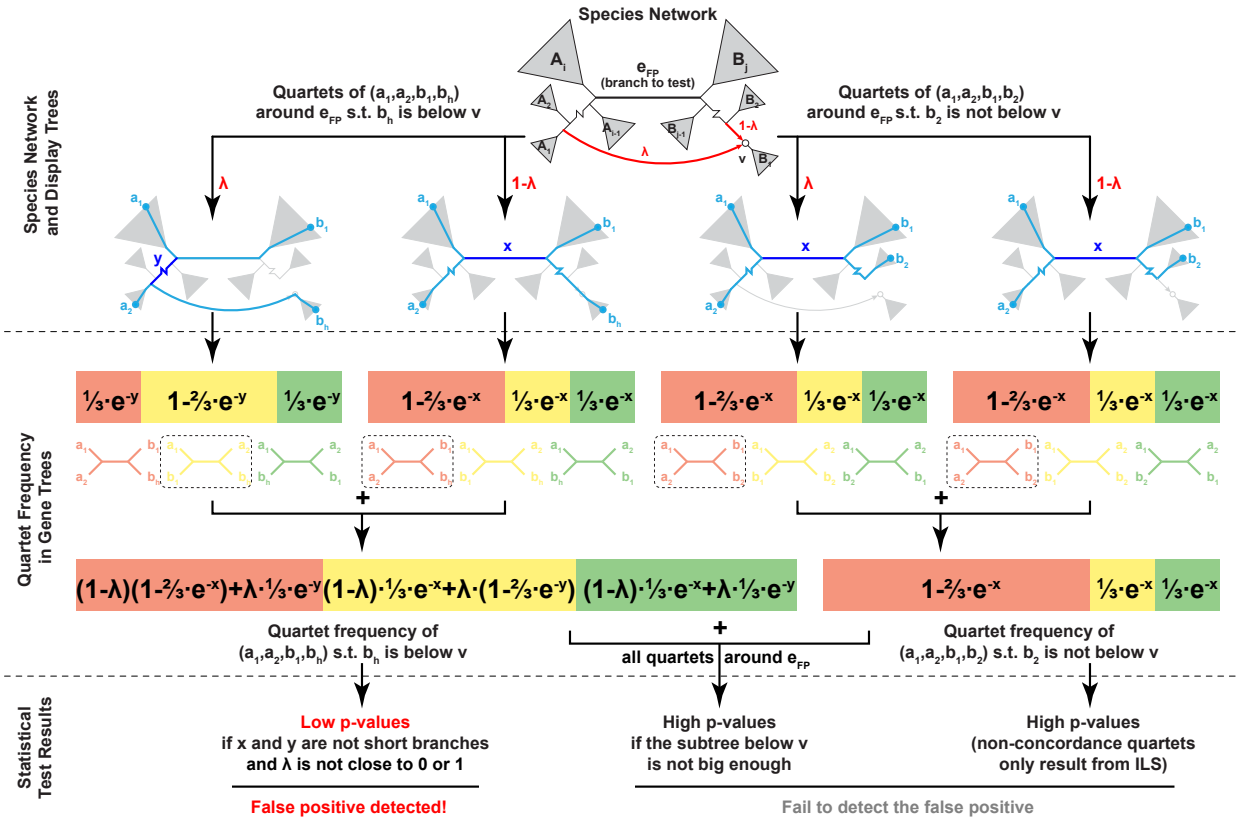


Figure 4.3: **Issues detecting blob edges from average quartet concordance factors around quadrapartition induced by a branch of interest.**

Algorithm 4.1: Heuristic Search: `minQuartetTreeTest(·)`

input : set $\mathscr{T}$ of gene trees, refinement tree T'_B, an internal edge e in T'_B, and an iteration limit

output: minimum p-value found during search, sampling quartets around the bipartition induced by e

$A, B \leftarrow$ `bipartition(`T'_B, e`)`;
$q^* \leftarrow ()$;
$i = 0$;
while $i <$ *iteration limit* **do**
  $q_{min} \leftarrow (a_1, a_2, b_1, b_2)$ `// randomly pick` a_1, a_2 `in` A `and` b_1, b_2 `in` B
  repeat
    $\mathbf{N} \leftarrow \varnothing$;
    for $a' \in A \setminus \{a_1, a_2\}$ **do** $\mathbf{N} \leftarrow \mathbf{N} \cup \{(a', a_1, b_1, b_2), (a', a_2, b_1, b_2)\}$;
    for $b' \in B \setminus \{b_1, b_2\}$ **do** $\mathbf{N} \leftarrow \mathbf{N} \cup \{(a_1, a_2, b', b_1), (a_1, a_2, b', b_2)\}$;
    $q_{new} \leftarrow$ **argmin**$_{q \in \mathbf{N}}$ `quartetTreeTest(qCF(`$\mathscr{T}, q$`))`;
    $i \leftarrow i + |\mathbf{N}|$;
    if *p-value of* q_{new} *is smaller than* q_{min} **then** $q_{min} \leftarrow q_{new}$;
  until q_{min} *is not updated* ***or*** $i >$ *iteration limit*;
  if *p-value of* q_{min} *is smaller than* q^* **then** $q^* \leftarrow q_{min}$;
return *p-value of* q^*

Algorithm 4.2: Tree of Blobs Construction

input : set $\mathscr{T}$ of gene trees, α, and β

output: estimated tree of blobs T_B

$T'_B \leftarrow$ refinement tree estimation using TREE-QMC;
for *each branch e in* T'_B **do**
  $A', A'', B', B'' \leftarrow$`quadrapatition(`$T'_B, e$`)`;
  $Q \leftarrow \{q(a_1, a_2, b_1, b_2) : a_1 \in A', a_2 \in A'', b_1 \in B', b_2 \in B''\}$;
  `avg` $\leftarrow \frac{1}{|Q|} \sum_{q \in Q}$ `qCF(`$q, \mathscr{T}$`)` ;
  $e.p_{star} \leftarrow$ `quartetStarTest(`*`avg`*`)`;
  $e.p_{tree} \leftarrow$ `minQuartetTreeTest(`$\mathscr{T}, T'_B, e$`)`;
$T_B \leftarrow$ copy T'_B;
for *each branch e in* T'_B **do**
  if $e.p_{star} > \beta$ ***or*** $e.p_{tree} < \alpha$ **then** contract e in T_B ;
return T_B

star-test can be performed on all branches in $O(n^2k)$ time, which does exceed that of running TREE-QMC. The tree-test can be performed on all branches in $O(nck)$ time, where c is the user-specified iteration limit. In practice, we set $c = O(n^2)$ so that the time complexity of tree-tests does not exceed that of running TREE-QMC. Lastly, we contract branches based on the α and β hyperparameters, which can be done in $O(n)$ time. To summarize, TOB-QMC has time complexity $O(n^3k)$, whereas TINNiK appears to have a time complexity of $\Omega(n^4k + n^5)$. Another advantage of TOB-QMC is that we can explore the impact of the α and β hyperparameters without any additional work, unlike TINNiK.

4.4 Empirical Evaluation

We now describe an experimental study evaluating TOB-QMC; see the Supplementary Materials for details.

4.4.1 Simulated Data Sets

Species network simulation. Phylogenetic networks with 50 and 100 taxa were simulated under a birth–death-hybridization process with the R package `SiPhyNetwork` (v1.1.0) [55]. We used the general sampling approach (GSA) function, which simulates m taxa under the simple sampling approach (SSA) and then samples from time periods where the desired number n of taxa is present. The desired network level cannot be specified explicitly in `SiPhyNetwork`. Instead, we performed a large number of simulation runs so that we are able to select the networks with desired levels. For 50 and 100 taxa, we took first 30 networks (replicates) of level-0, first 50 networks of level-1, and the first 50 of level-2. The summary statistics for the networks are reported in Table 4.1. The true tree of blobs was computed for each network using the algorithm introduced in [50].

Table 4.1: **Properties of simulated networks and trees of blobs.** Values in table are averages across 50 networks (replicates) plus/minus standard deviations. Reticulations is the number of reticulation nodes in the network. Inheritance rate is the average inheritance rate, taking the lower of the pair associated with each reticulation node in a network. Number of Blobs is the number of non-trivial blobs in the tree of blobs (i.e., number of internal nodes with degree greater than 3). Blob size is the average size of non-trivial blobs in the tree of blobs. Number of internal branches is also computed for the tree of blobs. As an example, the tree of blobs in Figure 4.1b has three non-trivial blobs, and average blob size of 5, and 4 internal branches.

Model		**Network**		**Tree of Blobs**		
Taxa	Level	Reticulations	Inheritance Rate	# of Blobs	Blob Size	Internal Branches
50	1	1.76 ± 0.86	0.42 ± 0.06	1.58 ± 0.78	8.18 ± 3.19	39.26 ± 4.36
	2	2.700 ± 0.90	0.418 ± 0.07	1.560 ± 0.73	12.154 ± 4.91	33.220 ± 4.53
100	1	1.64 ± 0.74	0.42 ± 0.06	1.62 ± 0.72	11.03 ± 3.98	85.12 ± 5.24
	2	2.54 ± 0.70	0.42 ± 0.06	1.52 ± 0.67	16.43 ± 7.05	77.68 ± 5.2

Gene tree simulation. Second, gene trees were simulated under NMSC with the Julia package `PhyloCoalSimulations` (v1.0.0) [36]. We simulated $1,000$ gene trees for each network, with one individual per species. To vary the amount of ILS, we multiplied branch lengths in the species tree by 2.0, 1.0, 0.5, and 0.25, and 0.125; henceforth referred to as very low, low, medium, high, and very high ILS, respectively. The amount of ILS can be easily evaluated for level-0 networks, as the normalized Robinson-Foulds (RF) distance [107] between gene trees and the level-0 networks (species trees), as there is no gene flow. The average RF distance ranged from 45% to 95% (Fig. 4.4), indicating that gene trees were largely discordant with the species trees; we expect this to also be true for networks with higher levels.

4.4.2 Methods

We implemented TOB-QMC within TREE-QMC v3.0.4 [45], invoking the code from TINNiK to perform star-tests and tree-tests. To evaluate the performance of the search algorithm for tree-tests, we ran TOB-QMC with an iteration limit at $2n$ and $n/4$, denoted as **TOB-QMC-m** and **TOB-QMC-f**, respectively. Both of these settings have the same time complexity but differ by constant factors. We also tested exhausted search, denoted **TOB-QMC-s**. We compared our TOB-QMC with TINNiK, as implemented within the R package `MSCQuartets` v3.2 [106]. The detailed

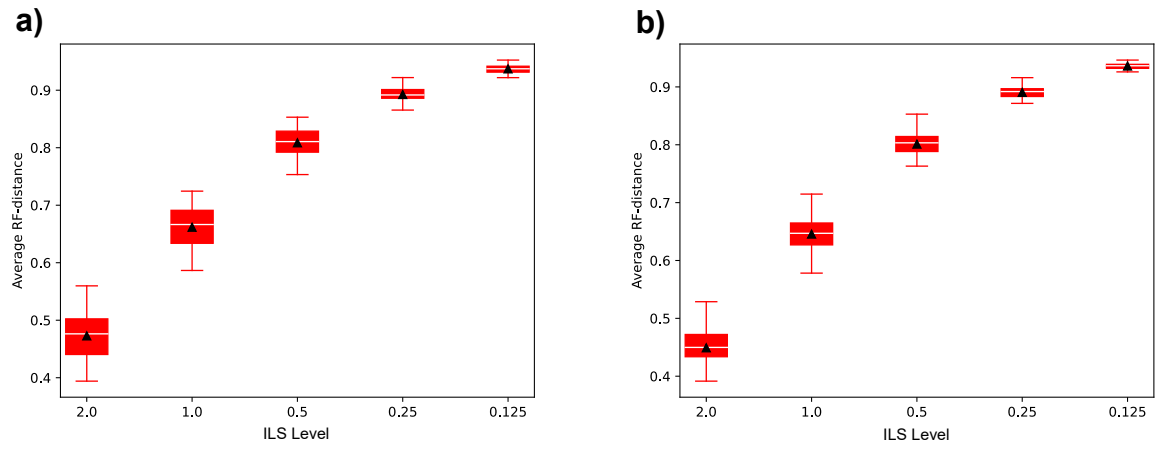


Figure 4.4: **ILS level of level-0 networks with a) 50 taxa and b) 100 taxa**. ILS was varied by multiplying the lengths of branches in networks by constant factor (y-axis). Smaller constant factors result in shorter branches and thus higher ILS levels. The y-axes represents the RF distance between the species tree and the gene tree, averaged across all gene trees. Boxplots show data for 30 replicate level-0 networks.

commands to run TOB-QMC and TINNiK are provided in the Supplementary Materials. For both methods, we varied the values of β from 0.90, 0.95, and 1.00, with 0.95 being the default setting of TINNiK. We also varied the values of α from 10^{-5}, 10^{-6}, 10^{-7}, 10^{-8}, 10^{-9}, and 0. Note that $\alpha = 0$ means the tree-test always passes and thus no edges are contracted due to tree-tests in TOB-QMC.

4.4.3 Evaluation Metrics

Topological Accuracy and Error. We evaluated methods by comparing the estimated and true trees of blobs. Let T_0 be the true tree and T_1 be the estimated tree. The number of false positives (FP) is defined as the number of bipartitions in T_1 but not in T_0. The number of false negatives (FN) is the number of bipartitions in T_0 but not in T_1. The number of true positives (TP) is the number of bipartitions shared by both trees. Precision and recall can be computed based on these values. We also report the normalized Robinson-Foulds distance [107] which is the sum of FP and FN divided by $2(n-3)$, where n is the number of taxa (note that both trees will only have $n-3$ internal branches if they are binary so this is essentially just dividing by a constant).

Runtime. We report the runtime of the methods on 10 replicates of each model condition, with ILS level factors limited to 1.0 (low), 0.5 (medium) and 0.25 (high). TINNiK was run multiple times for different values of α and β, whereas TOB-QMC was run once to build a display tree whose branches are annotated with p-values. After that, the tree of blobs was built for any set of levels of statistical tests in $O(n)$ time. We report the TOB-QMC time cost by adding the time to build the display tree and the time to contract branches.

4.5 Results

Impact of TOB-QMC's blob detection algorithm. To investigate the effectiveness of TOB-QMC's blob detection algorithm, we divide the FP branches into two types according to whether or not the bipartition is compatible with the true tree of blobs or not. Incompatible branches are due to errors in the estimated refinement tree, since all branches in a true refinement tree should be compatible with the true tree of blobs. We found that TOB-QMC's blob detection algorithm is able to contract the majority of FP branches that were compatible with the true tree of the blobs when the level of ILS was low or medium (Figures 4.5d–f,m–o and C.1–C.4 in the Supplementary Materials). However, a much smaller proportion of incompatible FPs were contracted (Figures 4.5g–i,p–r). These results suggest that TREE-QMC's blob detection algorithm is successful at removing FP edges that correspond to blobs but not those that correspond to errors in reconstructing the refinement tree.

Impact of TOB-QMC's heuristic search for tree-tests. To investigate the effectiveness of TOB-QMC's search heuristic, we compared the number of FP branches from heuristic search to those from exhaustive search (Fig. 4.6 and Figs. C.5–C.6 in the Supplementary Materials). For α values between 10^{-7} and 10^{-9}, the heuristic algorithm with a limit of $2n$ (TOB-QMC-m) did not introduce extra false positives and therefore achieved the same accuracy as the exhausted search.

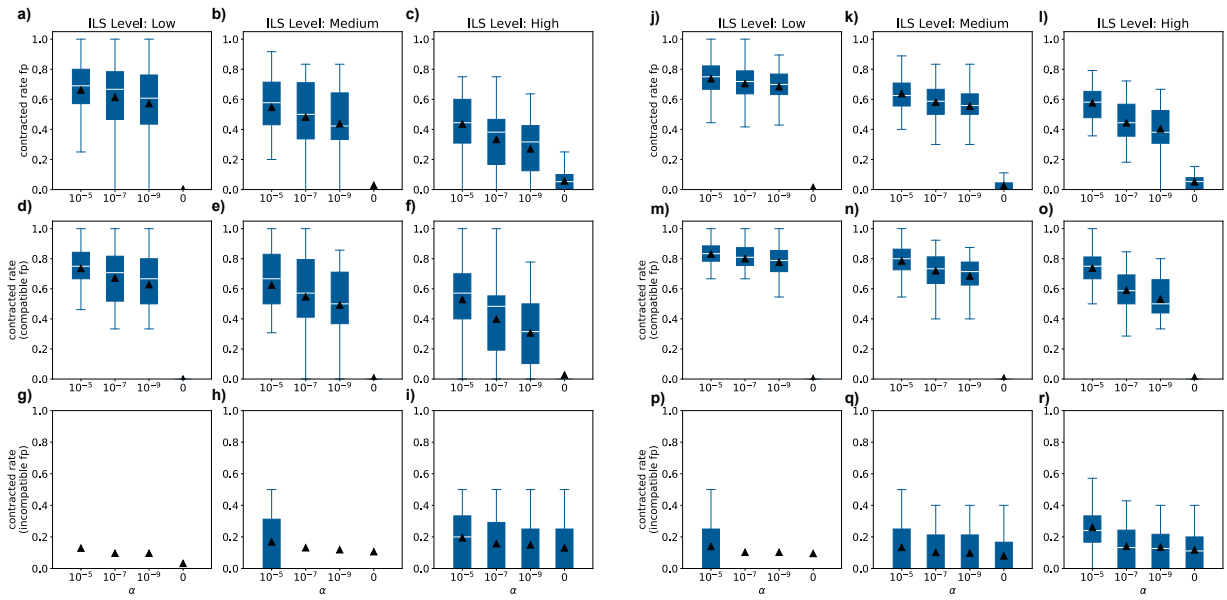


Figure 4.5: **Impact of TOB-QMC's blob detection algorithm.** Left and right subfigures (9 subplots each) are results on level-1 networks with 50 and 100 taxa, respectively. Columns represent different ILS levels. Rows show proportion of FPs contracted (i.e., the contraction rate). We divide the FP in the tree estimated by TOB-QMC-m ($\beta = 0.95$) into two types: those compatible with the true tree of blobs and those not. The first row shows the contraction rate of all FPs. The second and third rows show the contraction rate of compatible and incompatible FP, respectively.

The the fast heuristic search (TOB-QMC-f) with a limit of $n/4$ only increased the false positive edges by 5 at most.

Topological accuracy and error. The topological error and accuracy of TOB-QMC and TINNiK are shown in Figure 4.7 and Figures C.7–C.10 in the Supplementary Materials. The accuracy of both methods decreased when the level of ILS increased. For low ILS, both methods were close in accuracy, with TOB-QMC being slightly better; the difference in accuracy increased with the level of ILS, giving TOB-QMC a clear advantage. The values of α and β also affect the performance of the methods. Both methods tended to produce estimated tree of blobs with fewer branches when α increased or β decreased, which in turn decreased precision and increased recall. The best value

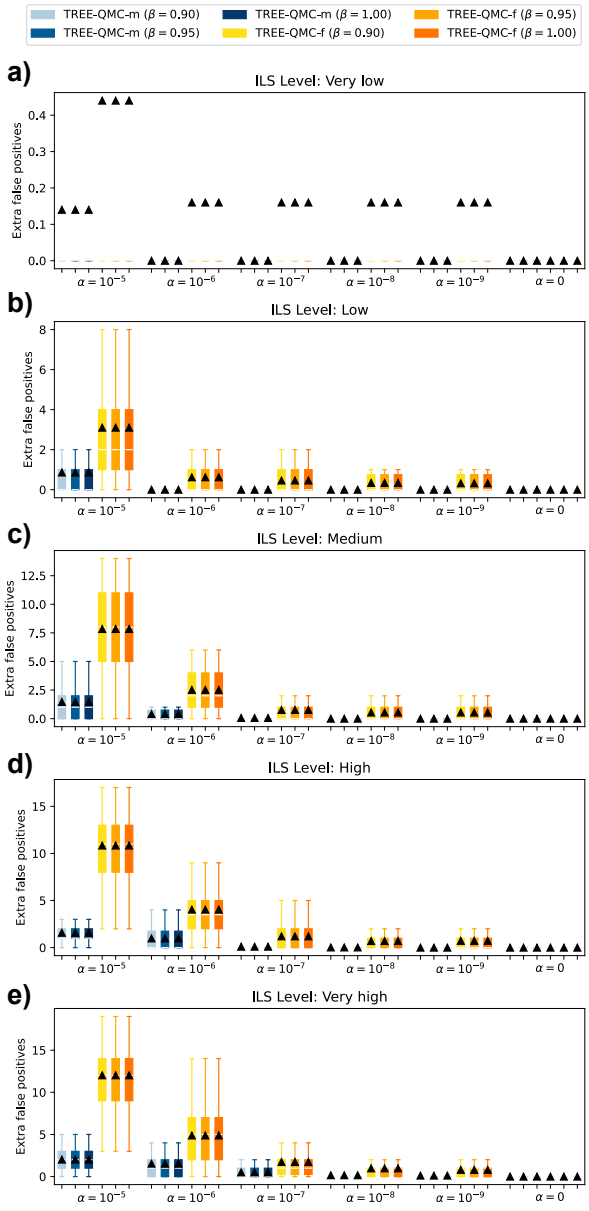


Figure 4.6: **False positives introduced by the heuristic search algorithm.** Number of additional FPs in trees produced by the heuristic algorithms TREE-QMC-m (blue) and TREE-QMC-f (yellow) compared to those produced by the exhausted algorithm on level-1 networks with 100 taxa. Columns correspond to the ILS levels

of α for both methods is between 10^{-7} and 10^{-9} when considering the RF error; the impact of β was less significant. Overall, we observed that TOB-QMC (with heuristic search) achieved better

accuracy than TINNiK in terms of RF distance, precision, and recall.

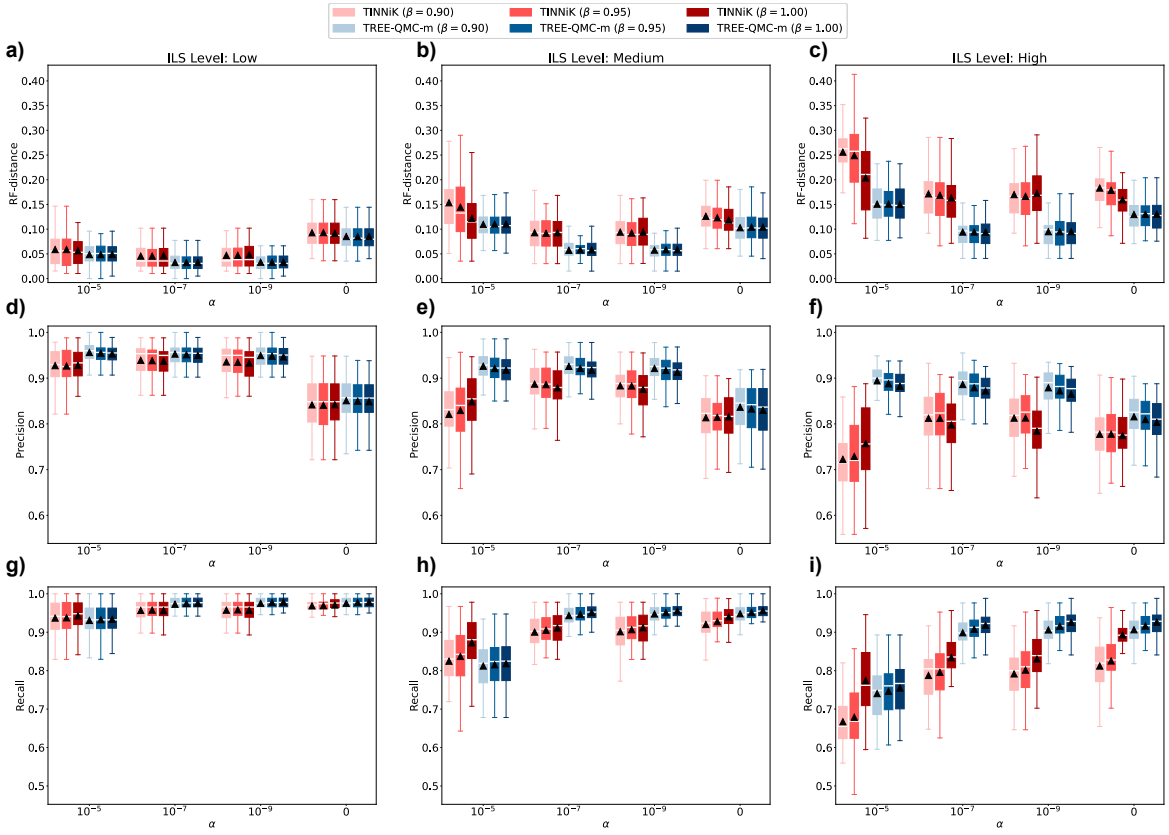


Figure 4.7: **Topological Error and Accuracy of TOB-QMC and TINNiK.** Topological Error and Accuracy of TOB-QMC-m and TINNiK on level-1 networks with 100 taxa. Red color corresponds to TINNiK, blue color corresponds to TOB-QMC-m (i.e., search heuristic with iteration limit $2n = 200$), and shades correspond to the β hyperparameter. Rows show normalized RF error, precision, and recall. Columns show ILS level. The x-axes of all subfigures represent different α values.

Runtime. The runtime of TOB-QMC and TINNiK is shown in Figure 4.8 and Figures C.11–C.12 in the Supplementary Materials. For all model conditions, TINNiK was the most time-consuming, taking even more time than TOB-QMC using the exhaustive search. TREE-QMC using the heuristic search (with a iteration limit of $2n$) was at least 2.5 times faster than TINNiK on the 50-taxon networks. For 100 taxon networks, the speedup was even more significant, with TOB-QMC being at least 8 times faster than TINNiK. Lowering the iteration limit down to $n/4$ further improves

efficiency.

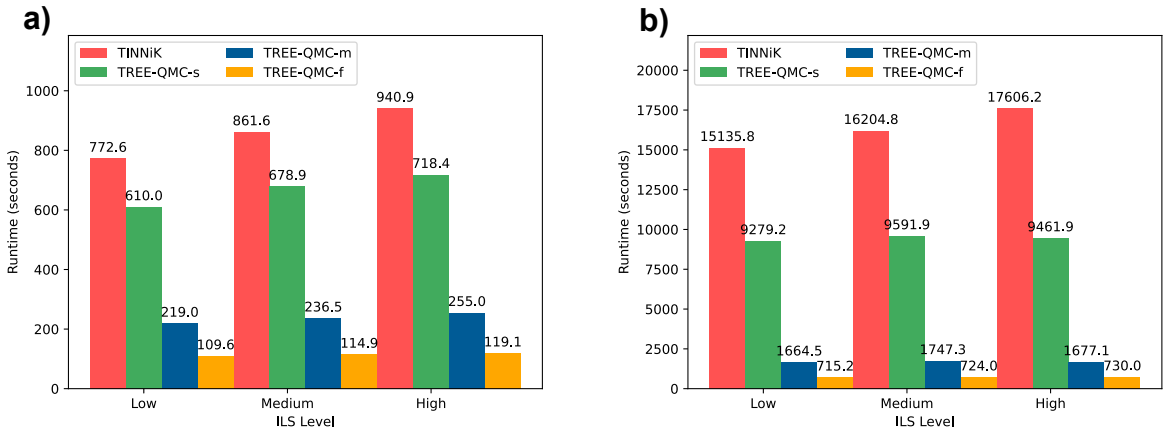


Figure 4.8: **Runtime of TREE-QMC and TINNiK on level-1 networks with a) 50 taxa and b) 100 taxa.** All methods are run with hyperparameters $\alpha = 10^{-7}$ and $\beta = 0.95$.

4.6 Discussion and Conclusions

We introduced a new approach to estimate the tree of blobs, called TOB-QMC, which operates by (1) seeking a refinement of the tree of blobs and then (2) identifying blob edges to contract. We showed that although the average qCF around a quadrapartition is efficient to compute, it is not effective at detecting blob edges, motivating us to devise a hill climbing heuristic to search for minimum p-values of the tree-test. The use of this heuristic gives TOB-QMC an overall time complexity of $O(n^3k)$. TOB-QMC was not only more efficient on larger numbers of taxa than TINNiK but also more accurate, suggesting that our heuristic search is effective in practice at least on the conditions we simulated. On the other hand, TINNiK has been proven statistically consistent under the NMSC, whereas there is no such guarantee for TOB-QMC. Future work should address this issue, starting with our conjecture that a solution to MQSST is a refinement of the true tree of blobs with high probability given sufficient data.

Additionally, TOB-QMC has yet to be evaluated on biological data; this should be addressed in future work. Biological data sets can have gene trees that have missing taxa or errors due

to gene tree estimation and other sources. We only used true gene trees in simulation; future work should address this issue, especially as gene tree estimation error can impact the test statistics on qCFs [17]. To address this issue, it may be useful to try species tree estimation methods based on weighted quartets for reconstructing refinement trees, as such methods are more robust to GTEE [47, 156]. Other hypothesis tests for gene flow among four species could also be employed, including those based on site patterns such as HyDe [12] and Patterson's D-Statistic [92].

During the time our study was conducted, a new method, called Squirrel [49] was published. Squirrel operates by creating candidate trees of blobs and then "resolving' the blobs to produce networks. The candidate trees of blobs are formed by contracting edges in estimated trees, similar to our proposed method TOB-QMC. However, Squirrel differs from TOB-QMC in several important ways. First, Squirrel and TOB-QMC differ in their inputs; Squirrel takes a dense set of quarnets (networks on four taxa) as input, which are estimated from multiple sequence alignments, whereas TOB-QMC essentially relies on quartet concordance factors computed from gene trees. Second, Squirrel and TOB-QMC differ in the approach they use to contract branches. Squirrel iteratively contracts the branch with the lowest split support, producing a collection of candidate trees of blobs, whereas TOB-QMC performs statistical tests to contract blob edges based on the NMSC. An interesting direction of future research is to evaluate TOB-QMC's in the context of species network reconstruction by using it to estimate a tree of blobs and then applying Squirrel's procedure for resolving blobs given quarnets estimated from gene trees under the NMSC. Along these lines, it would be interesting to test the effectiveness of the TOB-QMC method within InPhyNet's divide-and-conquer algorithms for network inference [59].

Chapter 5: Conclusion

Motivated by the challenges of genome-scale species phylogeny reconstruction, this dissertation explores whether the Quartet Max Cut (QMC) framework of Snir and Rao [124] can serve as a foundation for developing scalable and accurate summary methods. We present three new methods: TREE-QMC, Weighted TREE-QMC, and TOB-QMC. The first two methods (TREE-QMC and Weighted TREE-QMC) are designed for species tree reconstruction from gene trees. Both are faithful to the original QMC framework but introduce efficient algorithms for reconstructing the quartet graph directly from gene trees, while also ensuring that it is properly normalized and incorporating weights from gene tree branch lengths and support values, in the case of Weighted TREE-QMC. These algorithms alleviate the computational burden of extracting all quartets, which was previously required for researchers to run QMC as a summary method. Overall, TREE-QMC and Weighted-TREE-QMC advance the scalability, accuracy, and robustness of the QMC framework, positioning QMC as a leading summary method in comparison to the widely used ASTRAL family of methods [77]. The third method (TOB-QMC) leverages TREE-QMC within a new pipeline for tree of blob reconstruction, improving accuracy and scalability compared to the leading method TINNiK. Overall, TOB-QMC suggests the utility of the QMC framework for species network reconstruction. Study limitations and future work have been outlined throughout this dissertation; we now outline several broad directions.

5.1 Future Directions

Runtime Analyses. This dissertation evaluates scalability of methods in terms of empirical running time and worst-case (big oh) time complexity. QMC's divide-and-conquer framework makes it difficult to establish time complexity without making some strong assumptions. Our proofs of time complexity for both TREE-QMC and Weighted TREE-QMC assume that the subproblem decomposition is perfectly balanced, meaning that each graph cut splits the species set in half between the two child subproblems. One other hand, the worst-case time complexity does not always explain empirical performance very well; for example, Weighted TREE-QMC has a worst case time complexity of $O(n^4k)$ but scales much better in practice. Under the assumption that the number of artificial taxa is bounded by a constant, the time complexity of Weighted TREE-QMC reduces to $O(n^2 \log(n)k)$, same for the unweighted version. Rather than making these strong assumptions, it would be interesting to perform an average case analysis allowing for variation in subproblem division, both in terms of the total number of species in each subproblem as well as the proportion of artificial taxa. This would require us to consider more parameters and derive the expected time complexity given certain distributions of taxa.

Another direction is to argue for the optimality of our algorithms by analyzing the lower bounds on the time complexity for constructing quartet graphs and determining whether our proposed methods achieves these bounds. Conditional optimality [1, 8, 48] can provide strong evidence that an algorithm is near-optimal, assuming no major breakthrough occurs in complexity theory. For example, TREE-QMC constructs the quartet graph from a gene tree in $O(n^3)$ time, where n is the number of species. It would be beneficial to show that, under standard assumptions such as the Strong Exponential Time Hypothesis (SETH) or the Exponential Time Hypothesis (ETH), no significantly faster algorithm exists, i.e. that constructing the graph in $O(n^{3-\varepsilon})$ time for any constant $\varepsilon > 0$ would violate these assumptions.

Statistical Consistency. This dissertation focuses on quartet-based summary methods, many of which are heuristics for the Maximum Quartet Support Species Tree (MQSST) problem. The optimal solution to MQSST is a statistically consistent estimator under the Multi-Species Coalescent (MSC), which is, in part, why quartet-based summary methods popular for species tree estimation in the context of incomplete lineage sorting (ILS). However, it is unknown whether there are conditions under which TREE-QMC will return the optimal solution with high probability given sufficient data; in other words, there is no proof of statistical consistency for TREE-QMC. In the future, we hope to study this theoretical question, including looking at the impact of quartet graph normalization and quartet graph weighting based on gene tree branch lengths and support values, as the latter could be used to extend consistency results to the "MSC+Error+Support" model proposed by Zhang and Mirarab [156].

Models of Evolution. The evaluation studies in this dissertation focus on the condition where gene trees evolve with ILS (although gene trees could also be missing taxa or include false positive branches due to estimation errors). We additionally allowed gene tree discordance due to gene flow in Chapter 4, running TREE-QMC within a pipeline for reconstructing the tree of blobs for a species network. Although ILS and gene flow impact the evolution of major groups (e.g., whales [139], bats [98], birds [54, 90], and butterflies [27]), they are not the only sources of gene tree discordance. For example, genes (and even entire genomes) can be duplicated or deleted. Genome duplication is an important evolutionary mechanism in plants [143], for example. The result is that the history of gene evolution reflects not only coalescent events but also duplication and loss events. Increasingly, summary methods are being developed for gene trees that have evolved with duplications and losses, for example ASTRAL-pro [158]. In the future, we hope to evaluate and improve TREE-QMC in this context. We also hope to consider gene tree discordance due to horizontal gene transfer (HGT), which is common in bacteria. Although HGT results in non-tree-like evolution, it can be modeled differently than gene flow under the Network MSC (e.g., [109]).

Experimental Evaluation. The evaluation studies conducted in this dissertation are based on branch differences (e.g., the RF distance), either between the true and estimated tree for simulated data sets or between a reference and estimated tree for biological data sets. RF distance is a course-grained measurement based on tree topology alone. It treats branches equally regardless of their lengths and the support values, meaning that poorly supported differences can lead to high branch error compared to the true tree, which may be over-interpreted as significant errors. Likewise, branch errors near the leaves of the tree (more recent divergences) are penalized the same as branch errors near the root (deeper divergences), although one or the other of these cases may be more biologically significant to researchers. This obscures whether there are differences between methods in terms of their ability to reconstruct branches at different divergence depths/times. Additionally, two trees may have the same RF distance to the true/reference tree, but their shapes (i.e., to what extent they are balanced or imbalanced) could differ substantially. Tree shape is considered an important feature of evolutionary processes [57]. Moreover, methods may systematically favor certain tree shapes; this is of interest because we observed that TREE-QMC produced more balanced trees compared to ASTRAL-IV on two of the biological data sets studied. Thus, in the future, we hope to enhance our evaluation paradigms to leverage tree shape metrics [33] as well as metrics that are aware of branch length, support values, and divergence depths/times.

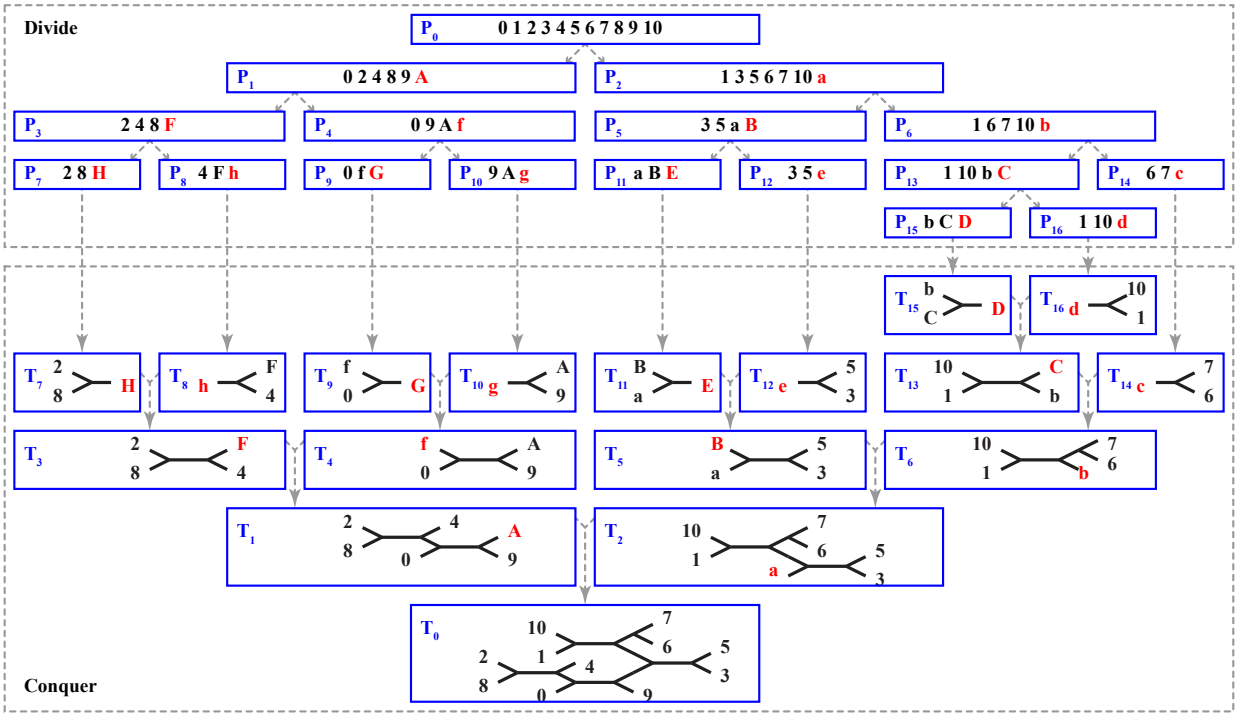
Future Data Sets and Scalability. This dissertation takes advantage of public data sets that were popular for method evaluation when the benchmarking studies were conducted. For species trees, the largest simulated data set we analyzed had 1000 species and 1000 genes [78], and the largest biological data set we analyzed had 124 species and 10,627 genes [145] (note that the avian data set from [54] had more genes 14,446 but fewer species 48). For trees of blobs, the largest simulated data set we analyzed had 100 species and 1000 genes. Although these data set sizes reflect those being analyzed in many systematic studies today; in the future, we hope to analyze larger data sets that have been recently published (e.g., a new avian data set from [137] has 363 species and

63,430 genes) and perform simulations that reflect data set sizes of the future (e.g., the Vertebrates Genome Project aims to assembly high quality genomes for >60,000 vertebrate species [105]).

Appendix A: Supplementary Materials for Chapter 2

A.1 Divide-and-Conquer Framework

Figure A.1: **Divide-and-conquer framework utilized by wQMC, TREE-QMC, and wQFM.** Initially, the input data (either gene trees or weighted quartets) are on species set $\mathscr{S} = \{0, 1, \ldots, 10\}$. At each step in the divide phase, we identify a bipartition and recurse on the implied subproblems. For problem P_0, we identify bipartition $\mathscr{E}|\mathscr{F}$, where $\mathscr{E} = \{0,2,4,8,9\}$ and $\mathscr{F} = \{1,3,5,6,7,10\}$. We then introduce artifical taxa A (representing the taxa in $\mathscr{F}$) and a (representing the taxa in $\mathscr{E}$) and recurse on species sets $\{A\} \cup \mathscr{E}$ (subproblem P_1) and $\{a\} \cup \mathscr{F}$ (subproblem P_2), updating the input data (either gene trees or weighted quartets) accordingly. We continue in this fashion until there are fewer than four taxa (e.g., problem P_7) in which case we return the only possible tree (e.g., tree T_7). At each step in the conquer phase, we connect pairs of trees using the artifical taxa. On the final conquer step, we connect trees T_1 and T_2 at artificial taxa A and a, producing a tree on species set $\mathscr{S}$.



A.1.1 Subproblems and Artificial Taxa.

At each step in the divide phase, wQMC finds a bipartition on species set $\mathscr{X}$ using the input quartets $\mathscr{Q}_{\mathscr{X}}$. The the quartets are then updated based on the implied subproblems [7]. Given bipartition $\mathscr{E}|\mathscr{F}$, a quartet $q = A,B|C,D \in \mathscr{Q}_{\mathscr{X}}$ is assigned to one of four cases:

1. *satisfied* if $A,B \in \mathscr{E}$ and $C,D \in \mathscr{F}$ (or vice versa),

2. *violated* if

   - $A,C \in \mathscr{E}$ and $B,D \in \mathscr{F}$ (or vice versa) or
   - $A,D \in \mathscr{E}$ and $B,C \in \mathscr{F}$ (or vice versa),

3. *untouched* if $|\{A,B,C,D\} \cap \mathscr{E}| = 4$ or $|\{A,B,C,D\} \cap \mathscr{F}| = 4$, and

4. *deferred* if $|\{A,B,C,D\} \cap \mathscr{E}| = 3$ or $|\{A,B,C,D\} \cap \mathscr{F}| = 3$.

If the quartet is satisfied or violated, it is discarded because it will be satisfied or violated regardless of the bipartitions produced at future steps in the divide phase of the algorithm. If the quartet is untoched or deferred, it is assigned to one of the two subproblems: $\mathscr{Q}_{\mathscr{E}+}$ or $\mathscr{Q}_{\mathscr{F}+}$, where $\mathscr{Q}_{\mathscr{E}+}$ denotes the set of quartets formed by adding the untouched and deferred quartets with 4 or 3 leaves labeled by elements of $\mathscr{E}$ (and similarly for $\mathscr{Q}_{\mathscr{F}+}$). To add a deferred quartet to $\mathscr{Q}_{\mathscr{E}+}$, its one leaf labeled by an element of $\mathscr{F}$ is relabeled with artificial taxon F. After this relabeling process has been completed for all deferred quartets, there can be multiples of the same quartet (each with its own weight). These multiples are replaced by a single quartet whose weight is the sum.

A.2 Experimental Study

A.2.1 Computational Resources and Empirical Runtime

Running time is reported as the wall-clock time (i.e., the amount of time that elapses between the beginning and end of the computation). All computational experiments were performed on cluster nodes outfitted with 128 GB RAM and 32 cores (2 $\times$ 16-Core AMD Opteron(tm) Processor 6378 2400MHz) and running Red Hat Enterprise Linux 7 (x86_64) operating system. There are 12 nodes with these specifications, which can be accessed via a queue that allows exclusive access but limits the allowed memory to 36 GB. Therefore, we ran methods with exclusive access to the node with access to 36GB of memory, assigning all analyses of the same replicate data set to the same node. All five methods we evaluated are single-threaded.

A.2.2 Species Tree Estimation Commands

ASTRAL, FASTRAL, and TREE-QMC.

ASTRAL version 5.7.7 [157] was run with the following command:

```
java -Xmx36G -D"java.library.path=<path to ASTRAL>/lib" \
    -jar <path to ASTRAL>/astral.5.7.7.jar \
    -t0 -i [input gene trees] -o [output species tree] &> [output log file]
```

Note that `-t0` means that branch length and support estimation are not performed.

FASTRAL [23] was run with the following command:

```
fastral --ns 1,10,20,20 --nt 1000,500,250,100 --k 1000 \
    --it [input gene trees] --os [output directory]/samples \
    --aggregate [output directory]/[output constraints] \
```

```
    --o [output directory]/[output species tree] \
    --time [output directory]/[output timing file] \
    --path_ASTRAL [ASTRAL path] \
    --path_ASTRID [ASTRID path] &> [output log file]
```

Note that `-nt 250,125,63,25 -k 250` where used for data sets with 250 genes. We ran FASTRAL with the version of ASTRAL and ASTRID distributed with the code; however, we modified the command to call ASTRAL so that it could use 36 GB of RAM and so that it did NOT compute branch support for the final tree.

TREE-QMC version 1.0.0 was run with the following command:

```
TREEQMC -n 0 -i [input gene trees] \
    -o [output species tree] &> [output log file]
```

Note that `-n 1` and `-n 2` where also run and in this case the input gene trees were the refined trees from running TREE-QMC with `-n 0`. This is so the only difference between run of TREE-QMC is the normalization scheme (and not the refinement of polytomies in the input gene trees).

wQMC and wQFM.

Unlike the previous methods, wQMC and wQFM take weighted quartets as input. We used the scripts provided with wQFM (`https://github.com/Mahim1997/wQFM-2020`) to extract weighted quartets from the gene trees. Specifically, we used the command:

```
quartet-controller.sh [input gene trees] \
    [output weighted quartets] &> [output log file]
```

which calls by `triplets.soda2103` executable from [14]. We then selected the binary quartets

```
grep "),(" [input weighted quartets] >> [output weighted quartets for wQFM]
```

and re-formatted them for wQMC:

```
sed 's/),(/|/g' [input weighted quartets for wQFM] | \
    sed 's/(//g' | sed 's/)//g' | sed 's/; /:/g' \
    > [output weighted quartets for wQMC]
```

wQFM version 1.3 [69] was run with the following command:

```
java -Xmx36G -jar [path to wQFM]/wQFM-v1.3.jar \
    -i [input weighted quartets] -o [output species tree] &> [output log file]
```

wQMC version 3.0 [7] was run with the following command:

```
max-cut-tree qrtt=[input weighted quartets] \
    weights=on otre=[output species tree] \
    &> [output log file]
```

A.2.3 Quartet Score and Branch Support Estimation Commands

Quartet score (and branch support) was computed for estimated species trees with the following command:

```
java -Xmx36G -D"java.library.path=<path to ASTRAL>/lib" \
     -jar <path to ASTRAL>/astral.5.7.7.jar \
     -t2 -q [input species tree] -i [input gene trees] \
     -o [output scored species tree] &> [output log file]
```

A.2.4 Replicates Excluded from ASTRAL-II Data

All comparisons between methods are made on the same set of replicate data sets. For analyses of estimated gene trees, we excluded replicate data set from analyses whenever more than half of the 1000 gene trees had the majority of their branches unresolved. Specifically, we excluded

- 1 replicate (# 41) from the 10-taxon model condition
- 2 replicates (# 21, 41) from the 50-taxon model condition
- 2 replicates (# 8, 47) from the 100-taxon model condition
- 3 replicates (# 8, 15, 49) from the 200 taxon model condition with very high ILS and shallow speciation

We also excluded replicates on which ASTRAL-III failed to complete; this occurred only for 3 replicates (# 6, 8, 38) from the 1000-taxon, 1000-gene model condition.

A.2.5 Properties of Simulated Data

Table A.1: **Properties of ASTRAL-II data sets.** ILS for each replicate is quantified as the normalized RF distance between the true species tree and the true gene tree average across all 1000 gene trees. GTEE for each replicate is quantified as the normalized RF distance between the true and estimated gene trees averaged across all 1000 gene trees. AD for each replicate is the normalized RF distance between the true species tree and estimated gene tree averaged across all 1000 gene trees. The values in the table is the average ($\pm$ standard deviation) across all replicates.

species tree height	speciation	# of taxa	ILS	GTEE	AD
0.5X	deep	200	0.68 ± 0.02	0.44 ± 0.14	0.74 ± 0.03
0.5X	shallow	200	0.69 ± 0.02	0.44 ± 0.12	0.74 ± 0.03
1X	shallow	10	0.17 ± 0.06	0.19 ± 0.09	0.28 ± 0.08
1X	shallow	50	0.31 ± 0.04	0.26 ± 0.11	0.42 ± 0.08
1X	shallow	100	0.33 ± 0.02	0.26 ± 0.09	0.44 ± 0.05
1X	deep	200	0.34 ± 0.02	0.34 ± 0.12	0.47 ± 0.08
1X	shallow	200	0.34 ± 0.02	0.27 ± 0.12	0.44 ± 0.07
1X	shallow	500	0.34 ± 0.01	0.28 ± 0.11	0.45 ± 0.07
1X	shallow	1000	0.35 ± 0.01	0.30 ± 0.11	0.47 ± 0.08
5X	deep	200	0.09 ± 0.01	0.28 ± 0.11	0.31 ± 0.10
5X	shallow	200	0.21 ± 0.02	0.21 ± 0.13	0.33 ± 0.09

Table A.2: **Properties of avian and mammalian simulated data.** Downloaded data sets [69], which did not include true gene trees for the 0.5X and 2X species tree scales for mammalian data set.

species tree scale	sequence length	ILS	GTEE	AD
Avian (48 taxa, 1000 estimated gene trees)				
0.5X	500	0.591 ± 0.002	0.597 ± 0.094	0.687 ± 0.001
1X	500	0.473 ± 0.002	0.597 ± 0.066	0.637 ± 0.002
2X	500	0.354 ± 0.002	0.619 ± 0.002	0.597 ± 0.002
Mammalian (37 taxa, 200 estimated gene trees)				
1X	250	0.330 ± 0.005	0.427 ± 0.009	0.538 ± 0.007
1X	500	0.330 ± 0.005	0.282 ± 0.058	0.438 ± 0.006
1X	1000	0.330 ± 0.005	0.161 ± 0.006	0.380 ± 0.006
1X	1500	0.330 ± 0.005	0.119 ± 0.005	0.362 ± 0.006

Table A.3: **Runtimes for ASTRAL-II data (1000 estimated gene trees).** Runtime (minutes) is reported for all methods (note: a runtime of zero minutes means it rounded down). Values shown are averages (± standard deviations) across across replicate data sets on which all methods completed (note: ASTRAL-III did not complete on 3 replicates with 1000 taxa and 1000 genes). Lowest median values are in bold. The value in parentheses for wQMC and wQFM is the fraction of the runtime spent weighting quartets to give to these methods as input.

Species tree height	# of taxa	spec.	ASTRAL-III	FASTRAL	TREE-QMC n0	TREE-QMC n1	TREE-QMC n2	wQMC	wQFM
1000 estimated gene trees									
0.5X	200	deep	77.4 ± 27.9	**1.6 ± 0.4**	2.7 ± 0.5	3.0 ± 0.6	3.0 ± 0.6	NA	NA
0.5X	200	shallow	72.7 ± 29.0	**1.5 ± 0.3**	2.6 ± 0.5	2.8 ± 0.5	2.8 ± 0.3	NA	NA
2X	10	shallow	**0.0 ± 0.0**	0.1 ± 0.0	**0.0 ± 0.0**	**0.0 ± 0.0**	**0.0 ± 0.0**	0.2 ± 0.0 (0.99)	0.2 ± 0.0 (0.94)
2X	50	shallow	0.6 ± 0.6	0.3 ± 0.0	**0.2 ± 0.0**	**0.2 ± 0.0**	**0.2 ± 0.0**	15.8 ± 3.0 (1.00)	16.8 ± 3.0 (0.94)
2X	100	shallow	3.0 ± 2.4	0.7 ± 0.1	**0.6 ± 0.1**	**0.6 ± 0.1**	**0.6 ± 0.1**	256.8 ± 34.7 (1.00)	290.2 ± 38.4 (0.88)
2X	200	deep	17.1 ± 11.4	**1.2 ± 0.1**	2.4 ± 0.1	2.5 ± 0.2	2.5 ± 0.2	NA	NA
2X	200	shallow	12.0 ± 9.5	**1.2 ± 0.2**	2.4 ± 0.4	2.5 ± 0.4	2.5 ± 0.4	NA	NA
2X	500	shallow	72.1 ± 46.2	**5.6 ± 1.1**	14.8 ± 2.7	15.8 ± 3.2	15.5 ± 2.5	NA	NA
2X	1000	shallow	317.8 ± 198.9	**32.4 ± 7.7**	59.2 ± 11.6	62.0 ± 9.2	63.5 ± 13.2	NA	NA
5X	200	deep	6.6 ± 7.7	**1.2 ± 0.0**	2.2 ± 0.1	2.3 ± 0.1	2.3 ± 0.1	NA	NA
5X	200	shallow	4.6 ± 7.9	**1.1 ± 0.0**	2.2 ± 0.0	2.3 ± 0.0	2.3 ± 0.0	NA	NA
250 estimated gene trees									
0.5X	200	deep	11.3 ± 4.8	0.8 ± 0.2	**0.7 ± 0.1**	**0.7 ± 0.1**	**0.7 ± 0.2**	NA	NA
0.5X	200	shallow	9.6 ± 3.4	0.7 ± 0.1	**0.6 ± 0.0**	0.7 ± 0.0	0.7 ± 0.0	NA	NA
2X	10	shallow	**0.0 ± 0.0**	0.1 ± 0.0	**0.0 ± 0.0**	**0.0 ± 0.0**	**0.0 ± 0.0**	0.1 ± 0.0 (0.98)	0.1 ± 0.0 (0.83)
2X	50	shallow	0.1 ± 0.0	0.1 ± 0.0	**0.0 ± 0.0**	**0.0 ± 0.0**	**0.0 ± 0.0**	4.1 ± 0.8 (0.99)	5.1 ± 1.0 (0.80)
2X	100	shallow	0.3 ± 0.2	**0.2 ± 0.0**	**0.2 ± 0.0**	**0.2 ± 0.0**	**0.2 ± 0.0**	65.4 ± 8.2 (1.00)	96.4 ± 12.5 (0.68)
2X	200	deep	1.7 ± 1.2	**0.5 ± 0.0**	0.6 ± 0.0	0.6 ± 0.0	0.6 ± 0.0	NA	NA
2X	200	shallow	1.1 ± 0.7	**0.5 ± 0.1**	0.6 ± 0.1	0.6 ± 0.1	0.6 ± 0.2	NA	NA
2X	500	shallow	7.1 ± 4.7	**3.0 ± 0.8**	3.7 ± 0.5	3.9 ± 0.7	3.9 ± 0.8	NA	NA
2X	1000	shallow	45.9 ± 50.4	22.4 ± 9.4	**15.2 ± 2.5**	15.8 ± 2.2	15.5 ± 1.5	NA	NA
5X	200	deep	0.6 ± 0.6	**0.4 ± 0.0**	0.6 ± 0.0	0.6 ± 0.0	0.6 ± 0.0	NA	NA
5X	200	shallow	0.5 ± 0.6	**0.4 ± 0.0**	0.6 ± 0.0	0.6 ± 0.0	0.6 ± 0.0	NA	NA

A.3 Additional Results on Simulated Data Sets

A.3.1 Number of taxa

Table A.4: **Testing for differences between TREE-QMC-n2 and other methods on ASTRAL-II data sets with varying numbers of taxa and 1000 estimated gene trees (Figure 2).** All data sets were simulated with 1X species tree height and shallow speciation. The table shows the number of replicates for which species tree estimation error was better for TREE-QMC-n2, better for the other method, or a tie as well as other metrics. The Δ false negative (FN) error is the difference in the number of false negative branches between the other method and TREE-QMC-n2, averaged ($\pm$ standard deviation) across all replicates (including ties). Positive values indicate that TREE-QMC-n2 is better on average, whereas negative values indicate that the other method is better on average. Differences between methods were evaluated using two-sided Wilcoxon signed-rank tests, after removing tied values [72] (note: the test was performed only if more than than 25 replicates were not ties). The symbol *, *, *** indicate significance at $p < 0.05$, 0.005, and 0.0005, respectively. Note that $p < 0.05/12 = 0.00416$ survives Bonferroni correction for the 12 comparisons made in this table (MC).

# of taxa	Other method	# of replicates TQMC-n2	OTHER	TIE	Δ FN error	p-value		MC
10	FASTRAL	0	0	49	0.00 ± 0.00	NA		
10	ASTRAL3	0	0	49	0.00 ± 0.00	NA		
50	FASTRAL	5	3	40	0.04 ± 0.41	NA		
50	ASTRAL3	5	3	40	0.04 ± 0.41	NA		
100	FASTRAL	9	8	31	0.04 ± 0.92	NA		
100	ASTRAL3	10	8	30	-0.08 ± 1.49	NA		
200	FASTRAL	20	6	24	0.44 ± 1.09	0.006217	*	N
200	ASTRAL3	20	6	24	0.48 ± 1.13	0.004217	**	Y
500	FASTRAL	34	9	7	2.40 ± 3.46	0.000009	***	Y
500	ASTRAL3	33	8	9	2.70 ± 3.89	0.000013	***	Y
1000	FASTRAL	32	10	5	3.94 ± 6.96	0.000123	***	Y
1000	ASTRAL3	35	8	4	4.64 ± 7.84	0.000008	***	Y

A.3.2 Species tree scale/height and thus ILS level

Table A.5: **Testing for differences between TREE-QMC-n2 and other methods on ASTRAL-II data sets with varying ILS levels and 1000 estimated gene trees (Figure 3)**. All data sets have 200 taxa. Note: rows with parentheses are duplicated from Table A.4.

species tree height	speciation	Other method	# of replicates TQMC-n2	OTHER	TIE	Δ FN error	p-value		MC
0.25X	deep	FASTRAL	22	21	7	-0.24 ± 3.46	0.70		
0.25X	deep	ASTRAL3	29	18	3	0.5 ± 3.52	0.10		
0.25X	shallow	FASTRAL	23	16	8	0.47 ± 2.24	0.16		
0.25X	shallow	ASTRAL3	23	15	9	0.77 ± 2.35	0.04414	*	N
1X	deep	FASTRAL	20	9	21	0.70 ± 2.15	0.04209	*	N
1X	deep	ASTRAL3	21	7	22	1.00 ± 2.25	0.00367	**	Y
(1X)	(shallow)	FASTRAL	20	6	24	0.44 ± 1.09	0.00622	*	N
(1X)	(shallow)	ASTRAL3	20	6	24	0.48 ± 1.13	0.00422	**	N
5X	deep	FASTRAL	27	6	17	1.10 ± 1.88	0.00032	***	Y
5X	deep	ASTRAL3	25	8	17	0.90 ± 2.26	0.00382	**	Y
5X	shallow	FASTRAL	13	8	29	0.44 ± 1.39	NA		
5X	shallow	ASTRAL3	13	9	28	0.42 ± 1.42	NA		

Table A.6: **Testing for differences between TREE-QMC-n2 and other methods on avian simulated data sets with species tree scales and thus ILS levels (Figure 4).** All data sets had 1000 gene trees. Differences between methods were evaluated using two-sided Wilcoxon signed-rank tests, after removing tied values [72] (note: the test was only performed if more than than 10 replicates were not ties). The symbol *, **, and *** indicate significance at $p < 0.05$, 0.005, and 0.0005. Note: $p < 0.05/18 = 0.00278$ would be significant after Bonferroni correction for the 18 comparisons made in this table (MC).

Gene trees	Other method	# of replicates TQMC-n2	OTHER	TIE	Δ FN error	p-value		MC
Figure 4A – 0.5X species tree scale								
estimated	wQFM	11	2	7	0.75 ± 1.12	0.011006	*	N
estimated	FASTRAL	8	3	9	0.75 ± 1.48	0.0461	*	N
estimated	ASTRAL3	18	2	0	2.10 ± 1.86	0.00101	**	Y
true	wQFM	5	7	8	-0.05 ± 1.340	0.94		
true	FASTRAL	5	8	7	-0.20 ± 1.24	0.56		
true	ASTRAL3	9	7	4	0.15 ± 1.42	0.56		
Figure 4B – 1X species tree scale								
estimated	wQFM	5	4	11	0.05 ± 0.69	NA		
estimated	FASTRAL	9	3	8	0.70 ± 1.34	0.0333	*	N
estimated	ASTRAL3	12	1	7	1.1 ± 1.21	0.00293	**	N
true	wQFM	2	6	12	-0.25 ± 0.72	NA		
true	FASTRAL	2	7	11	-0.30 ± 0.92	NA		
true	ASTRAL3	4	5	11	-0.05 ± 0.89	NA		
Figure 4C – 2X species tree scale								
estimated	wQFM	4	5	11	0.00 ± 0.80	NA		
estimated	FASTRAL	7	5	8	0.20 ± 0.95	0.36		
estimated	ASTRAL3	7	5	8	0.15 ± 0.88	0.46		
true	wQFM	0	0	20	0.00 ± 0.00	NA		
true	FASTRAL	2	1	17	0.05 ± 0.39	NA		
true	ASTRAL3	2	1	17	0.05 ± 0.39	NA		

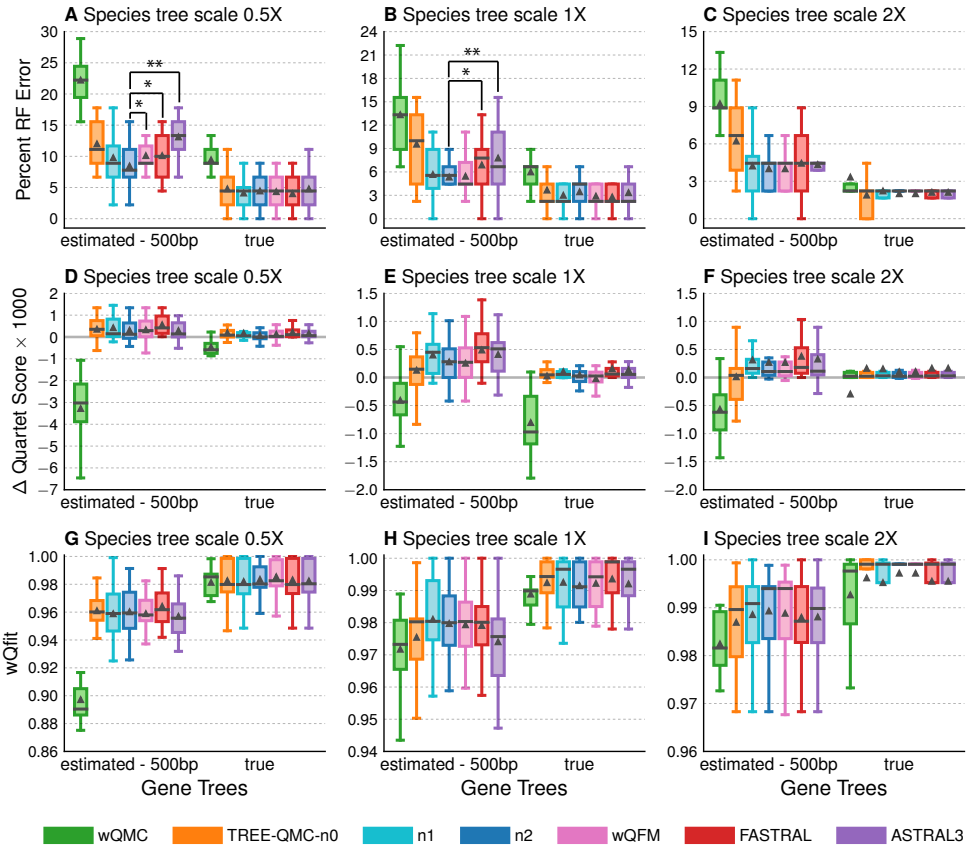


Figure A.2: Reproduction of Figure 4 in main text with different y-axis (so results for wQMC are not cut off) and additional results using the wQfit similarity score proposed by [7]. Subfigures A–C show the percent RF error between the tree and estimated species tree. Subfigures D–F show the difference in quartet score of the estimated versus true species tree (indicating how well the method solved MQSST and whether its solution is better than the true tree under the MQSST objective function). Subfigures G–I show the wQfit score, which compares the quartets in the true and estimated species trees, weighted by their frequency in the gene trees (note: the wQfit score is 1 if the true and estimated species trees have the same topology). Thus, wQfit evaluates a mixture of accuracy (as it depends on the true species tree topology) and MQSST solution quality (as it depends on quartet weights). Our results show that TREE-QMC achieves competitive wQfit scores with the leading methods (except for one model condition with species tree scale 1X and true gene trees; TREE-QMC is also less accurate for this model condition as shown in subfigure B). When gene trees are estimated, TREE-QMC outperforms the dominant method ASTRAL-III in terms of wQfit score on average. In subfigure G with estimated gene trees, TREE-QMC-n2, wQFM, FASTRAL, and ASTRAL-III have mean scores of 0.9602, 0.9589, 0.9642, 0.9570; thus, FASTRAL is the best method followed by TREE-QMC-n2, wQFM, and ASTRAL-III. For this condition, FASTRAL has higher error than TREE-QMC-n2 (mean: 10.1% vs. 8.3%) but also a higher quartet score (mean: 0.5686 vs. 0.5683). To our knowledge, the wQfit score has not been used in simulation studies evaluating summary methods when gene trees can differ from the species tree due to ILS and GTEE. Future studies should explore the utility of wQfit in this context, especially for conditions with high gene tree estimation error. For these data sets, the estimated gene trees have mean RF error of $\sim$60%.

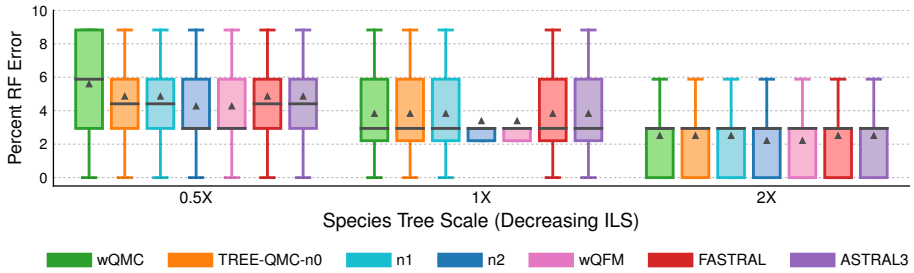


Figure A.3: **Species tree error and runtime for mammalian simulated data set with varying ILS levels.** All data sets have 200 estimated gene trees (sequence length: 500). Percent species tree error across replicates is shown using box plots (bars represent medians; triangles represent means; some outliers are not shown).

Table A.7: **Testing for differences between TREE-QMC-n2 and other methods on mammalian simulated data sets with varying species tree scales and thus ILS levels (Figure A.3).** All data sets had 200 estimated gene trees (sequence length: 500).

Species tree scale	Other method	# of replicates TQMC-n2	OTHER	TIE	Δ FN error	p-value
0.5X	wQFM	0	0	20	0.00 ± 0.00	NA
0.5X	FASTRAL	5	2	13	0.20 ± 0.70	NA
0.5X	ASTRAL3	5	2	13	0.20 ± 0.70	NA
1X	wQFM	0	0	20	0.00 ± 0.00	NA
1X	FASTRAL	5	2	13	0.15 ± 0.59	NA
1X	ASTRAL3	5	2	13	0.15 ± 0.59	NA
2X	wQFM	0	0	20	0.00 ± 0.00	NA
1X	FASTRAL	3	1	16	0.10 ± 0.45	NA
2X	ASTRAL3	3	1	16	0.10 ± 0.45	NA

A.3.3 Sequence length and thus GTEE level

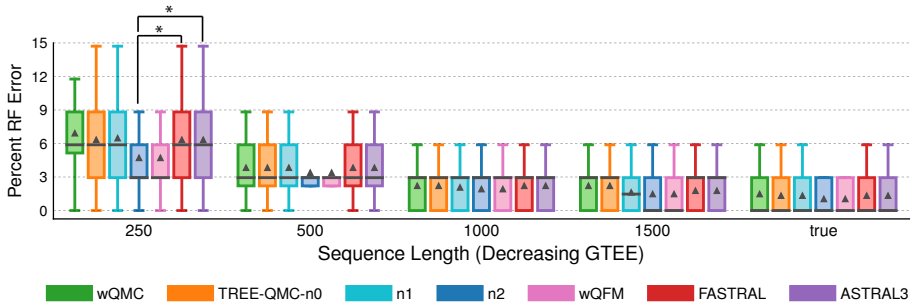


Figure A.4: **Species tree error and runtime for mammalian simulated data set with varying sequence lengths and GTEE levels.** All data sets were simulated from unscaled species trees (1X) and have 200 gene trees (note: results for sequence length 500 are duplicated from Figure A.3). Percent species tree error across replicates is shown using box plots (bars represent medians; triangles represent means; some outliers are not shown).

Table A.8: **Testing for differences between TREE-QMC-n2 and other methods on mammalian simulated data sets with varying sequence lengths and thus gene tree estimation error (Figure A.4).** All data sets were simulated with from unscaled species trees (1X) and have 200 gene trees. Note: rows with parentheses are duplicated from Table A.7.

Sequence length	Other method	# of replicates TQMC-n2	OTHER	TIE	Δ FN error	p-value	
250	wQFM	0	0	20	0.00 ± 0.00	NA	
250	FASTRAL	11	4	5	0.55 ± 1.10	0.04	*
250	ASTRAL3	11	4	5	0.55 ± 1.10	0.04	*
(500)	wQFM	0	0	20	0.00 ± 0.00	NA	
(500)	FASTRAL	5	2	13	0.15 ± 0.59	NA	
(500)	ASTRAL3	5	2	13	0.15 ± 0.59	NA	
1000	wQFM	0	0	20	0.00 ± 0.00	NA	
1000	FASTRAL	2	0	18	0.10 ± 0.31	NA	
1000	ASTRAL3	2	0	18	0.10 ± 0.31	NA	
1500	wQFM	0	0	20	0.00 ± 0.00	NA	
1500	FASTRAL	3	1	16	0.10 ± 0.45	NA	
1500	ASTRAL3	3	1	16	0.10 ± 0.45	NA	
true	wQFM	0	0	20	0.00 ± 0.00	NA	
true	FASTRAL	3	1	16	0.10 ± 0.45	NA	
true	ASTRAL3	3	1	16	0.10 ± 0.45	NA	

A.3.4 Number of Gene Trees

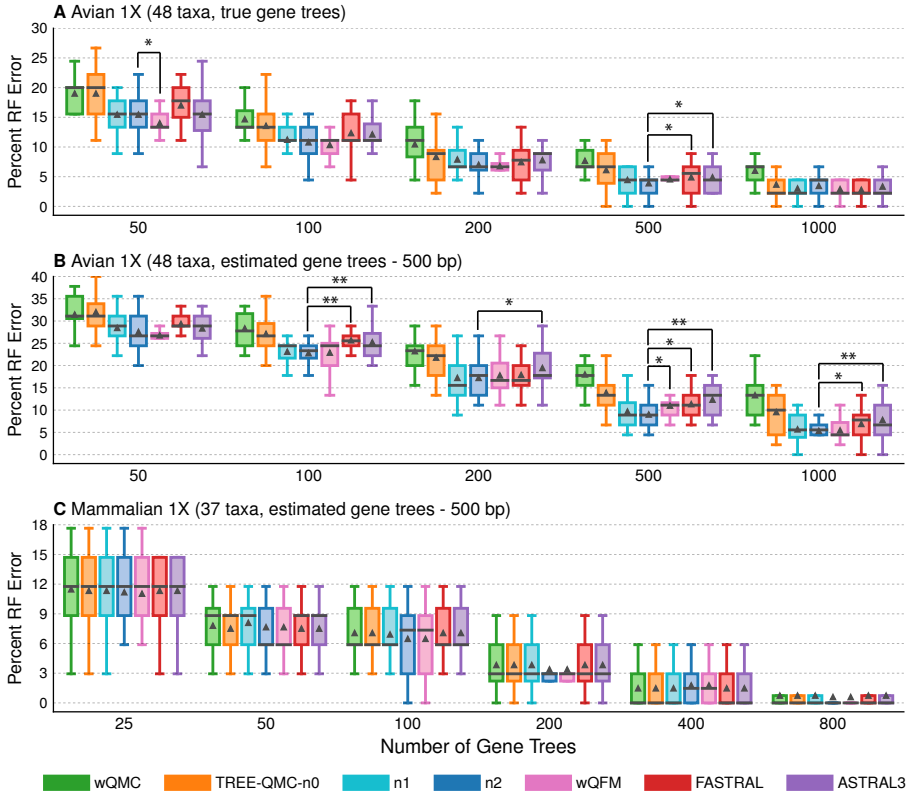


Figure A.5: **Species tree error and runtime for avian and mammalian simulated data sets.** Bars represent medians, and triangles represent means (outliers are not shown). The symbols *, **, and *** indicate significance at $p < 0.05$, 0.005, and 0.0005, respectively (see Tables A.9 and A.10 for details). All data sets have unscaled species trees (1X). Note: the results for avian simulated data sets with 1000 true gene trees are duplicated from Figure 4A–C, and the results for mammalian simulated data sets with 200 estimated gene trees (sequence length: 500) are duplicated from Figures A.3 and A.4.

Table A.9: **Testing for differences between TREE-QMC-n2 and other methods on avian simulated data sets with varying numbers of gene trees (Figure A.5).** Note: rows with parentheses are duplicated from Table A.6.

# of genes	Other method	# of replicates TQMC-n2	OTHER	TIE	Δ FN error	p-value	
Figure A.5A – Avian 1X with true gene trees							
50	wQFM	2	10	8	-0.65 ± 1.18	0.03714	*
50	FASTRAL	11	5	4	0.70 ± 1.56	0.07448	
50	ASTRAL3	8	8	4	0.00 ± 1.41	1.00	
100	wQFM	7	6	7	-0.20 ± 1.24	0.47	
100	FASTRAL	9	3	8	0.70 ± 1.56	0.07745	
100	ASTRAL3	9	4	7	0.60 ± 1.31	0.06917	
200	wQFM	2	5	13	-0.10 ± 0.97	NA	
200	FASTRAL	8	5	7	0.20 ± 1.20	0.49	
200	ASTRAL3	9	4	7	0.35 ± 1.23	0.25	
500	wQFM	5	0	15	0.30 ± 0.57	NA	
500	FASTRAL	10	3	7	0.45 ± 0.89	0.04249	*
500	ASTRAL3	10	3	7	0.45 ± 0.89	0.04249	*
(1000)	wQFM	2	6	12	-0.25 ± 0.72	NA	
(1000)	FASTRAL	2	7	11	-0.30 ± 0.92	NA	
(1000)	ASTRAL3	4	5	11	-0.05 ± 0.89	NA	
Figure A.5B – Avian 1X with estimated gene trees							
50	wQFM	3	8	9	-0.35 ± 1.66	0.47	
50	FASTRAL	11	5	4	0.80 ± 2.40	0.16	
50	ASTRAL3	8	6	6	0.35 ± 2.48	0.51	
100	wQFM	7	8	5	0.05 ± 1.67	0.93	
100	FASTRAL	13	1	6	1.30 ± 1.49	0.00202	**
100	ASTRAL3	13	2	5	1.10 ± 1.37	0.00276	**
200	wQFM	8	5	7	0.25 ± 1.41	0.39	
200	FASTRAL	9	6	5	0.30 ± 1.53	0.35	
200	ASTRAL3	11	4	5	1.00 ± 1.75	0.02670	*
500	wQFM	11	2	7	0.90 ± 1.45	0.01932	*
500	FASTRAL	11	3	6	1.05 ± 1.88	0.02633	*
500	ASTRAL3	15	2	3	1.50 ± 1.50	0.00112	**
(1000)	wQFM	5	4	11	0.05 ± 0.69	NA	
(1000)	FASTRAL	9	3	8	0.70 ± 1.34	0.03329	*
(1000)	ASTRAL3	12	1	7	1.10 ± 1.21	0.00293	**

Table A.10: **Testing for differences between TREE-QMC-n2 and other methods on mammalian simulated data sets with varying numbers of estimated gene trees (Figure A.5).** Note: rows with parentheses are duplicated from Tables A.7 and A.8.

# of genes	Other method	# of replicates TQMC-n2	OTHER	TIE	Δ FN error	p-value
Figure A.5C – Mammalian 1X with estimated gene trees (500 bp)						
25	wQFM	0	1	19	-0.05 ± 0.22	NA
25	FASTRAL	4	5	11	0.05 ± 0.89	NA
25	ASTRAL3	4	5	11	0.05 ± 0.89	NA
50	wQFM	0	0	20	0.00 ± 0.00	NA
50	FASTRAL	6	5	9	-0.05 ± 0.95	0.81
50	ASTRAL3	6	5	9	-0.05 ± 0.95	0.81
100	wQFM	0	0	20	0.00 ± 0.00	NA
100	FASTRAL	7	3	10	0.20 ± 0.70	NA
100	ASTRAL3	7	3	10	0.20 ± 0.70	NA
(200)	wQFM	0	0	20	0.00 ± 0.00	NA
(200)	FASTRAL	5	2	13	0.15 ± 0.59	NA
(200)	ASTRAL3	5	2	13	0.15 ± 0.59	NA
400	wQFM	0	0	20	0.00 ± 0.00	NA
400	FASTRAL	0	2	18	-0.10 ± 0.31	NA
400	ASTRAL3	0	2	18	-0.10 ± 0.31	NA
800	wQFM	0	0	20	0.00 ± 0.00	NA
800	FASTRAL	1	0	19	0.05 ± 0.22	NA
800	ASTRAL3	1	0	19	0.05 ± 0.22	NA

A.4 Additional Results on Biological Data Sets

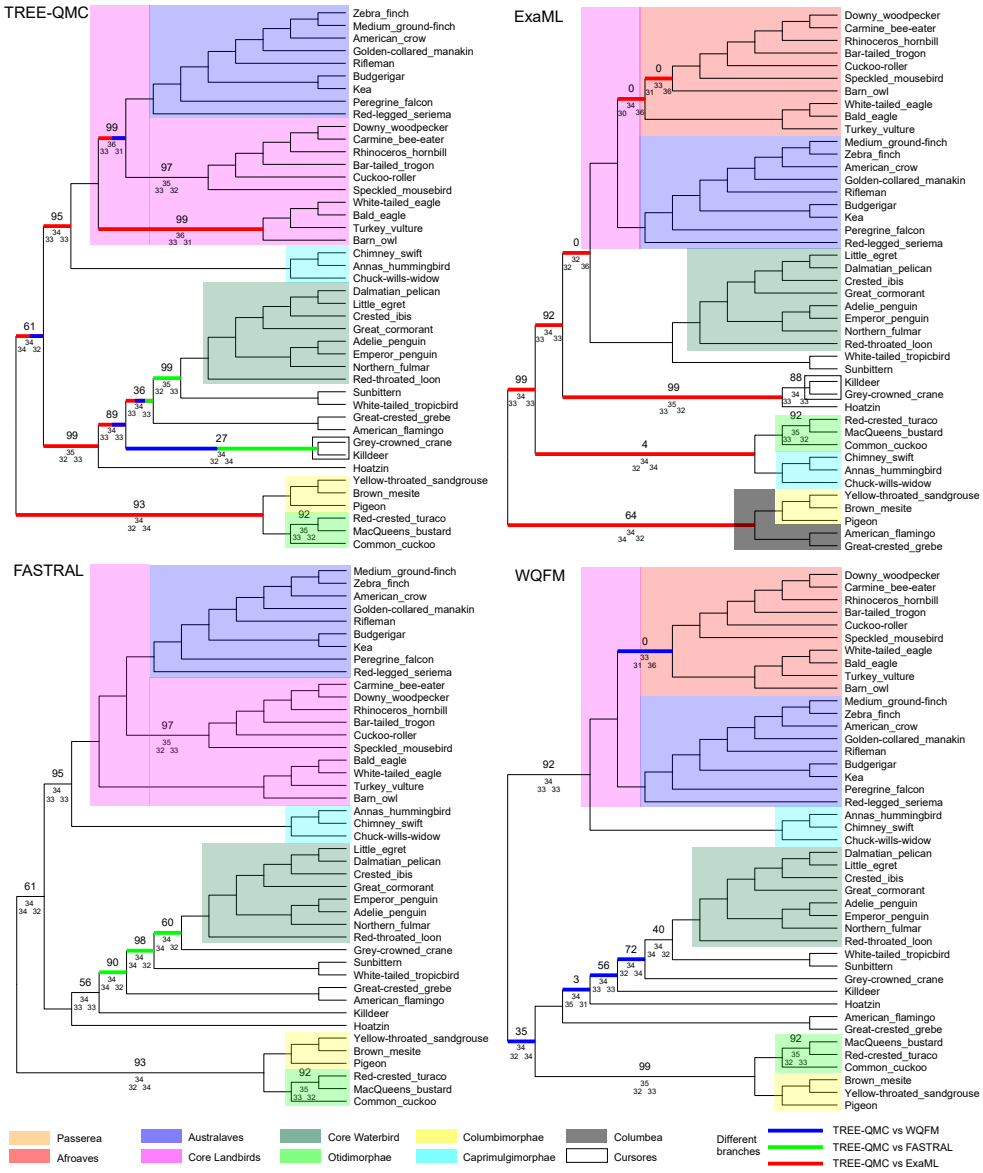


Figure A.6: Species trees estimated on TENT data by TREE-QMC-n2, ExaML, FASTRAL, and wQFM. Above the branch, we show support values estimated using ASTRAL's local posterior probability (multiplied by 100). Only support values less than 100 are shown. Below the branch, we show the quartet support (the two values below it correspond to quartet support for the two alternative resolutions of the branch). Importantly, these support calculations are based on estimated gene trees. Taxa outside of Neoaves are not shown as all methods recovered the same topology outside of Neoaves.

A.5 Additional Plots for ASTRAL-II Data Sets

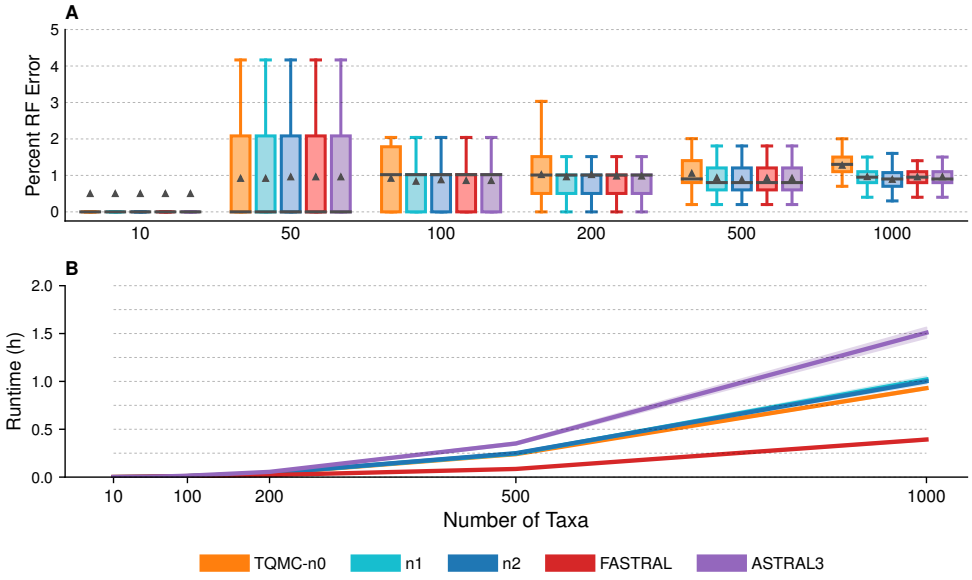


Figure A.7: **Species tree error and runtime for ASTRAL-II data sets with varying number of taxa (1000 true gene trees).** (A) Percent species tree error across replicates (bars represent medians; triangles represent means; some outliers are not shown). No statistical tests performed. (B) Mean runtime across replicates (shaded region indicates standard error). All data sets were simulated with 1X species tree height and shallow speciation.

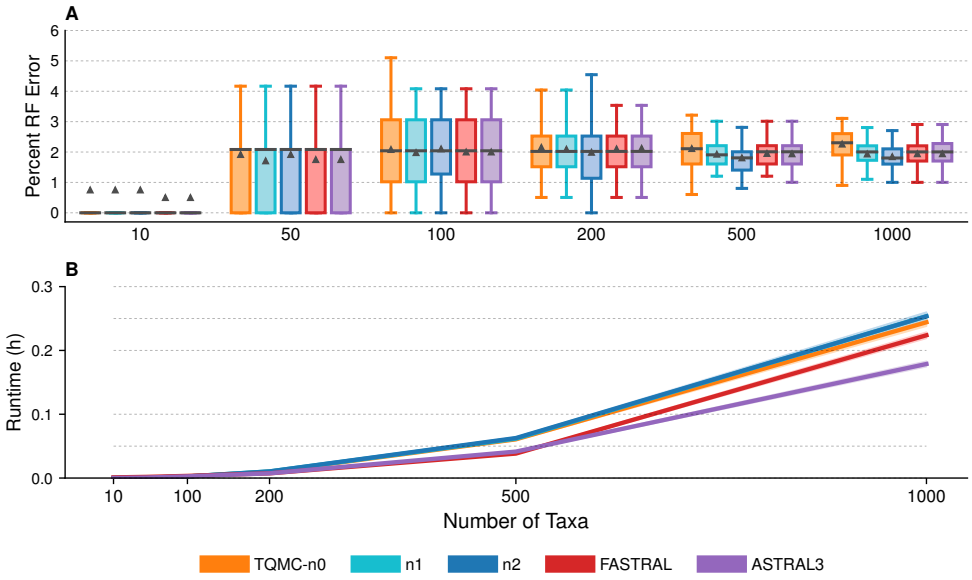


Figure A.8: **Species tree error and runtime for ASTRAL-II data sets with varying numbers of taxa (250 true gene trees).** (A) Percent species tree error across replicates (bars represent medians; triangles represent means; some outliers are not shown). No statistical tests performed. (B) Mean runtime across replicates (shaded region indicates standard error). All data sets were simulated with 1X species tree height and shallow speciation.

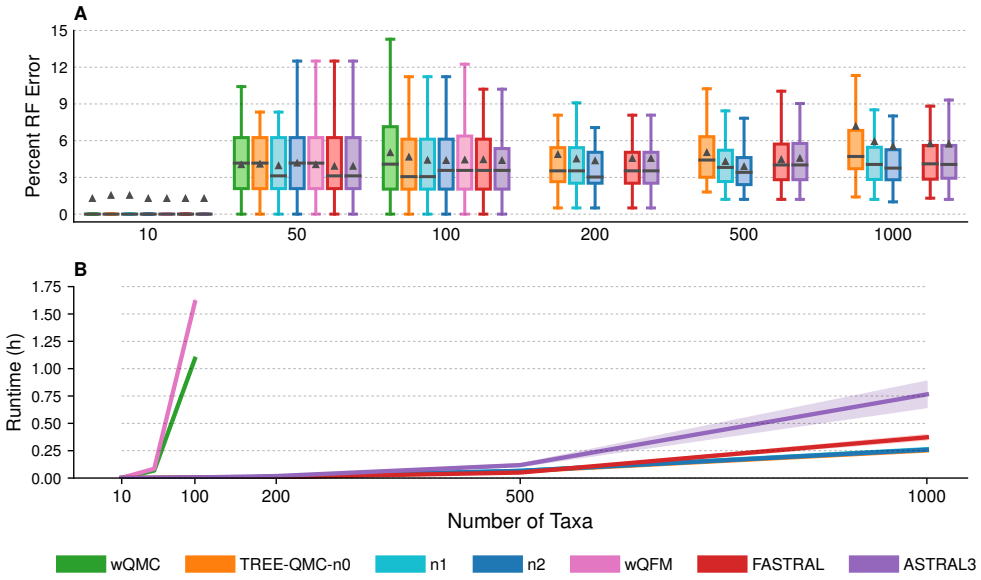


Figure A.9: **Species tree error and runtime for ASTRAL-II data sets with varying numbers of taxa (250 estimated gene trees).** (A) Percent species tree error across replicates (bars represent medians; triangles represent means; some outliers are not shown). No statistical tests performed. (B) Mean runtime across replicates (shaded region indicates standard error). All data sets were simulated with 1X species tree height and shallow speciation.

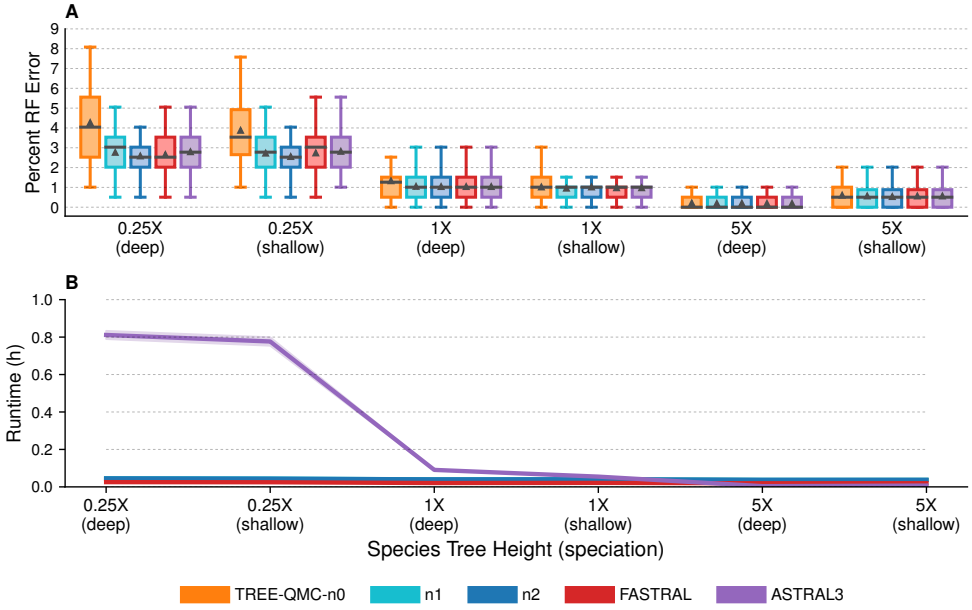


Figure A.10: **Species tree error and runtime for ASTRAL-II data sets with varying ILS levels (1000 true gene trees).** (A) Percent species tree error across replicates (bars represent medians; triangles represent means; some outliers are not shown). No statistical tests performed. (B) Mean runtime across replicates (shaded region indicates standard error). All data sets have 200 taxa. One model condition with species tree height 1X and shallow speciation is repeated from Figure A.7.

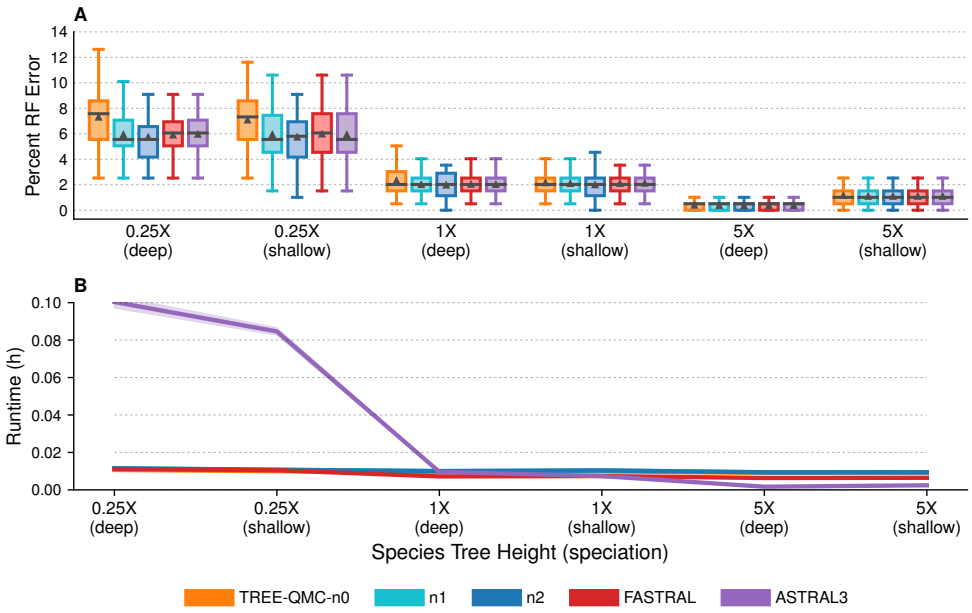


Figure A.11: **Species tree error and runtime for ASTRAL-II data sets with varying ILS levels (250 true gene trees).** (A) Percent species tree error across replicates (bars represent medians; triangles represent means; some outliers are not shown). No statistical tests performed. (B) Mean runtime across replicates (shaded region indicates standard error). All data sets have 200 taxa. One model condition with species tree height 1X and shallow speciation is repeated from Figure A.8.

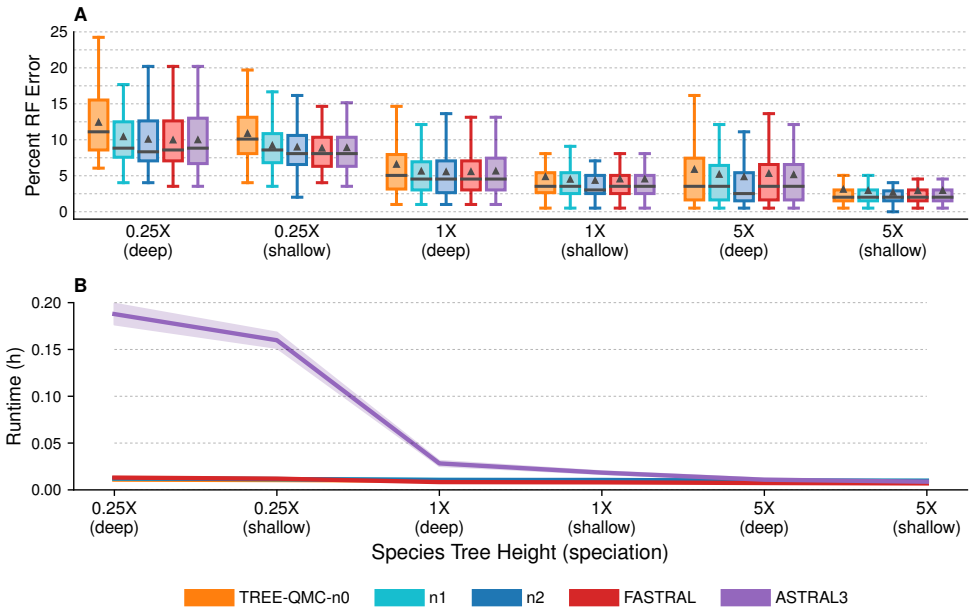


Figure A.12: **Species tree error and runtime for ASTRAL-II data sets with varying ILS levels (250 estimated gene trees).** (A) Percent species tree error across replicates (bars represent medians; triangles represent means; some outliers are not shown). No statistical tests performed. (B) Mean runtime across replicates (shaded region indicates standard error). All data sets have 200 taxa. One model condition with species tree height 1X and shallow speciation is repeated from Figure A.9.

A.6 TREE-QMC Algorithm

A.6.1 Quartet Graph Construction Case 1 cont. (Two Singletons)

To compute the normalized version of $\mathbb{B}[X,Y]$ using the previous algorithm, we set $c_D[v]$ to be the sum of the importance values of the leaves below vertex v that are labeled by D (i.e., $c_D[v] = \sum_{m \in L(v), M=D} I(m)$ where $L(v)$ denotes the set of leaves below v). The proof of correctness follows from Lemma A.1, in which we show that the total weight of selecting two uniquely labeled leaves below vertex u equals $g_0[u]$. All other quantities $(p, \mathbb{A}, \mathbb{L}, \mathbb{R})$ are computed from $g_0[u]$.

Lemma A.1. *The total weight of all taxon pairs in the subtree rooted at internal vertex u*

$$\sum_{\substack{z,w \in L(u): \\ Z \neq W}} I(z)I(w) = g_0[u] \tag{A.1}$$

where $L(u)$ is the set of leaves below u.

Proof. Let $\mathscr{S}(u)$ be the set of singletons below vertex u, and let $\mathscr{A}(u)$ be the set of artificial taxa

below u. Then,

$$\sum_{\substack{z,w\in L(u):\\ Z\neq W}} I(z)I(w) = \sum_{\substack{z,w\in \mathscr{L}(u):\\ Z,W\in \mathscr{S}(u), Z\neq W}} I(z)I(w) + \sum_{\substack{z,w\in L(u):\\ Z\in \mathscr{S}(u), W\in \mathscr{A}(u)}} I(z)I(w) + \sum_{\substack{z,w\in L(u):\\ Z,W\in \mathscr{A}(u), Z\neq W}} I(z)I(w)$$

$$\begin{aligned}
&= \sum_{\substack{z,w\in L(u):\\ Z,W\in \mathscr{S}(u), Z\neq W}} 1 \\
&\quad + \left(\sum_{\substack{z\in L(u):\\ Z\in \mathscr{S}(u)}} I(z)\right)\cdot\left(\sum_{\substack{w\in L(u):\\ W\in \mathscr{A}(u)}} I(w)\right) \\
&\quad + \frac{\left(\sum_{Z\in \mathscr{A}(u)}\sum_{m\in L(u):M=Z} I(m)\right)^2 - \left(\sum_{Z\in \mathscr{A}(u)}\left(\sum_{m\in L(u):m=Z} I(m)\right)^2\right)}{2} \\
&= \binom{c_0[u]}{2} + c_0[u]\cdot G_1[u] + \frac{G_1[u]^2 - G_2[u]}{2} \\
&= g_0[u]
\end{aligned}$$

where $G_1[u] = \sum_{D\in \mathscr{A}(u)} c_D[u]$, and $G_2[u] = \sum_{D\in \mathscr{A}(u)} c_D[u]^2$. □

Lastly, we need to compute the good edges $\mathbb{G}[X,Y]$, which is the total weight of quartets in which X,Y are not siblings. This can be done in constant time, following Lemma A.2.

Lemma A.2. *Let T be a multi-labeled gene tree T, and let X,Y be singletons. Then,*

$$\mathbb{G}[X,Y] + \mathbb{B}[X,Y] = \binom{c_0[r]-2}{2} + (c_0[r]-2)\cdot G_1[r] + \frac{G_1[r]^2 - G_2[r]}{2} \tag{A.2}$$

where r is the root of T.

Proof.

$$\begin{aligned}
\mathbb{G}[X,Y]+\mathbb{B}[X,Y] &= \sum_{\substack{z,w\in L(r):\\ Z\neq W\neq X\neq Y}} I(x)I(y)I(z)I(w) \\
&= \sum_{\substack{z,w\in L(r):\\ Z\neq W\neq X\neq Y}} I(z)I(w) \\
&= \sum_{\substack{z,w\in L(u):\\ Z,W\in\mathscr{S}(u)\setminus\{X,Y\},\\ Z\neq W}} I(z)I(w) + \sum_{\substack{z,w\in L(u):\\ Z\in\mathscr{S}(u)\setminus\{X,Y\},\\ W\in\mathscr{A}(u)}} I(z)I(w) + \sum_{\substack{z,w\in L(u):\\ Z,W\in\mathscr{A}(u),\\ Z\neq W}} I(z)I(w) \\
&= \binom{c_0[r]-2}{2} + (c_0[r]-2)\cdot G_1[r] + \frac{G_1[r]^2 - G_2[r]}{2}
\end{aligned}$$

□

A.6.2 Quartet Graph Construction Case 2 (Singleton and Artificial Taxon)

In this section, we consider the computation of $\mathbb{B}[X,Y]$ when X is a singleton and Y is an artificial taxon (or vice versa). Our previous algorithm computed $\mathbb{B}[X,Y]$ when it reached the lowest common ancestor (LCA) X and Y during a postorder traversal. However, there are now multiple multiple leaves labeled Y and thus there can be multiple LCAs. Alternatively, we might count the number of quartets with X and Y as siblings that have vertex v as their LCA, meaning that X labels a leaf below vertex $v.left$ and Y labels a leaf below vertex $v.right$ or vice versa. We let $\mathbb{B}_v[X,Y]$ denote the first quantity and $\mathbb{B}_v[Y,X]$ denote the second quantity, defining $\mathbb{A}_v[X,Y]$, $\mathbb{L}_v[X,Y]$, and $\mathbb{R}_v[X,Y]$ in the natural way. Now the total number of quartets with X,Y as siblings can be computed as

$$\mathbb{B}[X,Y] = \mathbb{B}[Y,X] = \sum_{v\in V(T)} (\mathbb{B}_v[X,Y] + \mathbb{B}_v[Y,X]) \tag{A.3}$$

Without loss of generality, we show how to compute $\mathbb{B}_v[X,Y]$, where X is a singleton and Y is

an artificial taxon. Let's assume this quantity is not zero so X labels *exactly one* leaf below $v.left$ and Y labels *at least one* leaf below $v.right$. In our previous algorithm, we utilized an approach for counting number of ways to select two z, w below vertex u such that $Z \neq W$. Now we must additionally require $Z \neq Y$ because Y already labels a leaf in the quartet. This gives us the modified binomial coefficient

$$g_Y[u] = \binom{c_0[u]}{2} + \left(c_0[u] \cdot (G_1[u] - c_Y[u])\right) + \left(\frac{(G_1[u] - c_Y[u])^2 - (G_2[u] - c_Y[u]^2)}{2}\right) \tag{A.4}$$

from which can compute the modified prefix of v: $p_Y[v] = p_Y[v.parent] + g_Y[v.sibling]$. Now we compute a vector g of size $b+1$ and a vector p of size b but note that the time complexity of preprocessing phase is still $O(bn)$. Afterward, we can compute the quantities:

$$\mathbb{A}_v[X,Y] = c_Y[v.right] \cdot g_Y[v.above] \tag{A.5}$$

$$\mathbb{L}_v[X,Y] = c_Y[v.right] \cdot (p_Y[x] - p_Y[v.left]) \tag{A.6}$$

These are the same formulas given in the appendix, except that we now exclude Y when taking the modified binomial coefficient and then multiply by $c_Y[v.right]$ (because we can select any leaf labeled Y below $v.right$).

Computing $\mathbb{R}_v[X,Y]$ is more complicated, and thus, we define a *different* dynamic programming algorithm that counts triplets of the form $((W,Z),Y)$. Specifically, we define

$$\mathbb{R}_v[X,Y] = f_{Y,0}[v.right] \tag{A.7}$$

where $f_{Y,0}[u]$ is the number of triplets below vertex $u = v.right$ of the form $((W,Z),Y)$. This quantity can be computed in $O(n)$ time a preorder traversal, after initializing $f_{y,0}[leaf] = 0$, as

follows:

$$f_{Y,0}[u] = f_{Y,0}[u.left] + f_{Y,0}[u.right] + \Big(c_Y[u.left] \cdot g_Y[u.right]\Big) + \Big(c_Y[u.right] \cdot g_Y[u.left]\Big) \tag{A.8}$$

where the first and second term come from selecting three leaves below $u.left$ and $u.right$, respectively (these triplets have already been counted). The last two terms come from counting triplets whose LCA is u. We repeat this preprocessing to compute f for all b artificial taxa; thus, the total time complexity of the preprocessing phase is still $O(bn)$.

After the preprocessing phase, we can compute $\mathbb{B}_v[X,Y]$ in constant time (a similar argument can be made for $\mathbb{B}_v[Y,X]$). Both of these quantities will equal zero (and do not need to be computed) for vertices that are not on the path from the singleton X to the root; thus, we can compute $\mathbb{B}[X,Y]$ in $O(n)$ time. Repeating this calculation for all ways of selecting one singleton and one artificial taxon gives us the final time complexity: $O(abh+bn)$.

A.6.3 Quartet Graph Construction Case 3 (Two Artificial Taxa)

Lastly, we consider the computation of $\mathbb{B}[X,Y]$ when X and Y are both artificial taxa. This largely proceeds as previously described in the previous Section. First, we update the binomial coefficient to exclude both X and Y (instead of just Y) as follows:

$$\begin{aligned} g_{X,Y}[v] = &\binom{c_0[v]}{2} + \Big(c_0[v] \cdot (G_1[v] - c_Y[v] - c_X[v])\Big) \\ &+ \left(\frac{(G_1[v] - c_Y[v] - c_X[v])^2 - (G_2[v] - c_Y[v]^2 - c_X[v]^2)}{2}\right) \end{aligned} \tag{A.9}$$

Second, we update the quantity for counting triplets

$$f_{Y,X}[v] = f_{Y,X}[v.left] + f_{Y,X}[v.right] + c_Y[v.left] \cdot g_{X,Y}[v.right] + c_Y[v.right] \cdot g_{X,Y}[v.left] \tag{A.10}$$

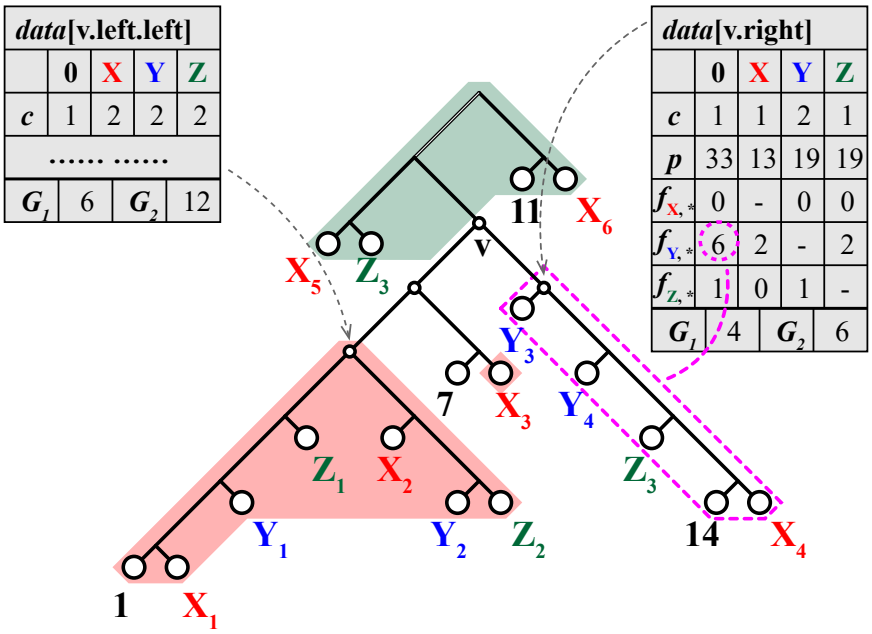


Figure A.13: $\mathbb{B}_v$ **with one artificial taxon.** Above is an example of the data stored to compute $\mathbb{B}_v[7,Y]$, where 7 is a singleton in $v.left$ and Y is an artificial taxa in $v.right$. $\mathbb{B}_v[7,Y] = \mathbb{A}_v[7,Y] + \mathbb{L}_v[7,Y] + \mathbb{R}_v[7,Y] = 2 \cdot 5 + 2 \cdot 8 + 6 = 32$.

Third, we update quantities associated with $\mathbb{B}_v[X,Y]$, specifically

$$\mathbb{A}_v[X,Y] = c_X[v.left] \cdot c_Y[v.right] \cdot g_{X,Y}[v.above] \tag{A.11}$$

$$\mathbb{L}_v[X,Y] = c_Y[v.right] \cdot f_{X,Y}[v.left] \tag{A.12}$$

$$\mathbb{R}_v[X,Y] = c_X[v.left] \cdot f_{Y,X}[v.right] \tag{A.13}$$

The key thing to note is that we are now storing a matrix of size b^2 at each vertex v. This brings the preprocessing phase to $O(b^2n)$, after which we can compute $\mathbb{B}_v[X,Y]$ in constant time. This quantity could be non-zero for any internal vertex of T so the total time to compute $\mathbb{B}[X,Y]$ is once again $O(n)$. We repeat this calculation for all ways of selecting one two artificial taxa to get the total time complexity is $O(b^2n)$.

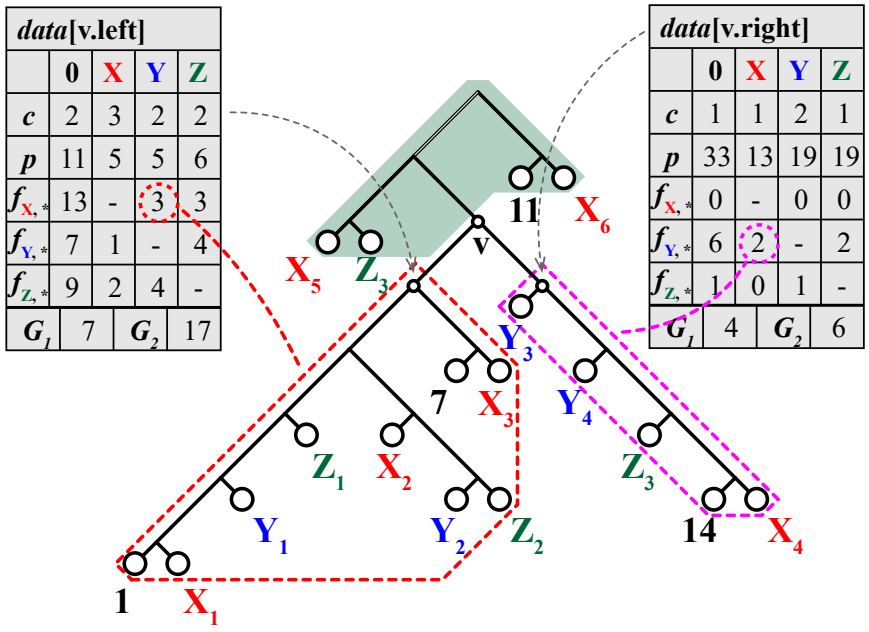


Figure A.14: $\mathbb{B}_v$ **with two artificial taxa.** In subfigure b), we show how to compute $\mathbb{B}_v[X,Y]$ where both X and Y are artificial. Again, $\mathbb{B}_v[X,Y]$ is given by the sum $\mathbb{A}_v[X,Y]$, $\mathbb{L}_v[X,Y]$, and $\mathbb{R}_v[X,Y]$. $\mathbb{A}_v[X,Y]$ is still derived from the green subtree above v except that not only Y but also X is excluded. Consequently, $g_{X,Y}[v.above] = 1$ since there are only two taxa in the subtree except X, and $\mathbb{A}_v[X,Y]$ is therefore $c_X[v.right] \cdot c_Y[v.right] \cdot g_{X,Y}[v.above] = 6$. On the other hand, $\mathbb{L}_v[X,Y]$ is $c_Y[v.right] \cdot f_{X,Y}[v.left] = 2 \times 3 = 6$ while $\mathbb{R}_v[X,Y] = 3 \times 2 = 6$ because $c_X[v.left] = 3$ and $f_{Y,X}[v.right] = 2$. Finally, we have $\mathbb{B}_v[X,Y] = 6+6+6 = 18$. Similarly, $\mathbb{B}_v[Y,X] = 2+1+0 = 3$ because $c_Y[v.left] = 2$, $c_X[v.right] = 1$, $g_{X,Y}[v.above] = 1$, $f_{X,Y}[v.right] = 0$ and $f_{Y,X}[v.left] = 1$.

A.6.4 Algorithms

Algorithm A.1: Initialization

Data: the input gene tree T and pre-computed $c[v]$ and $g[v]$ values
Result: the $p[v]$ and $f[v]$ values for each node v in T

for *each node v of T in the pre-order* **do**
  if *v is the root* **then**
    $p_0[v] \leftarrow 0$
    for *each artificial taxa X* **do**
      $p_X[v] \leftarrow 0$
  else
    if *v is not a leaf* **then**
      $p_0[v] \leftarrow p_0[v.parent] + g_0[v.sibling]$
      for *each artificial taxa X* **do**
        $p_X[v] \leftarrow p_X[v.parent] + g_X[v.sibling]$
for *each node v of T in the post-order* **do**
  if *v is a leaf* **then**
    for *each artificial taxa Y* **do**
      $f_{Y,0}[v] \leftarrow 0$
    for *each artificial taxa Y* **do**
      for *each artificial taxa* $X \neq Y$ **do**
        $f_{Y,X}[v] \leftarrow 0$
  else
    for *each artificial taxa Y* **do**
      $f_{Y,0}[v] \leftarrow$
      $f_{Y,0}[v.left] + f_{Y,0}[v.right] + c_Y[v.left] \cdot g_Y[v.right] + c_Y[v.right] \cdot g_Y[v.left]$
    for *each artificial taxa Y* **do**
      for *each artificial taxa* $X \neq Y$ **do**
        $f_{Y,X}[v] \leftarrow f_{Y,X}[v.left] + f_{Y,X}[v.right] + c_Y[v.left] \cdot g_{X,Y}[v.right] +$
        $c_Y[v.right] \cdot g_{X,Y}[v.left]$

Algorithm A.2: Quartet Graph Construction – Bad Edges

Data: the input gene tree T with all pre-computed values
Result: $\mathbb{B}[X,Y]$: the weights of the bad edges between each pair of X and Y

for *each internal node v in* T **do**
  for *each taxa* X *below* $v.left$ **do**
    for *each taxa* Y *below* $v.right$ *such that* $X \neq Y$ **do**
      if X *is an artificial taxa* **then**
        if Y *is an artificial taxa* **then**
          $\mathbb{A}_v[X,Y] \leftarrow c_X[v.left] \cdot c_Y[v.right] \cdot g_{X,Y}[v.above]$
          $\mathbb{L}_v[X,Y] \leftarrow c_Y[v.right] \cdot f_{X,Y}[v.left]$
          $\mathbb{R}_v[X,Y] \leftarrow c_X[v.left] \cdot f_{Y,X}[v.right]$
        else
          $\mathbb{A}_v[X,Y] \leftarrow c_Y[v.right] \cdot g_Y[v.above]$
          $\mathbb{L}_v[X,Y] \leftarrow c_Y[v.right] \cdot (p_Y[x] - p_Y[v.left])$ where y is the unique leaf labeled Y
          $\mathbb{R}_v[X,Y] \leftarrow f_{Y,0}[v.right]$
      else
        if Y *is an artificial taxa* **then**
          $\mathbb{A}_v[X,Y] \leftarrow c_X[v.left] \cdot g_X[v.above]$
          $\mathbb{L}_v[X,Y] \leftarrow f_{X,0}[v.left]$
          $\mathbb{R}_v[X,Y] \leftarrow c_X[v.left] \cdot (p_X[y] - p_X[v.right])$
        else
          $\mathbb{A}_v[X,Y] \leftarrow g_0[v.above]$
          $\mathbb{L}_v[X,Y] \leftarrow p_0[x] - p_0[v.left]$
          $\mathbb{R}_v[X,Y] \leftarrow p_0[y] - p_0[v.right]$
      $\mathbb{B}[X,Y] \leftarrow \mathbb{B}[X,Y] + \mathbb{A}_v[X,Y] + \mathbb{L}_v[X,Y] + \mathbb{R}_v[X,Y]$
      $\mathbb{B}[X,Y] \leftarrow \mathbb{B}[X,Y] + \mathbb{A}_v[X,Y] + \mathbb{L}_v[X,Y] + \mathbb{R}_v[X,Y]$

Algorithm A.3: Quartet Graph Construction – Good Edges

Data: the input gene tree T with root r, all pre-computed values for T, and $\mathbb{B}$
Result: $\mathbb{G}[X,Y]$: the weights of the good edges between each pair of X and Y

for *each taxa X* **do**
  for *each taxa Y* **do**
    if *X is an artificial taxa* **then**
      if *Y is an artificial taxa* **then**
        $\mathbb{G}[X,Y] = g_{X,Y}[r] \cdot c_X[r] \cdot c_Y[r] - \mathbb{B}[X,Y]$
      else
        $\mathbb{G}[X,Y] = \left(\binom{c_0[r]-1}{2} + \left((c_0[r]-1) \cdot (G_1[v] - c_X[v]) \right) + \left(\frac{(G_1[r]-c_X[r])^2 - (G_2[r]-c_X[r]^2)}{2} \right) \right) \cdot c_X[r] - \mathbb{B}[X,Y]$
    else
      if *Y is an artificial taxa* **then**
        $\mathbb{G}[X,Y] = \left(\binom{c_0[r]-1}{2} + \left((c_0[r]-1) \cdot (G_1[r] - c_Y[r]) \right) + \left(\frac{(G_1[r]-c_Y[r])^2 - (G_2[r]-c_Y[r]^2)}{2} \right) \right) \cdot c_Y[r] - \mathbb{B}[X,Y]$
      else
        $\mathbb{G}[X,Y] = \binom{c_0[r]-2}{2} + (c_0[r]-2) \cdot G_1[r] + \frac{G_1[r]^2 - G_2[r]}{2} - \mathbb{B}[X,Y]$

A.6.5 Time Complexity

Theorem A.1. *Let n leaves in the gene tree T, and let s be the number of leaf labels for the current subproblem. Then, we compute $\mathbb{B}$ and $\mathbb{G}$ in $O(s^2n)$ time using Algorithms A.1–A.3.*

Proof. Let a be the number of singletons in the subproblem, and let b be the number of artificial taxa (so $a+b=s$). To compute $\mathbb{B}$, we need to compute $\mathbb{A},\mathbb{L},\mathbb{R}$, which in turn depend on other quantities that are computed during a preprocessing phase.

Preprocessing phase: Computing the **number of taxa** below vertex v:

- $c_0[v]$ takes $O(n)$ time
- $c_Y[v]$ for each artificial taxon Y takes $O(bn)$ time
- all $G_1[v]$ and $G_2[v]$ takes $O(bn)$ time

via a post-order traversal on T. Computing the **modified binomial coefficients**:

- all $g_0[v]$ takes $O(n)$ time
- all $g_Y[v]$ for each artificial taxon Y takes $O(bn)$ time
- all $g_{X,Y}[v]$ for each pair X,Y of artificial taxa takes $O(b^2n)$ time

using $G_1[v]$ and $G_2[v]$. Computing the **prefixes**

- $p_0[v]$ takes $O(n)$ time
- $p_X[v]$ for each artificial taxon X takes $O(bn)$ time

see lines 1-10 of Algorithm A.1. Computing the **triplet counters**

- $f_{Y,0}[v]$ for each artificial taxon Y takes $O(bn)$ time

- $f_{Y,X}[v]$ for each pair X, Y of artificial taxa takes $O(b^2 n)$ time

see lines 11-23 of Algorithm A.1. To sum up, the total initialization time is $O(b^2 n)$ for each gene tree.

Now we can compute the weights of the **bad edges** $\mathbb{B}$ via Algorithm A.2. Each iteration of Algorithm A.2 (lines 4-23) costs $O(1)$ time. In other words, given any triple of v, X, and Y, the corresponding $\mathbb{B}_v[X,Y]$ is computed in constant time. Consequently, the total time complexity depends on the number of the distinct tripples (v,X,Y). If both X and Y are singletons, then there is only one unique v at which $\mathbb{B}[X,Y]$ or $\mathbb{B}[Y,X]$ are updated because v must be the LCA of X and Y. As a result, the triplet of (v,X,Y) is unique when X and Y are singleton, and the number of triplets for all singleton pairs of X and Y is $O(a^2)$. If X is a singleton and Y is an artificial taxon, the number of v is $O(h)$, where h is the height of T, because v must be one of the ancestors of X. In this case, the number of the triplets is $O(abh)$. The same holds in the case where Y is a singleton and X is an artificial taxa. If both X and Y are artificial taxa, then the number of v is $O(n)$ since v could be any inner node in the tree, and hence the number of triplets is $O(b^2 n)$. To sum up, the total number of (v,X,Y), as well as the time complexity of Algorithm A.2, is $O(a^2 + abh + b^2 n) = O(s^2 n)$. In the last step, we compute the weights of the **good edges** $\mathbb{G}$. We can update each $\mathbb{G}$ in $O(s^2)$ time using Algorithm A.3. Thus, the time complexity is $O(s^2)$.

To conclude, the total time complexity is $O(s^2 n)$, and for k gene trees it is $O(s^2 nk)$. □

Algorithm A.4: Max-Cut Heuristic

Data: a input graph G, iteration limits N and M
Result: an approximate maximum cut C_{max}

initialize C by a random assignment
$C_{max} \leftarrow C$
repeat N **times**
    initialize θ' to a random perturbation of the angular representation of C
    starting from θ', minimize $f(\theta)$ by gradient descent with at most M iterations
    sort the elements of θ keeping track of how indices map to vertices in G
    identify the cut C associated with θ and angle 0 (and compute its weight)
    if $C > C_{max}$ **then** $C_{max} \leftarrow C$
    $\gamma \leftarrow 0,\ i \leftarrow 1,\ j \leftarrow n+1,\ \theta_{n+1} \leftarrow 2\pi$
    if $\exists\ j\ s.t.\ \theta_j > \pi$ **then**
        $j \leftarrow argmin_j\ \theta_j > \pi$
    while $\gamma < \pi$ **do**
        if $\theta_i \leq \theta_j - \pi$ **then**
            $i \leftarrow i+1,\ \gamma \leftarrow \theta_i$
            update and evaluate C by moving i to the other side
        else
            $j \leftarrow j+1,\ \gamma \leftarrow \theta_j - \pi$
            update and evaluate C by moving j to the other side
        if $C > C_{max}$ **then** $C_{max} \leftarrow C$
return C_{max}

Lemma A.3. *The time complexity of computing an approximate max-cut with Algorithm A.4 is $O(s^2)$, where s is the number of vertices in a complete graph.*

Proof. Algorithm A.4 computes an approximate max-cut using relaxation

$$\min_{\theta \in \mathbb{R}^s} f(\theta) = \min_{\theta \in \mathbb{R}^s} \frac{1}{2} G \cdot \cos T(\theta)$$

where G is an $s \times s$ adjacency matrix and θ is an s-vector (note that each element of θ is an angle on the interval $[0, 2\pi)$). The idea is that each entry of θ maps a vertex in G to the unit circle so a cut C is given by an angle in $[0, \pi)$. To evaluate f, we build an $s \times s$ matrix $[\cos T(\theta)]_{i,j} = \cos(\theta_i - \theta_j)$ and then compute the element-wise dot product: $A \cdot B = \sum_{i,j} a_{i,j} \cdot b_{i,j}$. Thus, the time complexity to evaluate f is $O(s^2)$. Algorithm A.4 (line 5) minimizes f using gradient descent; this has a time complexity of $O(s^2)$ because ∇f has a closed form solution, which also can be evaluated in $O(s^2)$ time, and only a fixed number M of iterations is performed ($M = 200$ in our implementation by 26). After gradient descent, Algorithm A.4 (line 6) identifies the cut C associated with θ and angle 0 and computes its weight. The former can be done in $O(s)$ time because the cut C is given by dividing the angles of θ into two groups: $[0, \pi)$ and $[\pi, 2\pi)$. Its weight can be computed in $O(s^2)$ time.

Lastly, Algorithm A.4 (lines 8–18) seeks to improve upon the cut C by moving the angle so that one vertex switches to the opposite side and then traversing around the circle. This is achieved by sorting the elements of θ, which if can done with a reasonable method takes $O(s \log s)$ time. Then, we can consider $O(s)$ possible cuts from moving the angle so that one vertex switch sides. The weight of the new cut can be computed in $O(s)$ time because this is the number of edges affected by moving a single vertex to the other side of the cut. Thus, the best cut C according to θ can be found in $O(s^2)$ time.

This entire process has time complexity $O(s^2)$ and is repeated for a fixed number N of times ($N = 10$ in our implementation by 26). This concludes our proof. For more details, please refer

to [16]. □

Theorem A.2. *Let $\mathbb{B}$ and $\mathbb{G}$ be the $s \times s$ matrices of bad and good edges in the quartet graph, respectively. Then, our approach for seeking a max-cut of the quartet graph has time complexity $O(s^2)$.*

Proof. To seek a max-cut of the quartet graph, we use a technique similar to the one introduced by [124]. They propose to seek a cut C to maximize

$$\sum_{(X,Y)\in C} (\mathbb{G}[X,Y] - \alpha\mathbb{B}[X,Y]) \tag{A.14}$$

where $\alpha > 0$ is a hyperparameter. The idea is to perform binary search to maximize the value of $\alpha > 0$ such that the maximum cut of the graph is non-empty. In TREE-QMC version 1.0.0, we search for α on the interval $[0,6]$ for a fixed level of precision (0.1), so the search terminates after $\log_2((6-0)/0.1)$ steps. At each step, we use an efficient heuristic for max-cut [16] that takes $O(s^2)$ time (Lemma A.3). Because the binary search iterates a constant number of times, we conclude that our approach for seeking a max-cut of the quartet graph has time complexity $O(s^2)$.

Our approach deviates from that of [124] in two ways. First, they apply a different heuristic for max-cut. Second, they perform the binary search for α over a larger interval. The latter is motivated by their theoretical result, showing α is at most $4|Q|$, where $|Q|$ is the size of the input quartet set, if the max-cut heuristic always returns the exact maximum cut. In the context of TREE-QMC, we build the quartet graph from k gene trees on n species, so $|Q| = O(n^4k)$. If we revised our approach to search for α on the associated interval for a fixed level of precision, it would terminate after $\log n + \log k$ steps. This revision would increase the time complexity of our approach for seeking a max-cut of the quartet graph to $O(s^2 \log n + s^2 \log k)$. However, it would not change the time complexity of TREE-QMC because the dominant cost for each subproblem would still be the time to compute the quartet graph from the gene trees (see Theorem A.1). These algorithmic differences between wQMC and TREE-QMC could be explored as part of future work. □

Theorem A.3. *Let n be the number of species, and let k be the number of gene trees. TREE-QMC has time complexity $O(n^3k)$ if subproblems are produced in a perfectly balanced fashion.*

Proof. At each step in the divide phase of the algorithm, we compute a bipartition and then recurse on two subproblems. Each subproblem is defined by taking the taxa on one side of the bipartition and adding an artificial taxon to represent the taxa on the other side of the bipartition. Thus, a subproblem of size 2^p+2 is divided into two subproblems of size $\frac{2^p+2}{2}+1 = 2^{p-1}+2$ if the division is balanced. The work f per subproblem has time complexity $O(s^2nk+s^2) = O(s^2nk)$ where the first term comes from constructing the quartet graph by applying our algorithm to all gene trees (Theorem A.1) and the second term comes from applying our approach for seeking a max-cut (Theorem A.2). Therefore,

$$\begin{aligned}
T(2^p+2) &= \sum_{i=0}^{p} 2^i \cdot f(2^{p-i}+2) \quad \text{where } f \text{ is } O(s^2nk) \\
&= \sum_{i=0}^{p} 2^i \cdot (2^{p-i}+2)^2 \cdot (2^p+2) \cdot k \\
&= (2^p+2) \cdot k \cdot \sum_{i=0}^{p} 2^i \cdot (2^{p-i}+2)^2 \\
&= (2^p+2) \cdot k \cdot \sum_{i=0}^{p} (2^{2(p-i)+i} + 2^{p-i+2+i} + 2^{2+i}) \\
&= (2^p+2) \cdot k \cdot (2^p \cdot (2^{p+1}-1) + 2^{p+2} \cdot (p+1) + 2^{p+3} - 4) \\
&= O(2^p \cdot k \cdot 2^{2p}) = O(2^{3p} \cdot k) = O(n^3k)
\end{aligned}$$

□

A.6.6 Correctness

Lemma A.4. *Let v be an internal node in T, let X be a taxon below v.left, and let Y, Z, and W be three taxa below v.right. The total weight of quartets $\{X,Y,Z,W\}$ in which X and Y are siblings,*

denoted by $\mathbb{R}_v[X,Y]$*, is*

$$\begin{cases} c_X[v.left] \cdot f_{Y,X}[v.right] & \text{if both } X \text{ and } Y \text{ are artificial} \\ f_{Y,0}[v.right] & \text{if only } Y \text{ is artificial} \\ c_x[v.left] \cdot (p_X[y] - p_x[v.right]) & \text{if only } X \text{ is artificial} \\ p_0[y] - p_0[v.right] & \text{if both } X \text{ and } Y \text{ are singletons} \end{cases} \tag{A.15}$$

where the values of $p_0[v]$*,* $p_X[v]$*,* $f_{Y,0}[v]$*, and* $f_{Y,X}[v]$ *are given by Algorithm A.1. Similarly, when* Z *and* W *are from* $v.left$*, the total weight, denoted by* $\mathbb{L}_v[X,Y]$*, is*

$$\begin{cases} c_y[v.right] \cdot f_{X,Y}[v.left] & \text{if both } X \text{ and } Y \text{ are artificial} \\ f_{X,0}[v.left] & \text{if only } Y \text{ is artificial} \\ c_Y[v.right] \cdot (p_Y[x] - p_Y[v.left]) & \text{if only } X \text{ is artificial} \\ p_0[x] - p_0[v.left] & \text{if both } X \text{ and } Y \text{ are singletons} \end{cases} \tag{A.16}$$

Finally, when Z *and* W *are from v.above, the total weight, denoted by* $\mathbb{A}_v[X,Y]$*, is*

$$\begin{cases} c_X[v.left] \cdot c_Y[v.right] \cdot g_{X,Y}[v.above] & \text{if both } X \text{ and } Y \text{ are artificial} \\ c_Y[v.left] \cdot g_Y[v.above] & \text{if only } Y \text{ is artificial} \\ c_X[v.right] \cdot g_X[v.above] & \text{if only } X \text{ is artificial} \\ g_0[v.above] & \text{if both } X \text{ and } Y \text{ are singletons} \end{cases} \tag{A.17}$$

Proof. Note: capital letters are used to denote the label of a leaf. For example, leaf m would have label M in the current subproblem.

The proof for $\mathbb{A}_v[X,Y]$ is simple since there is only one subtree above v. We prove the case in

which both X and X are artificial,

$$\begin{aligned}
\mathbb{A}_v[X,Y] &= \sum_{\substack{m\in L(v.left):\\ M=X}} \sum_{\substack{n\in L(v.right)\\ N=Y}} \sum_{\substack{z,w\in L(v.above):\\ Z\neq Y\neq X\neq W}} I(m)I(n)I(z)I(w) \\
&= \sum_{\substack{m\in L(v.left):\\ M=X}} I(m)\cdot\left(\sum_{\substack{n\in L(v.right):\\ N=Y}} I(n)\cdot\left(\sum_{\substack{z,w\in L(v.above):\\ Z\neq W\neq X\neq Y}} I(z)I(w)\right)\right) \\
&= c_X[v.left]\cdot c_Y[v.right]\cdot g_{X,Y}[v.above],
\end{aligned}$$

These equations are for artificial taxa X,Y but we can allow X to be a singleton by setting $c_X[u] = I(x) = 1$ and decreasing $c_0[u]$ by $I(x) = 1$ when computing $g_{X,Y}[u]$ (and similarly to allow Y to be a singleton). The other three cases of $\mathbb{A}_v[X,Y]$ are similar. In the following, we focus on $\mathbb{R}_v[X,Y]$ because the proof for $\mathbb{L}_v[x,y]$ is the same due to the symmetry. Recall that X is a taxon below $v.left$, and Y, Z, and W be three taxa below $v.right$.

If both X and Y are singletons, the prefix-sum technique is applied:

$$\begin{aligned}
\mathbb{R}_v[X,Y] &= \sum_{\substack{z,w\in L(v.right):\\ Z\neq W\neq Y\\ q(x,y,z,w)=x,y|z,w}} I(x)I(y)I(z)I(w) \\
&= \sum_{\substack{z,w\in L(v.right):\\ Z\neq W\neq Y\\ q(x,y,z,w)=x,y|z,w}} I(z)I(w) \\
&= \sum_{u\in\mathscr{P}(v.right,y)} \sum_{\substack{z,w\in L(u)\\ Z\neq W}} I(z)I(w) \\
&= \sum_{u\in\mathscr{P}(v.right,y)} g_0[u] \\
&= \sum_{u\in\mathscr{P}(root,y)} g_0[u] - \sum_{u\in\mathscr{P}(root,v.right)} g_0[u] \\
&= p_0[y] - p_0[v.right],
\end{aligned}$$

where $\mathscr{P}(i,j)$ is the set of the subtrees adjacent to the path from leaves i (included) to j (excluded). Note that $g_0[u]$ is used to compute $\sum_{z,w\in\mathscr{L}(u),Z\neq W} I(z)I(w)$ according to Lemma 1 in the main text.

If X is a singleton while Y is an artificial taxon, there could be multiple leaves labeled by Y in $L(v.right)$. Consequently, Y must be excluded when Z and W are selected, and thus $g_Y[u]$ is used instead of $g_0[u]$:

$$\begin{aligned}
\mathbb{R}_v[X,Y] &= \sum_{\substack{n\in L(v.right):\\ N=Y}} \sum_{\substack{z,w\in L(v.right):\\ Z\neq W\neq Y\\ q(x,n,z,w)=x,n|z,w}} I(x)I(n)I(z)I(w) \\
&= \sum_{\substack{n\in L(v.right):\\ N=Y}} I(n)\cdot\left(\sum_{u\in\mathscr{P}(v.right,n)} \sum_{\substack{z,w\in L(u):\\ Z\neq W\neq Y}} I(z)I(w)\right) \\
&= \sum_{\substack{n\in L(v.right):\\ N=Y}} I(n)\cdot \sum_{u\in\mathscr{P}(v.right,n)} g_Y[u] \\
&= f_{Y,0}[v.right]
\end{aligned}$$

Algorithm A.1 computes $f_{Y,0}[v]$ according to the following recurrence:

$$f_{Y,0}[v] = \begin{cases} 0 & \text{if } v \text{ is a leaf} \\ f_{Y,0}[v.left] + f_{Y,0}[v.right] & \\ +c_Y[v.left]\cdot g_Y[v.right] + c_Y[v.right]\cdot g_Y[v.left] & \text{if } v \text{ is an inner node} \end{cases}.$$

The correctness of the recurrence is shown as follows.

$$
\begin{aligned}
f_{Y,0}[v] &= \sum_{\substack{n\in L(v):\\ N=Y}} I(n)\cdot \sum_{u\in\mathscr{P}(v,n)} g_Y[u] \\
&= \sum_{\substack{n\in L(v.left):\\ N=Y}} I(n) \sum_{u\in\mathscr{P}(v,n)} g_y[u] + \sum_{\substack{n\in L(v.right):\\ N=Y}} I(n) \sum_{u\in\mathscr{P}(v,n)} g_Y[u] \\
&= \sum_{\substack{n\in L(v.left):\\ N=Y}} I(n)\cdot \left(g_Y[v.right] + \sum_{u\in\mathscr{P}(v.left,n)} g_y[u] \right) \\
&\quad + \sum_{\substack{n\in L(v.right):\\ N=Y}} I(n)\cdot \left(g_Y[v.left] + \sum_{u\in\mathscr{P}(v.right,n)} g_Y[u] \right) \\
&= \sum_{\substack{n\in L(v.left):\\ N=Y}} I(n)\cdot \sum_{u\in\mathscr{P}(v.left,n)} g_Y[u] + \sum_{\substack{n\in L(v.right):\\ N=Y}} I(n)\cdot \sum_{u\in\mathscr{P}(v.right,n)} g_Y[u] \\
&\quad + g_Y[v.right]\cdot \sum_{\substack{n\in L(v.left):\\ N=Y}} I(n) + g_Y[v.left]\cdot \sum_{\substack{n\in L(v.right):\\ N=Y}} I(n) \\
&= f_{Y,0}[v.left] + f_{Y,0}[v.right] + c_Y[v.left]\cdot g_Y[v.right] + c_Y[v.right]\cdot g_y[v.left].
\end{aligned}
$$

If X is an artificial taxon and Y is a singleton, X could also appear in $L(v.right)$ so we must be excluded to select Z and W, and therefore $g_X[u]$ is used to compute the sum of $I(z)I(w)$ in a

subtree.

$$
\begin{aligned}
\mathbb{R}_v[x,y] &= \sum_{\substack{m\in L(v.left)\\ M=X}} \sum_{\substack{z,w\in L(v.right):\\ Z\neq W\neq X\\ q(m,y,z,w)=m,y|z,w}} I(m)I(y)I(z)I(w) \\
&= \sum_{\substack{m\in L(v.left):\\ M=X}} I(m) \sum_{u\in\mathscr{P}(v.right,y)} \sum_{\substack{z,w\in L(u):\\ Z\neq W\neq X}} I(z)I(w) \\
&= \sum_{\substack{m\in L(v.left):\\ M=X}} I(m)\cdot\left(\sum_{u\in\mathscr{P}(v.right,y)} \sum_{\substack{z,w\in L(u):\\ Z\neq W\neq X}} I(z)I(w)\right) \\
&= \sum_{\substack{m\in L(v.left):\\ M=X}} I(m)\cdot\left(\sum_{u\in\mathscr{P}(v.right,y)} g_x[u]\right) \\
&= \sum_{\substack{m\in L(v.left):\\ M=X}} I(m)\cdot\left(\sum_{u\in\mathscr{P}(root,y)} g_X[u] - \sum_{u\in\mathscr{P}(root,v.right)} g_X[u]\right) \\
&= \sum_{\substack{m\in L(v.left):\\ M=X}} I(m)\cdot(p_X[y]-p_X[v.right]) \\
&= c_X[v.left]\cdot(p_X[y]-p_X[v.right]).
\end{aligned}
$$

Finally, if both X and Y are artificial taxa, $g_{X,Y}[u]$ is used.

$$\begin{aligned}
\mathbb{R}_v[X,Y] &= \sum_{\substack{m\in L(v.left):\\ M=X}} \sum_{\substack{N\in L(v.right):\\ n=Y}} \sum_{\substack{z,w\in L(v.right):\\ Z\neq W\neq X\neq Y\\ q(m,n,z,w)=m,n|z,w}} I(m)I(n)I(z)I(w) \\
&= \sum_{\substack{m\in L(v.left):\\ M=X}} I(m)\cdot\left(\sum_{\substack{n\in L(v.right):\\ N=Y}} I(n)\cdot\left(\sum_{u\in\mathscr{P}(v.right,n)} \sum_{\substack{z,w\in\mathscr{L}(u):\\ Z\neq W\neq X\neq Y}} I(z)I(w)\right)\right) \\
&= \sum_{\substack{m\in L(v.left):\\ M=X}} I(m)\cdot\left(\sum_{\substack{n\in L(v.right):\\ N=Y}} I(n)\cdot\left(\sum_{u\in\mathscr{P}(v.right,n)} g_{X,Y}[u]\right)\right) \\
&= \sum_{\substack{m\in L(v.left):\\ M=X}} I(m)\cdot f_{Y,X}[v.left] \\
&= c_X[v.left]\cdot f_{Y,X}[v.left]
\end{aligned}$$

Algorithm A.1 computes $f_{Y,X}[v]$ according to the following recurrence:

$$f_{Y,X}[v] = \begin{cases} 0 & \text{if } v \text{ is a leaf} \\ f_{Y,X}[v.left] + f_{Y,X}[v.right] & \\ +c_Y[v.left]\cdot g_{X,Y}[v.right] + c_Y[v.right]\cdot g_{X,Y}[v.left] & \text{if } v \text{ is an inner node} \end{cases}$$

whose proof of correctness is similar to $f_{Y,0}[v]$, except that $g_{X,Y}[u]$ is used instead of $g_X[u]$. □

Lemma A.5. *Let T be a binary tree, and let X and Y be taxa ($X \neq Y$). Then,*

$$\mathbb{B}[X,Y] = \sum_{v\in V(T)} \Big(\mathbb{A}_v[X,Y] + \mathbb{L}_v[X,Y] + \mathbb{R}_v[X,Y]\Big) + \Big(\mathbb{A}_v[Y,X] + \mathbb{L}_v[Y,X] + \mathbb{R}_v[Y,X]\Big)$$

where $V(T)$ is the set of internal nodes in gene tree T.

Proof. By definition, $\mathbb{B}[X,Y]$ is the total weight of the quartet in T whose topology is $X,Y|Z,W$,

where Z and W are two other taxa, so we have

$$
\begin{aligned}
\mathbb{B}[X,Y] &= \sum_{\substack{m\in L(T):\\ M=X}} \sum_{\substack{n\in L(T):\\ N=Y}} \sum_{\substack{z,w\in L(T):\\ Z\neq W\neq X\neq Y\\ q(n,m,z,w)=m,n|z,w}} I(m,n,z,w) \\
&= \sum_{v\in V(T)} \sum_{\substack{m\in L(v.left):\\ M=X}} \sum_{\substack{n\in L:(v.right)\\ N=Y}} \sum_{\substack{z,w\in L(T):\\ Z\neq W\neq X\neq Y\\ q(m,n,z,w)=m,n|z,w}} I(m,n,z,w) \\
&\quad + \sum_{v\in V(T)} \sum_{\substack{m\in L(v.right):\\ M=X}} \sum_{\substack{n\in L:(v.left)\\ N=Y}} \sum_{\substack{z,w\in \mathscr{L}(T):\\ Z\neq W\neq X\neq Y\\ q(m,n,z,w)=m,n|z,w}} I(m,n,z,w) \\
&= \sum_{v\in V(T)} \sum_{\substack{m\in L(v.left):\\ M=X}} \sum_{\substack{n\in L(v.right):\\ N=Y}} \Bigg(\sum_{\substack{z,w\in L(v.above):\\ Z\neq W\neq X\neq Y\\ q(n,m,z,w)=m,n|z,w}} I(m,n,z,w) \\
&\quad + \sum_{\substack{z,w\in L(v.left):\\ Z\neq W\neq X\neq Y\\ q(m,n,z,w)=m,n|z,w}} I(m,n,z,w) + \sum_{\substack{z,w\in L(v.right):\\ Z\neq W\neq X\neq Y\\ q(m,n,z,w)=m,n|z,w}} I(m,n,z,w) \Bigg) \\
&\quad + \sum_{v\in V(T)} \sum_{\substack{m\in L(v.right):\\ M=X}} \sum_{\substack{n\in L(v.left):\\ N=Y}} \Bigg(\sum_{\substack{z,w\in L(v.above):\\ Z\neq W\neq X\neq Y\\ q(m,n,z,w)=m,n|z,w}} I(m,n,z,w) \\
&\quad + \sum_{\substack{z,w\in L(v.left):\\ Z\neq W\neq X\neq Y\\ q(m,n,z,w)=m,n|z,w}} I(m,n,z,w) + \sum_{\substack{z,w\in L(v.right):\\ Z\neq W\neq X\neq Y\\ q(m,n,z,w)=m,n|z,w}} I(m,n,z,w) \Bigg)
\end{aligned}
$$

According to Lemma A.4, this is

$$
\sum_{v\in V(T)} \Big(\mathbb{A}_v[X,Y] + \mathbb{L}_v[X,Y] + \mathbb{R}_v[X,Y] \Big) + \Big(\mathbb{A}_v[Y,X] + \mathbb{L}_v[Y,X] + \mathbb{R}_v[Y,X] \Big)
$$

□

The following theorem follows from the Lemma above.

Theorem A.4. *Let T be a binary tree, and let X,Y be taxa ($X \neq Y$). Algorithm A.2 computes $\mathbb{B}[X,Y]$ correctly for T.*

Lemma A.6. *Let X and Y be taxa. Then,*

$$\mathbb{G}[X,Y] + \mathbb{B}[X,Y] = g_{X,Y}[r] \cdot c_X[r] \cdot c_Y[r] \tag{A.18}$$

where r is the root of gene tree T. These equations are for artificial taxa; however, they can be modified to accommodate singletons as follows (see Lemma A.2 for details). If X is a singleton, set $c_X[r] = 1$ in the equation above. Then, compute $g_{X,Y}[r]$ by setting $c_X[r]$ to 0 and subtracting 1 from $c_0[r]$. If Y is a singleton, set $c_Y[r] = 1$ in the equation above. Then, compute $g_{X,Y}[r]$ by setting $c_Y[r]$ to 0 and subtracting 1 from $c_0[r]$.

Proof.

$$\begin{aligned}
\mathbb{G}[X,Y] + \mathbb{B}[X,Y] &= \sum_{\substack{m,n,z,w \in L(r): \\ M=X, N=Y, \\ Z \neq W \neq X \neq Y}} I(m,n,z,w) \\
&= \sum_{\substack{m \in L(T): \\ M=X}} I(m) \cdot \left(\sum_{\substack{n \in L(T): \\ N=Y}} I(n) \cdot \left(\sum_{\substack{z,w \in L(T): \\ Z \neq W \neq X \neq Y}} I(z) I(w) \right) \right) \\
&= c_X[T] \cdot c_Y[T] \cdot g_{X,Y}[T].
\end{aligned}$$

Note: capital letters are used to denote the label of a leaf. For example, leaf m would have label M in the current subproblem. □

The theorem below follows from the Theorem and Lemma above.

Theorem A.5. *Let T be a binary tree, and let X,Y be taxa ($X \neq Y$). Algorithm A.3 computes $\mathbb{G}[X,Y]$ correctly for T.*

Appendix B: Supplementary Materials for Chapter 3

B.1 Quartet Weight Normalization

B.1.1 Preliminaries

A quartet is an unrooted, binary tree on four leaves $\{w,x,y,z\}$; thus it takes on one of three possible topologies, denoted $x,y|z,w$ or $x,z|y,w$ or $x,w|z,y$. Let L denote the mapping function from leaf vertices to taxon labels for the subproblem. To avoid confusion, we use *lower case letters to indicate leaf vertices* and *upper case letters to indicate leaf labels, also referred to as taxa or species*. This distinction is necessary because the introduction of artificial taxa results in multiple leaves being labeled by the same label (Fig. 3.1a–c).

B.1.2 Normalization, Importance Values, and Forest Data Structure

A key observation in the original TREE-QMC method is that gene trees can contribute more than one quartet per subset of four species when they are multi-labeled [45]. Thus, we wish to construct the normalized quartet graph, with quartet weights normalized so that each gene tree votes once for each subset of four labels present in the tree. To implement normalization, TREE-QMC assigns an **importance value** $\mathtt{I}(x)$ to each leaf vertex x and then **normalization factor** for each quartet $q = x,y|z,w$ by multiplying the importance values of its leaves:

$$\mathtt{I}(q) = \mathtt{I}(x,y,z,w) = \mathtt{I}(x)\mathtt{I}(y)\mathtt{I}(z)\mathtt{I}(w). \tag{B.1}$$

Later, we will use similar notation to indicate importance values for two or three leaves being multiplied together (e.g., $\mathtt{I}(x,y) = \mathtt{I}(x)\mathtt{I}(y)$ and $\mathtt{I}(x,y,z) = \mathtt{I}(x)\mathtt{I}(y)\mathtt{I}(z)$). It is easy to show that if

$$\sum_{\substack{a \in \mathtt{LeafSet}(T): \\ L(x)=X}} \mathtt{I}(x) = 1 \tag{B.2}$$

for every taxon label X, then the importance values form a valid normalization scheme, specifically

$$\sum_{\substack{a,b,c,d \in \mathtt{LeafSet}(T): \\ \{L(x),L(y),L(z),L(w)\}=\{X,Y,Z,W\}}} \mathtt{I}(x,y,z,w) = 1 \tag{B.3}$$

for selections of four unique taxon labels $\{X,Y,Z,W\}$ from the subproblem. Note that in the **unnormalized** case (**n0**), the importance values are set to one for all leaves, which does not satisfy Equation B.2 beyond the first subproblem in which all leaves are uniquely labeled.

TREE-QMC implements two normalization schemes: **uniform** (**n1**) and **non-uniform** (**n2**). The latter requires storing the hierarchical relationships between artificial taxa and real taxa for the subproblem as a collection of disjoint trees $\mathscr{F} = \{F_1, F_2, \ldots, F_x\}$, which we refer to as a **forest data structure**. The roots of trees in $\mathscr{F}$ are taxa in the current subproblem, whose importance values sum to one following Equation B.2. Leaf vertices of the trees in $\mathscr{F}$ are singleton taxa for the subproblem, which appear at most once in any gene tree; all other vertices represent artificial taxa for the subproblem.

To construct the quartet graph for the subproblem, the leaves of each gene tree are labeled by the roots of trees in $\mathscr{F}$ based on ancestor-descendant relationships (Fig. 3.1a–c). Importance values can be computed using $\mathscr{F}$ for both normalization schemes. In uniform normalization (n1), all leaves in the same tree $F_i \in \mathscr{F}$ are assigned the same importance value equal to the inverse of the number of leaves in F_i. In non-uniform normalization (n2), the importance values are computed

via the recurrence:

$$\mathtt{I}(x) = \begin{cases} 1 & \text{if } x \text{ is a root of some tree } F_i \in \mathscr{F} \\ \mathtt{I}(x.parent) \cdot (1/x.parent.outdegree) & \text{otherwise} \end{cases}$$

where $x.parent.outdegree$ returns the number of outgoing edges for the parent of x in F_i (Fig. 3.1c). The recurrence can be implemented via a preorder traversal of each tree in $\mathscr{F}$.

B.1.3 Normalization for Incomplete Gene Trees

If there are no missing taxa, the same importance values can be used for all gene trees (Fig. 3.1a–d). However, if a gene tree is incomplete, the importance values for subsets of four taxa may not sum to one, violating the requirement of normalization (Fig. 3.1d–e). To normalize correctly, importance values need to be computed for each gene tree based on its missing taxa (Fig. 3.1f). Early versions of the TREE-QMC code addressed the issue of incomplete gene trees by copying the forest data structure for each gene tree and then deleting the missing taxa while updating the importance values accordingly. This naive approach suffers due to frequent memory allocation and thus cache misses.

To improve performance, the current version of the TREE-QMC computes importance values from the forest data structure for the subproblem for each gene tree on the fly in two steps (see Algorithm 1 in the Supplementary Materials). First, the outdegree of internal nodes in the forest is updated to account for missing data via postorder traversal. Second, the importance values are updated according to the recurrence via preorder traversal. The time complexity to get the normalization values per gene tree is thus $O(n)$, which is lower than constructing the quartet graph from the gene tree. To implement normalization, we maintain the forest data structure for each subproblem on the stack during the divide phase of TREE-QMC. As the storage complexity of each forest is $O(n)$ and the maximum depth of recursion is $O(n)$, the total storage complexity for

the forest data structure is $O(n^2)$, which is the same as the quartet graph on the complete species set. Thus, our normalization approach does not increase the time or storage complexity of TREE-QMC.

Algorithm B.1: Computing Importance Values for Normalization on the Fly

input : Forest data structure F that stores importance values for a subproblem and gene tree g, which may be complete or incomplete
output: Forest data structure with updated importance values based on the taxa in gene tree g

```
1  Q ← all leaves l in F such that l is also in g;
2  while Q ≠ ∅ do
3      v ← Q.pop()
4      F[v].importance ← 1
5      if v.parent in F is not visited then
6          Q.push(F[v].parent)
7          F[v].parent.outdegree ← 0
8      F[v].parent.outdegree ← F[v].parent.outdegree + 1
9  Q ← all roots r in F with r.importance = 1
10 while Q ≠ ∅ do
11     v ← Q.pop()
12     for each child u of v do
13         Q.push(u)
14         F[u].importance ← F[u].importance · F[v].outdegree^-1
15 return F
```

B.2 Overview of Weighted Quartet Graph Construction

We now describe how to construct the (normalized) quartet graph using the weighting schemes of [156] without explicitly extracting all weighted quartets from the gene trees (Fig. B.1).

Strategy for Computing Weighted Bad Edges and Good for Subproblem

Recall that vertices in the quartet graph represent taxa from the subproblem and edges between taxa are weighted by quartets implied by the gene trees, whose leaves map to taxa. If the

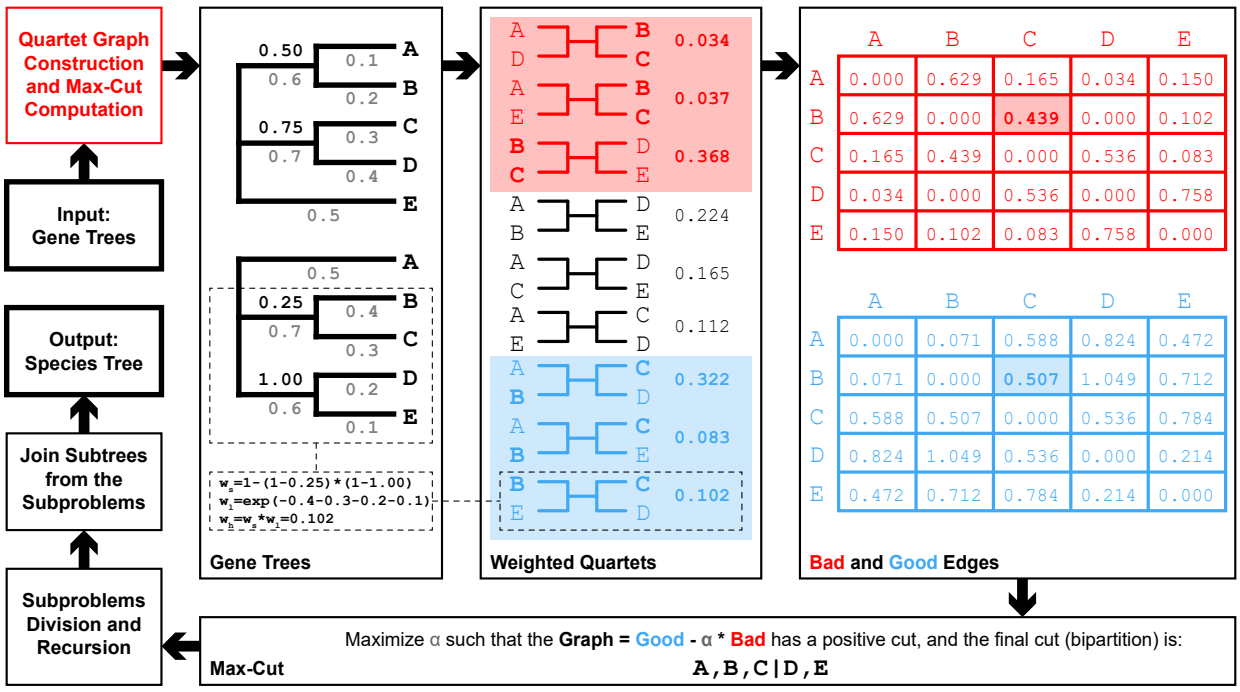


Figure B.1: **Weighted Quartet Graph Construction.** The goal of weighted TREE-QMC is to take into account gene tree branch lengths and support values when constructing the quartet graph, whose (max) cut determines a bipartition of taxa in the output unrooted species tree. Consider the two input gene trees on 5 taxa with branches are labeled by branch lengths (below) and branch support values (above). The naive approach to construct quartet graph first extracts all weighted quartets from the gene trees. For example, the quartet $q = B,C|D,E$ only exists in one of the gene trees. Its two internal branches have support values of 0.25 and 1.00 and its four terminal branches have lengths of 0.4, 0.3, 0.2, and 0.1, respectively. As a result, its length weight is $\mathtt{W}_\mathtt{l}(q) = \exp(-0.4-0.3-0.2-0.1) = 0.102$, its support weight is $\mathtt{W}_\mathtt{s}(q) = 1-(1-0.25)\cdot(1-1) = 1$, and its hybrid weight is $\mathtt{W}_\mathtt{h}(q) = \mathtt{W}_\mathtt{s}(q)\cdot\mathtt{W}_\mathtt{l}(q) = 0.102$, following Equations 1, 2, and 3 in the main text, respectively. To compute the good and bad edge hybrid weights between the taxa B and C, we consider all quartets containing them. There are three quartets in which B and C are siblings—the total hybrid weight of the bad edges between B and C, denoted $\mathbb{B}(B,C)$, is $0.034+0.039+0.368 = 0.439$. Similarly, we consider all quartets involving B and C in which B and C are not siblings—the total weight of the good edges between B and C, denoted $\mathbb{G}(B,C)$, is $0.322+0.083+0.102 = 0.507$. Finally, we compute the optimal cut of the quartet graph and obtain the bipartition $A,B,C|D,E$, which is possible by brute in this example because the number of taxa is small. The TREE-QMC method continues by recursion on the subproblems defined by the two halves of the bipartition: A,B,C and D,E, after the introduction of artificial taxa $\mathscr{A}_1$ and $\mathscr{A}_2$ to represent taxa on the other side of the bipartition $\{D,E\}$ and $\{A,B,C\}$, respectively. To solve the first subproblem, we would construct quartet graph relabeling the gene trees with taxa $\{A,B,C,\mathscr{A}_1\}$ and seek its max cut. The termination criterion is already satisfied for the second subproblem on $\{D,E,\mathscr{A}_2\}$ because only unrooted phylogenetic tree is possible on three leaves.

leaves of some quartet $q = x,y|z,w$ implied by a gene tree T have distinct labels, then q contributes two **bad** edges $(L(x),L(y))$ and $(L(z),L(w))$ as well as four **good** edges $(L(x),L(z))$, $(L(x),L(w))$, $(L(y),L(z))$, and $(L(y),L(w))$ to the quartet graph, provided that $L(x) \neq L(y) \neq L(z) \neq L(w)$; otherwise q does not contribute any good or bad edges. Good and bad edges inherit the weight of the quartet that contributed them, with the weight set to one in the unweighted algorithm and set using Equations 3.1, 3.2, or 3.3 for the length, support, or hybrid weighting schemes, respectively. Observe that under the hybrid weighting scheme, branch support weighting can be turned off by setting all support values to one and similarly length weighting can be turned off by setting all branch lengths to zero. Moreover, **non-binary gene trees** can be handled by simply refining polytomies and setting the resulting edges to have support zero and length zero (all other edges have support one and length zero in the unweighted algorithm or inherit their respective values from the unrefined tree in the weighted algorithm). As non-binary trees as well as length and support weights can be handled via the hybrid weighting scheme, we focus on hybrid weights henceforth.

For ease of computation, we split the calculation of the hybrid weight into two parts:

$$\begin{aligned}
\mathtt{W_h}(q) &= \mathtt{W_l}(q) \cdot \mathtt{W_s}(q) \\
&= \mathtt{W_l}(q) \cdot \left(1 - \prod_{e \in u \to v} (1 - \mathtt{s}(e))\right) \\
&= \mathtt{W_l}(q) - \mathtt{W_l}(q) \cdot \prod_{e \in u \to v} (1 - \mathtt{s}(e)) \\
&= \mathtt{W_l}(q) - \Delta\mathtt{W}(q)
\end{aligned} \tag{B.4}$$

where u and v denote the anchor vertices for quartet q in gene tree T (Fig. 3.2). Then, the total

weight of bad edges between X and Y in the quartet graph computed from gene tree T becomes

$$\begin{aligned}\mathbb{B}(X,Y) &= \sum_{\substack{x,y,z,w\in \texttt{LeafSet}(T)\\ L(x)\neq L(y)\neq L(z)\neq L(w)\\ L(x)=X, L(y)=Y\\ q=L(x),L(y)|L(z),L(w)}} \texttt{W}_\texttt{h}(q)\texttt{I}(q) \\ &= \left(\sum \texttt{W}_\texttt{l}(q)\texttt{I}(q)\right) - \left(\sum \Delta\texttt{W}(q)\texttt{I}(q)\right) \\ &= \left(\sum \texttt{W}_\texttt{l}(q)\texttt{I}(q)\right) - \Delta\mathbb{B}(X,Y) \end{aligned} \tag{B.5}$$

where $\texttt{I}(q)$ corresponds to the importance value for normalizing the weight of each quartet q (Equation B.3) and the summations loop over all ways of selecting four unique leaves x, y, z, w from $\texttt{LeafSet}(T)$ such that the implied quartet has distinct labels and separates two leaves labeled by X and Y from the other two leaves. Observe that an algorithm for computing the left term $\Delta\mathbb{B}(X,Y)$ given gene tree T can be used to compute the right term after setting all branch support values in T to zero. Summing the left and right terms together for all pairs of taxa (X,Y) and repeating across all k gene trees yields the bad edges $\mathbb{B}$ in the weighted quartet graph for the subproblem. The good edges, denoted $\mathbb{G}$, can be computed in a similar fashion.

We therefore focus on algorithms for computing $\Delta\mathbb{B}(X,Y)$ and $\Delta\mathbb{G}(X,Y)$ from a single unrooted gene tree T henceforth. We assume that T is binary (because any non-binary tree can be transformed into a binary tree with support values of zero on its refined edges, as previously mentioned) with n leaves mapping to $s = a + b$ taxon labels for the subproblem, where a denotes the number of singleton taxa and b denotes the number of artificial taxa (Fig. 3.1). For ease of computation, we root T arbitrarily, letting h denote its height h (note that in our complexity analysis h and n are taken to be the maximum values across the k gene trees). We let $t.p$, $t.s$, $t.l$, and $t.r$ denote the parent, sibling, left child, and right child of vertex t, respectively. We let $\texttt{below}(t)$ denote the set of leaves "below" the subtree rooted at vertex t (i.e., they are descendants of vertex t). Conversely, we let $\texttt{above}(t)$ denote the set of leaves "above" or "outside" the subtree rooted at t (i.e., they are

in the set $\texttt{LeafSet}(\texttt{T}) \setminus \texttt{below}(t)$).

To provide some intuition for computing $\Delta\mathbb{B}(X,Y)$ and $\Delta\mathbb{G}(X,Y)$, consider a pair of leaves x and y in $\texttt{LeafSet}(T)$ such that $L(x) = X$ and $L(y) = Y$. Let $x \to y$ denote the (unique) path between x and y in T, and let $\mathscr{F}$ denote the set of subtrees obtained from deleting edges on the path from $x \to y$ as well as their endpoints from T. To form a bad or good quartet for leaf pair x,y, we must select two leaves z and w from $\mathscr{F}$ such that $L(z) \neq L(w) \neq L(x) \neq L(y)$ (Fig. 6A in 45). All pairs of z and w selected from the same subtree in $\mathscr{F}$ correspond to a "bad quartet for X,Y" because x,y are siblings (one anchor vertex is on the path from $x \to y$ and the other anchor vertex is within the subtree from which z,w were selected). Likewise, all pairs of z and w selected from different subtrees in $\mathscr{F}$ correspond to a "good quartet for X,Y" because x,y are NOT siblings (both anchor vertices of the quartet are on the path $x \to y$).

Building upon this idea, we propose to compute $\Delta\mathbb{B}(X,Y)$ and $\Delta\mathbb{G}(X,Y)$ by summing up the weights from bad and good quartets for X,Y at their anchor vertices in gene tree T. Specifically, we let $\Delta\mathbb{B}_t(X,Y)$ and $\Delta\mathbb{G}_t(X,Y)$ correspond to the weights from bad and good quartets for X,Y such that one leaf is below the left child of t (call x), one leaf is below the right child of t (call y), and one of x,y is labeled by X and the other is labeled by Y (w.l.o.g. X is the taxa labeling x and Y is the taxon labeling y; see Definitions 1 and 2). Because T can be multi-labeled (e.g., due to the introduction artificial taxa), we must consider all pairs x,y such that $x \in \{L(l) = X : l \in \texttt{below}(t.l)\}$ and $y \in \{L(l) = Y : l \in \texttt{below}(t.r)\}$. To complete the bad or good quartet, we must consider all ways of selecting two additional leaves z and w from T such that $L(z) \neq L(w) \neq L(x) \neq L(y)$. This allows us to break the computation of $\Delta\mathbb{B}_t(X,Y)$ and $\Delta\mathbb{G}_t(X,Y)$ into six cases: (a) $z,w \in \texttt{below}(t.l)$, (b) $z \in \texttt{below}(t.l)$ and $w \in \texttt{above}(t.p)$, (c) $z,w \in \texttt{above}(t.p)$, (d) $z \in \texttt{above}(t.p)$ and $w \in \texttt{below}(t.r)$ (symmetric with case (b)), (e) $z,w \in \texttt{below}(t.r)$ (symmetric with case (a)), and (f) $z \in \texttt{below}(t.l)$ and $w \in \texttt{below}(t.r)$. We define equations to compute the weights from each of these cases and then sum the relevant cases to get $\Delta\mathbb{B}_t(X,Y)$ and $\Delta\mathbb{G}_t(X,Y)$ (Theorems 1 and 2). For example, it is easy to see that case (c) is not relevant for the computation of $\Delta\mathbb{G}(X,Y)$

because z and w are drawn from the same subtree off the path from any x, y under consideration, thus forming a bad quartet. Our equations are given based on the precomputation of six auxiliary values, summarized below.

- $\mathtt{w}^1_X(t)$ and $\overline{\mathtt{w}}^1_X(t)$: **singlet auxiliary values** give the sum of length weights from leaves labeled by taxon X in the subtree below and above vertex t, respectively (Definition B.3 and B.4). Singlets are useful for computing the branch lengths of terminal edges to leaves labeled X in the quartets. They can be computed for all taxa X and for all vertices $t \in V(T)$ in $O((a+b)n)$ time and require $O((a+b)n)$ storage to access later (Lemmas B.1 and B.2).

- $\mathtt{w}^2_{X,Y}(t)$ and $\overline{\mathtt{w}}^2_{X,Y}(t)$: **doublet auxiliary values** give the sum of weights corresponding to pairs of leaves labeled by taxa X, Y in the subtree below and above vertex t (Definitions B.5 and B.6). Doublets are useful for accessing the complete branch lengths of terminal edges for leaf pairs labeled X, Y attached to the same in anchor in a quartet and computing the branch support for the internal edge. They can be computed for all pairs of taxa X, Y and for all vertices $t \in V(T)$ in $O((a+b)^2 n)$ time and require $O((a+b)^2 n)$ storage to access later (Lemmas B.3 and B.4).

- $\mathtt{w}^{x|z,w}_{X,Y}(t)$: **bad triplet auxiliary values** give the sum of weights from three leaves x, z, w in the subtree below t such that the triplet corresponds to a bad quartet for X, Y (Definition B.7). Bad triplets are useful for accessing the branch lengths for the three terminal edges and the branch support for the internal edge in a bad quartet. They can be computed for all pairs of taxa X, Y and all vertices $t \in V(T)$ in $O((a+b)^4 n)$ time and require $O((a+b)^2 n)$ storage to access later (Lemma B.5).

- $\mathtt{w}^{x,z|w}_{X,Y}(t)$: **good triplet auxiliary values** give the sum of weights from three leaves x, z, w in the subtree below t such that the triplet corresponds to a good quartet for X, Y (Definition B.8). Good triplets can be computed for all pairs of taxa X, Y and for all vertices $t \in V(T)$ in

$O((a+b)^4 n)$ time and require $O((a+b)^2 n)$ storage to access later (Lemma B.6).

After precomputation, $\Delta\mathbb{B}_t(X,Y)$ and $\Delta\mathbb{G}_t(X,Y)$ can be computed in constant time. Summing across all vertices in T enables us to build $\Delta\mathbb{B}(X,Y)$ and $\Delta\mathbb{G}(X,Y)$ for all X,Y in $O((a+b)^2 n)$ time and $O((a+b)^2)$ space (Theorem B.3). This does not exceed the requirements for precomputation, putting the final time and space complexity for computing $\Delta\mathbb{B}$ and $\Delta\mathbb{G}$ given gene tree T at $O((a+b)^4 n)$ and $O(a^2 n + abn + b^2 n)$, respectively.

Efficient Algorithm for Computing Weighted Bad Edges and Good for Subproblem

Our core contribution is reducing the final time and space complexity to $O(a^2 b + ab^2 h + b^3 n)$ and $O(a^2 + ab + b^2 n)$, respectively (Theorem B.4). We summarize our strategy below.

Reducing the time complexity of triplet auxiliary values by aggregating singletons. Our first goal is to reduce the time complexity of computing triplets. To achieve this goal, we introduce a new label 1 to represent all singleton taxa and then extend the computation of singlets and doublets to this new label in the natural way (Definitions B.9-B.11 and Corollaries B.1–B.3), effectively aggregating the computation across all singleton taxa. We then reorganize the computation of triplet recurrences to leverage the aggregated singlets and doublets, along with some other tricks. This drops the time complexity of the triplet recurrences from $O((a+b)^2)$ to $O(b)$ (Lemmas B.7–B.10 and Corollaries B.4–B.5 and B.7–B.8), enabling us to compute triplet auxiliary values for all vertices in T and for all taxa X,Y in $O((a+b)^2 bn)$ time (Corollaries B.6 and B.9) instead of $O((a+b)^4 n)$ time.

Reducing the storage and time complexity of auxiliary values for singleton taxa. Our second goal is to reduce the storage and time complexity of auxiliary values for singleton taxa. To achieve this goal, we break the computation of $\Delta\mathbb{B}(X,Y)$ and $\Delta\mathbb{G}(X,Y)$ into three cases (i)–(iii), depending

on whether zero, one, or both of X and Y are singleton taxa. In case (i), there are multiple leaves labeled X, each with its own path to the root of T, and similarly for the leaves labeled Y. This means there are multiple vertices (lowest common ancestors of X, Y) in gene tree T where we need to compute $\Delta\mathbb{B}_t(X,Y)$ and $\Delta\mathbb{G}_t(X,Y)$, with the number of vertices is $O(n)$. In case (ii), there are multiple leaves labeled Y, each with its own path to the root, but there is only one path from the unique leaf labeled X to the root. Again, there are multiple vertices in gene tree T where we need to compute $\Delta\mathbb{B}_t(X,Y)$ and $\Delta\mathbb{G}_t(X,Y)$, but the number of vertices is $O(h)$. Lastly, in case (iii), there is one path from the unique leaf x labeled X to the root and one path from the unique leaf y labeled Y to the root. So there is just one vertex in gene tree T where we need to compute $\Delta\mathbb{B}_t(X,Y)$ and $\Delta\mathbb{G}_t(X,Y)$. The same analysis holds for the auxiliary values.

We can exploit path uniqueness for singleton taxa to reduce the storage and time complexity of auxiliary values. Take the singlet auxiliary value $\mathtt{w}^1_X$ as an example. Fortunately, the singlet $\mathtt{w}^1_X$ is only evaluated at $t.l$ and $t.r$ when computing $\Delta\mathbb{B}_t(X,Y)$ and $\Delta\mathbb{G}_t(X,Y)$ (Theorems 1 and 2). If X is a singleton taxon, there are three possible cases at vertex t: (1), X labels a leaf below $t.l$ so $\mathtt{w}^1_X(t.l)$ exists and $\mathtt{w}^1_X(t.r)$ is undefined, (2) X labels a leaf below $t.r$ so $\mathtt{w}^1_X(t.l)$ is undefined and $\mathtt{w}^1_X(t.r)$ exists, and (3) X labels a leaf above t so both $\mathtt{w}^1_X(t.l)$ and $\mathtt{w}^1_X(t.r)$ are undefined. $\Delta\mathbb{B}_t(X,Y)$ and $\Delta\mathbb{G}_t(X,Y)$ are also undefined in case (3), so we only need to consider cases (1) and (2). In both cases, we can compute $\Delta\mathbb{B}_t(X,Y)$ and $\Delta\mathbb{G}_t(X,Y)$ in a postorder traversal, so that we do not need to store $\mathtt{w}^1_X(t.r)$ or $\mathtt{w}^1_X(t.l)$ after processing vertex t. Before continuing the traversal from t, we need to compute and store $\mathtt{w}^1_X(t)$ to access it at $t.p$. In case (1), we can compute $\mathtt{w}^1_X(t)$ from $\mathtt{w}^1_X(t.l)$, and in case (2) we can compute $\mathtt{w}^1_X(t)$ from $\mathtt{w}^1_X(t.r)$. Thus, if X is a singleton, we can simplify the recurrence for $\mathtt{w}^1_X$ (denoted $\mathtt{p}^1_X$), and we can overwrite the old value at $t.l$ or $t.r$ with the new value at t. This reduces the storage and time complexity of $\mathtt{w}^1_X(t)$ from $O(an+bn)$ to $O(a+bn)$ (Corollary B.10). Similar results can be given for the other auxiliary quantities (Corollaries B.11–B.14).

Ultimately, the auxiliary values with at least one singleton taxon can be computed on the fly during weighted graph construction and do not affect time or storage complexity (proof of Theorem

4). Thus, we only need to compute and store auxiliary values for artificial taxa during precomputation. This reduces the time complexity of precomputation from $O((a+b)^4 n)$ to $O(b^3 n)$ and reduces the storage complexity from $O((a+b)^2 n)$ to $O(b^2 n)$ (proof of Theorem B.4).

Wrapping up. The computation of $\Delta\mathbb{B}_t(X,Y)$ and $\Delta\mathbb{G}_t(X,Y)$ can be improved by applying the previously introduced tricks based on aggregating singleton taxa and splitting up the computation into three cases based on whether X and/or Y are singletons (Corollaries 15 and 16). After putting everything together, it takes $O((a^2b+ab^2h+b^3n)k)$ time and $O((a+b)^2+b^2n) = O(a^2+ab+b^2n)$ space to construct the weighted quartet graph for the subproblem given k gene trees, including precomputation (Theorem B.4).

B.3 Details of Weighted Quartet Graph Construction

We now provide the details of weighted graph construction.

Definition B.1 (bad edges below t)**.** *The total weight of bad edges between taxa X,Y below vertex t in gene tree T is defined as*

$$\Delta\mathbb{B}_t(X,Y) = \sum_{\substack{x,y,z,w \in \texttt{LeafSet}(T) \\ L(x)\neq L(y)\neq L(z)\neq L(w) \\ x\in\texttt{below}(t.l), y\in\texttt{below}(t.r) \\ L(x)=X, L(y)=Y \\ q=x,y|z,w}} \texttt{I}(q)\cdot\Delta\texttt{W}(q) \tag{B.6}$$

where $x,y,z,w \in \texttt{LeafSet}(T)$ enumerates all unordered *tuples of four leaves from T such that $L(x) \neq L(y) \neq L(z) \neq L(w)$, one leaf is below the left child of t (call x), one leaf below the right child of t (call y), one of x,y is labeled by X and the other is labeled by Y (w.l.o.g. X is the taxa labeling x and Y is the taxon labeling y), and X and Y are siblings in the resulting quartet.*

Definition B.2 (good edges below t)**.** *The total weight of good edges between taxa X,Y below*

vertex t in gene tree T is defined as

$$\Delta\mathbb{G}_t(X,Y) = \sum_{\substack{x,y,z,w \in \texttt{LeafSet}(T) \\ L(x)\neq L(y)\neq L(z)\neq L(w) \\ x\in\texttt{below}(t.l), y\in\texttt{below}(t.r) \\ L(x)=X, L(y)=Y \\ q=x,z|y,w}} \texttt{I}(q)\cdot\Delta\texttt{W}(q) \tag{B.7}$$

where $x,y,z,w \in \texttt{LeafSet}(T)$ enumerates all unordered *tuples of four unique leaves from T such that $L(x) \neq L(y) \neq L(z) \neq L(w)$, one leaf is below the left child of t (call x), one leaf below the right child of t (call y), one of x,y is labeled by X and the other is labeled by Y (w.l.o.g. X is the taxa labeling x and Y is the taxon labeling y), and X and Y are NOT siblings in the resulting quartet.*

The taxa in the subproblem are broken down into two subsets: the set $\mathscr{S}$ of singleton taxa and the set $\mathscr{A}$ of artificial taxa. For time and space complexity analyses, $a = |\mathscr{S}|$, $b = |\mathscr{A}|$, n is the number of leaves in gene tree T, and h is the height of T.

B.3.1 Auxiliary Values $\texttt{w}$

Before giving equations to compute $\Delta\mathbb{B}_t(X,Y)$ and $\Delta\mathbb{G}_t(X,Y)$ from T, we define six auxiliary values, described below.

Definition B.3 (singlet auxiliary values below t)**.** *Let $\texttt{w}^1_X(t)$ be the sum of weights from leaves labeled by X in the subtree* below *vertex t in gene tree T. More precisely, the singlet auxiliary values of X* below *t are defined as*

$$\texttt{w}^1_X(t) = \sum_{\substack{x\in\texttt{below}(t) \\ L(x)=X}} \texttt{I}(x)\cdot\texttt{exp}\Big(\sum_{e\in x\to t} -\texttt{l}(e)\Big) \tag{B.8}$$

where $\texttt{s}(e)$ and $\texttt{l}(e)$ denote the support value and branch length of an edge e in T.

Lemma B.1 (singlet auxiliary values below t)**.** $\mathtt{w}^1_X(t)$ *can be computed according to the following recurrence:*

$$\mathtt{w}^1_X(t) = \begin{cases} \mathtt{I}(t) & \text{if } t \text{ is a leaf in } T \\ \mathtt{exp}(-\mathtt{l}(t.l)) \cdot \mathtt{w}^1_X(t.l) + \mathtt{exp}(-\mathtt{l}(t.r)) \cdot \mathtt{w}^1_X(t.r) & \text{otherwise} \end{cases} \tag{B.9}$$

It takes $O((a+b)n)$ time to compute $\mathtt{w}^1_X(t)$ for all vertices $t \in V(T)$ and for all taxa $X \in \mathscr{S} \cup \mathscr{A}$. Likewise, it takes $O((a+b)n)$ space to access these singlet auxiliary values later. The time and storage complexity drops to $O(bn)$ when limiting the computation to artificial taxa (i.e., $X \in \mathscr{A}$).

Proof. In the base case, vertex t is a leaf, so $\mathtt{w}^1_X(t) = I(x)$ because there are no edges on the path from any leaf x to itself so we only need to consider the importance value at the leaf. Otherwise t is a non-leaf vertex, so we can split the weight into two parts: weight from leaves in the subtree below $t.l$ and weight from leaves in the subtree below $t.r$:

$$\begin{aligned}
\mathtt{w}^1_X(t) &= \sum_{\substack{x \in \mathtt{below}(t) \\ L(x)=X}} \mathtt{I}(x) \cdot \mathtt{exp}\Big(\sum_{e \in x \to t} -\mathtt{l}(e)\Big) \\
&= \sum_{\substack{x \in \mathtt{below}(t.l) \\ L(x)=X}} \mathtt{I}(x) \cdot \mathtt{exp}\Big(\sum_{e \in x \to t} -\mathtt{l}(e)\Big) + \sum_{\substack{x \in \mathtt{below}(t.r) \\ L(x)=X}} \mathtt{I}(x) \cdot \mathtt{exp}\Big(\sum_{e \in x \to t} -\mathtt{l}(e)\Big) \\
&= \mathtt{exp}(-\mathtt{l}(t.l)) \cdot \sum_{\substack{x \in \mathtt{below}(t.l) \\ L(x)=X}} \mathtt{I}(x) \cdot \mathtt{exp}\Big(\sum_{e \in x \to t.l} -\mathtt{l}(e)\Big) \\
&\quad + \mathtt{exp}(-\mathtt{l}(t.r)) \cdot \sum_{\substack{x \in \mathtt{below}(t.r) \\ L(x)=X}} \mathtt{I}(x) \cdot \mathtt{exp}\Big(\sum_{e \in x \to t.r} -\mathtt{l}(e)\Big) \\
&= \mathtt{exp}(-\mathtt{l}(t.l)) \cdot \mathtt{w}^1_X(t.l) + \mathtt{exp}(-\mathtt{l}(t.r)) \cdot \mathtt{w}^1_X(t.r).
\end{aligned}$$

This recurrence can be computed by traversing the $O(n)$ vertices of gene tree T in a postorder fashion. Evaluating the recurrence for space for vertex t and taxon X takes $O(1)$ time and storing

the resulting auxiliary value requires $O(1)$. Repeating for all $O(a+b)$ taxa $X \in \mathscr{S} \cup \mathscr{A}$ gives the time and storage complexity. □

Definition B.4 (singlet auxiliary values above t)**.** *Let $\overline{\mathtt{w}}^1_X(t)$ be the sum of weights from leaves labeled by X in the subtree* above *vertex t in gene tree T. More precisely, the singlet auxiliary values of X* above *t are defined as*

$$\overline{\mathtt{w}}^1_X(t) = \sum_{\substack{x \in \mathtt{above}(t) \\ L(x)=X}} \mathtt{I}(x) \cdot \mathtt{exp}\Big(\sum_{e \in x \to t} -\mathtt{l}(e) \Big), \tag{B.10}$$

where $\mathtt{s}(e)$ and $\mathtt{l}(e)$ denote the support value and branch length of an edge e in T.

Lemma B.2 (singlet auxiliary values above t)**.** *$\overline{\mathtt{w}}^1_X(t)$ can be computed according to the following recurrence*

$$\overline{\mathtt{w}}^1_X(t) = \begin{cases} 0 & \text{if } t \text{ is the root of } T \\ \mathtt{exp}(-\mathtt{l}(t)) \Big(\cdot\, \overline{\mathtt{w}}^1_X(t.p) + \mathtt{exp}(-\mathtt{l}(t.s)) \cdot \mathtt{w}^1_X(t.s) \Big) & \text{otherwise} \end{cases} \tag{B.11}$$

After computing $\mathtt{w}^1_X$ (Lemma B.1), the time and space complexity for $\overline{\mathtt{w}}^1_X$ is the same as for $\mathtt{w}^1_X$ (Lemma B.1).

Proof. In the base case, vertex t is a root, so $\mathtt{w}^1_X(t) = 0$ because there are no edges or importance values associated with the root. Otherwise t is a non-root vertex, so we can split the weight into two parts: weight from the leaves in the subtree above $t.p$ and weight from the leaves in the subtree

below $t.s$:

$$\begin{aligned}
\overline{\mathtt{w}}^1_X(t) &= \sum_{\substack{x \in \mathtt{above}(t) \\ L(x)=X}} \mathtt{I}(x) \cdot \mathtt{exp}\Big(\sum_{e \in x \to t} -\mathtt{l}(e)\Big) \\
&= \sum_{\substack{x \in \mathtt{above}(t.p) \\ L(x)=X}} \mathtt{I}(x) \cdot \mathtt{exp}\Big(\sum_{e \in x \to t} -\mathtt{l}(e)\Big) + \sum_{\substack{x \in \mathtt{below}(t.s) \\ L(x)=X}} \mathtt{I}(x) \cdot \mathtt{exp}\Big(\sum_{e \in x \to t} -\mathtt{l}(e)\Big) \\
&= \mathtt{exp}(-\mathtt{l}(t)) \cdot \Bigg(\sum_{\substack{x \in \mathtt{above}(t.p) \\ L(x)=X}} \mathtt{I}(x) \cdot \mathtt{exp}\Big(\sum_{e \in x \to t.p} -\mathtt{l}(e)\Big) \\
&\qquad + \mathtt{exp}(-\mathtt{l}(t.s)) \cdot \sum_{\substack{x \in \mathtt{below}(t.s) \\ L(x)=X}} \mathtt{I}(x) \cdot \mathtt{exp}\Big(\sum_{e \in x \to t.s} -\mathtt{l}(e)\Big) \Bigg) \\
&= \mathtt{exp}(-\mathtt{l}(t)) \cdot \Big(\overline{\mathtt{w}}^1_X(t.p) + \mathtt{exp}(-\mathtt{l}(t.s)) \cdot \mathtt{w}^1_X(t.s) \Big).
\end{aligned}$$

After precomputation $\overline{\mathtt{w}}^1_X(t)$ can be computed by traversing the $O(n)$ vertices of gene tree T in a preorder fashion. The remainder of the complexity analysis for $\overline{\mathtt{w}}^1_X(t)$ is the same as for $\mathtt{w}^1_X(t)$ (Lemma B.1). □

Definition B.5 (doublet auxiliary values below t)**.** *Let $\mathtt{w}^2_{X,Y}(t)$ be the sum of the weights from leaf pairs x and y in the subtree* below *vertex t in gene tree T such that one leaf (w.l.o.g. x) is labeled X, the other leaf (w.l.o.g. y) is labeled Y, and $X \neq Y$. More precisely, the doublet auxiliary values of X and Y* below *t are defined as*

$$\mathtt{w}^2_{X,Y}(t) = \mathtt{w}^2_{Y,X}(t) = \sum_{\substack{x,y \in \mathtt{below}(t) \\ L(x)=X, L(y)=Y}} \mathtt{I}(x,y) \cdot \mathtt{exp}\Big(\sum_{e \in x,y \to u} -\mathtt{l}(e)\Big) \cdot \prod_{e \in u \to t} (1 - \mathtt{s}(e)) \tag{B.12}$$

(B.13)

where u is the lowest common ancestor of x, y, denoted $lca(x,y)$.

Lemma B.3 (doublet auxiliary values below t)**.** *$\mathtt{w}^2_{X,Y}(t)$ can be computed according to the follow-*

ing recurrence:

$$\mathtt{w}^2_{X,Y}(t) = \mathtt{w}^2_{Y,X}(t) = \begin{cases} 0 & \textit{if } t \textit{ is a leaf in } T \\ (1-\mathtt{s}(t.l)) \cdot \mathtt{w}^2_{X,Y}(t.l) + (1-\mathtt{s}(t.r)) \cdot \mathtt{w}^2_{X,Y}(t.r) & \\ +\mathtt{exp}(-\mathtt{l}(t.l)) \cdot \mathtt{w}^1_X(t.l) \cdot \mathtt{exp}(-\mathtt{l}(t.r)) \cdot \mathtt{w}^1_Y(t.r) & \\ +\mathtt{exp}(-\mathtt{l}(t.l)) \cdot \mathtt{w}^1_Y(t.l) \cdot \mathtt{exp}(-\mathtt{l}(t.r)) \cdot \mathtt{w}^1_X(t.r) & \textit{otherwise} \end{cases} \tag{B.14}$$

After computing $\mathtt{w}^1_X$ *and* $\mathtt{w}^1_Y$ *(Lemma B.1), it takes* $O((a+b)^2 n)$ *time to compute* $\mathtt{w}^2_{X,Y}(t)$ *for all vertices* $t \in V(T)$ *and for all pairs of taxa* $X, Y \in \mathscr{S} \cup \mathscr{A}$*. Likewise, it takes* $O((a+b)^2 n)$ *space to access these doublet auxiliary values later. The time and storage complexity drops to* $O(b^2 n)$ *when only pairs of artificial taxa are considered (i.e.,* $X, Y \in \mathscr{A}$*).*

Proof. In the base case, we have $\mathtt{w}^2_{X,Y}(t) = 0$ because it is impossible to select the two leaves when t is a leaf. Otherwise t is a non-leaf vertex, so we split the sum into four cases:

$$\begin{aligned} \mathtt{w}^2_{X,Y}(t) &= \sum_{\substack{x,y \in \mathtt{below}(t) \\ L(x)=X, L(y)=Y}} \mathtt{I}(x,y) \cdot \Delta\mathtt{W}(x,y|t) \\ &= \sum_{\substack{x,y \in \mathtt{below}(t.l) \\ L(x)=X, L(y)=Y}} \mathtt{I}(x,y) \cdot \Delta\mathtt{W}(x,y|t) + \sum_{\substack{x,y \in \mathtt{below}(t.r) \\ L(x)=X, L(y)=Y}} \mathtt{I}(x,y) \cdot \Delta\mathtt{W}(x,y|t) \\ &+ \sum_{\substack{x \in \mathtt{below}(t.l) \\ y \in \mathtt{below}(t.r) \\ L(x)=X, L(y)=Y}} \mathtt{I}(x,y) \cdot \Delta\mathtt{W}(x,y|t) + \sum_{\substack{x \in \mathtt{below}(t.r) \\ y \in \mathtt{below}(t.l) \\ L(x)=X, L(y)=Y}} \mathtt{I}(x,y) \cdot \Delta\mathtt{W}(x,y|t). \end{aligned}$$

where

$$\Delta\mathtt{W}(x,y|t) = \sum_{\substack{x,y \in \mathtt{below}(t) \\ L(x)=X, L(y)=Y}} \mathtt{I}(x,y) \cdot \mathtt{exp}\Big(\sum_{e \in x,y \to u} -\mathtt{l}(e) \Big) \cdot \prod_{e \in u \to t} (1-\mathtt{s}(e))$$

with $u = lca(x,y)$.

In the first and second terms, both x and y are from the same subtree so we recurse on the subtree. Assume $x, y \in v.l$ (the other case is similar) and we have

$$
\begin{aligned}
&\sum_{\substack{x,y\in \texttt{below}(t.l)\\ L(x)=X, L(y)=Y}} \texttt{I}(x,y)\cdot \Delta\texttt{W}(x,y|t)\\
=&\sum_{\substack{x,y\in \texttt{below}(t.l)\\ L(x)=X, L(y)=Y}} \texttt{I}(x,y)\cdot \texttt{exp}(\sum_{e\in x,y\to u} -\texttt{l}(e))\cdot \prod_{e\in u\to t}(1-\texttt{s}(e))\\
=&(1-\texttt{s}(t.l))\cdot\left(\sum_{\substack{x,y\in \texttt{below}(t.l)\\ L(x)=X, L(y)=Y}} \texttt{I}(x,y)\cdot \texttt{exp}(\sum_{e\in x,y\to u} -\texttt{l}(e))\cdot \prod_{e\in u\to t.l}(1-\texttt{s}(e))\right)\\
=&(1-\texttt{s}(t.l))\cdot\sum_{\substack{x,y\in \texttt{below}(t.l)\\ L(x)=X, L(y)=Y}} \texttt{I}(x,y)\cdot \Delta\texttt{W}(x,y|t.l)\\
=&(1-\texttt{s}(t.l))\cdot \texttt{w}^2_{X,Y}(t.l).
\end{aligned}
$$

In the third and fourth terms, x and y are from different subtrees so their lowest common ancestor is t. Assume $x \in \texttt{below}(t.l)$ and $y \in \texttt{below}(t.r)$. We may independently choose any x from $below(t.l)$ and any y from $below(t.r)$ to form a doublet. As a result, the total weight of doublets is a product

of the total weights of singlets in the subtrees:

$$
\begin{aligned}
&\sum_{\substack{x\in\texttt{below}(t.l)\\ y\in\texttt{below}(t.r)\\ L(x)=X,L(y)=Y}} \texttt{I}(x,y)\cdot\Delta\texttt{W}(x,y|t)\\
=&\sum_{\substack{x\in\texttt{below}(t.l)\\ y\in\texttt{below}(t.r)\\ L(x)=X,L(y)=Y}} \texttt{I}(x,y)\cdot\texttt{exp}\Big(\sum_{e\in x,y\to t}-\texttt{l}(e)\Big)\\
=&\Bigg(\sum_{\substack{x\in\texttt{below}(t.l)\\ L(x)=X}} \texttt{I}(x)\cdot\texttt{exp}\Big(\sum_{e\in x\to t}-\texttt{l}(e)\Big)\Bigg)\cdot\Bigg(\sum_{\substack{y\in\texttt{below}(t.r)\\ L(y)=Y}} \texttt{I}(y)\cdot\texttt{exp}\Big(\sum_{e\in y\to t}-\texttt{l}(e)\Big)\Bigg)\\
=&\texttt{exp}(-\texttt{l}(t.l))\cdot\Bigg(\sum_{\substack{x\in\texttt{below}(t.l)\\ L(x)=X}} \texttt{I}(x)\cdot\texttt{exp}\Big(\sum_{e\in x\to t.l}-\texttt{l}(e)\Big)\Bigg)\\
&\quad\cdot\texttt{exp}(-\texttt{l}(t.l))\cdot\Bigg(\sum_{\substack{y\in\texttt{below}(t.r)\\ L(y)=Y}} \texttt{I}(y)\cdot\texttt{exp}\Big(\sum_{e\in y\to t.r}-\texttt{l}(e)\Big)\Bigg)\\
=&\texttt{exp}(-\texttt{l}(t.l))\cdot\texttt{w}^1_X(t.l)\cdot\texttt{exp}(-\texttt{l}(t.r))\cdot\texttt{w}^1_Y(t.r)
\end{aligned}
$$

Adding the weights from four terms gives the final recurrence:

$$
\begin{aligned}
\texttt{w}^2_{X,Y}(t) =&(1-\texttt{s}(t.l))\cdot\texttt{w}^2_{X,Y}(t.l)+(1-\texttt{s}(t.r))\cdot\texttt{w}^2_{X,Y}(t.r)\\
&+\texttt{exp}(-\texttt{l}(t.l))\cdot\texttt{w}^1_X(t.l)\cdot\texttt{exp}(-\texttt{l}(t.r))\cdot\texttt{w}^1_Y(t.r)\\
&+\texttt{exp}(-\texttt{l}(t.l))\cdot\texttt{w}^1_Y(t.l)\cdot\texttt{exp}(-\texttt{l}(t.r))\cdot\texttt{w}^1_X(t.r).
\end{aligned}
$$

After precomputation, $\texttt{w}^2_{X,Y}(t)$ can be computed by traversing the $O(n)$ vertices of gene tree T in a postorder fashion. Evaluating the recurrence for vertex t and pair of taxa X,Y takes $O(1)$ time and storing the resulting auxiliary value takes $O(1)$ space. Repeating for all $O((a+b)^2)$ pairs of taxa $X,Y\in\mathscr{S}\cup\mathscr{A}$ gives the time and storage complexity. □

Definition B.6 (doublet auxiliary values above t)**.** *Let $\overline{\texttt{w}}^2_{X,Y}(t)$ be the sum of the weights from leaf*

pairs x and y in the subtree above *vertex t in gene tree T such that one leaf (w.l.o.g. x) is labeled X, the other leaf (w.l.o.g. y) is labeled Y, and $X \neq Y$. More precisely, the doublet auxiliary values of X and Y* above *t are defined as*

$$\overline{\mathtt{w}}^2_{X,Y}(t) = \overline{\mathtt{w}}^2_{Y,X}(t) = \sum_{\substack{x,y \in \mathtt{above}(t) \\ L(x)=X, L(y)=Y}} \mathtt{I}(x,y) \cdot \mathtt{exp}\Big(\sum_{e \in u \to x,y} -\mathtt{l}(e) \Big) \cdot \prod_{e \in t \to u} (1 - \mathtt{s}(e)) \tag{B.15}$$

where u is either $lca(x,y)$, $lca(x,t)$ or $lca(y,t)$ depending on which of these three vertices is farthest from the root of T.

Lemma B.4 (double auxiliary values above *t*)**.** *$\overline{\mathtt{w}}^2_{X,Y}(t)$ can be computed according to the following recurrence:*

$$\overline{\mathtt{w}}^2_{X,Y}(t) = \overline{\mathtt{w}}^2_{Y,X}(t) = \begin{cases} 0 & \text{if } t \text{ is the root of } T \\ (1 - \mathtt{s}(t)) \cdot \overline{\mathtt{w}}^2_{X,Y}(t.p) & \\ \quad +(1 - \mathtt{s}(t)) \cdot (1 - \mathtt{s}(t.s)) \cdot \mathtt{w}^2_{X,Y}(t.s) & \\ +(1 - \mathtt{s}(t)) \cdot \mathtt{exp}(-\mathtt{l}(t.s)) \cdot \overline{\mathtt{w}}^1_X(t.p) \cdot \mathtt{w}^1_Y(t.s) & \\ +(1 - \mathtt{s}(t)) \cdot \mathtt{exp}(-\mathtt{l}(t.s)) \cdot \mathtt{w}^1_X(t.s) \cdot \overline{\mathtt{w}}^1_Y(t.p) & \text{otherwise} \end{cases} \tag{B.16}$$

After computing $\mathtt{w}^1_X$ and $\mathtt{w}^1_Y$ (Lemma B.1), $\overline{\mathtt{w}}^1_X$ and $\overline{\mathtt{w}}^1_Y$ (Lemma B.2), and $\mathtt{w}^2_{X,Y}$ (Lemma B.3), the time and space complexity for $\overline{\mathtt{w}}^2_{X,Y}$ is the same as for $\mathtt{w}^2_{X,Y}$ (Lemma B.3).

Proof. In the base case, we have $\overline{\mathtt{w}}^2_{X,Y}(t) = 0$ because the subtree above the root t is empty. Otherwise t is a non-root vertex, so we divide $\overline{\mathtt{w}}^2_{X,Y}(t)$ into four parts, which is similar to the proof for

$\mathtt{w}^2_{X,Y}(t)$:

$$\begin{aligned}
\overline{\mathtt{w}}^2_{X,Y}(t) &= \sum_{\substack{x,y\in\mathtt{above}(t)\\ L(x)=X,L(y)=Y}} \mathtt{I}(x,y)\cdot\Delta\overline{\mathtt{W}}(x,y|t) \\
&= \sum_{\substack{x,y\in\mathtt{above}(t.p)\\ L(x)=X,L(y)=Y}} \mathtt{I}(x,y)\cdot\Delta\overline{\mathtt{W}}(x,y|t) + \sum_{\substack{x,y\in\mathtt{below}(t.s)\\ L(x)=X,L(y)=Y}} \mathtt{I}(x,y)\cdot\Delta\overline{\mathtt{W}}(x,y|t) \\
&+ \sum_{\substack{x\in\mathtt{above}(t.p)\\ y\in\mathtt{below}(t.s)\\ L(x)=X,L(y)=Y}} \mathtt{I}(x,y)\cdot\Delta\overline{\mathtt{W}}(x,y|t) + \sum_{\substack{x\in\mathtt{below}(t.s)\\ y\in\mathtt{above}(t.p)\\ L(x)=X,L(y)=Y}} \mathtt{I}(x,y)\cdot\Delta\overline{\mathtt{W}}(x,y|t)
\end{aligned}$$

where

$$\Delta\overline{\mathtt{W}}(x,y|t) = \mathtt{I}(x,y)\cdot\mathtt{exp}(\sum_{e\in u\to x,y} -\mathtt{l}(e))\cdot\prod_{e\in t\to u}(1-\mathtt{s}(e))$$

and u is $lca(x,y)$, $lca(x,t)$ or $lca(y,t)$ depending on which is farthest from the root of T. It is easy to see that u is an ancestor of $t.p.p$ for the first term, u is a descendant of $t.s$ for the second term, and u is $t.p$ for third and fourth terms. Expanding the four terms in a similar fashion to Lemma B.3 and adding the weights together gives the final recurrence:

$$\begin{aligned}
\overline{\mathtt{w}}^2_{X,Y}(t) =& (1-\mathtt{s}(t))\cdot\overline{\mathtt{w}}^2_{X,Y}(t.p) \\
&+ (1-\mathtt{s}(t))\cdot(1-\mathtt{s}(t.s))\cdot\mathtt{w}^2_{X,Y}(t.s) \\
&+ (1-\mathtt{s}(t))\cdot\mathtt{exp}(-\mathtt{l}(t.s))\cdot\overline{\mathtt{w}}^1_X(t.p)\cdot\mathtt{w}^1_Y(t.s) \\
&+ (1-\mathtt{s}(t))\cdot\mathtt{exp}(-\mathtt{l}(t.s))\cdot\mathtt{w}^1_X(t.s)\cdot\overline{\mathtt{w}}^1_Y(t.p)
\end{aligned}$$

After precomputation, $\overline{\mathtt{w}}^2_{X,Y}(t)$ can be computed by traversing the $O(n)$ vertices of gene tree T in a preorder fashion. The remainder of the complexity analysis for $\overline{\mathtt{w}}^2_{X,Y}(t)$ is the same as for $\mathtt{w}^2_{X,Y}(t)$

(Lemma B.3). □

Definition B.7 (bad triplet auxiliary values)**.** *Let* $\mathtt{w}_{X,Y}^{x|z,w}(t)$ *be the sum of the weights from three leaves x, z, and w in the subtree below t, where the topology of the triplet of* (x,z,w) *is* $x|z,w$*, the outgroup of the triplet (x) is labeled by X, and ingroups (z and w) satisfy* $L(z) \neq L(w) \neq X \neq Y$*. Thus the triplet corresponds to a bad quartet for* X,Y*. The bad triplet auxiliary value below t is defined as*

$$\mathtt{w}_{X,Y}^{x|z,w}(t) = \sum_{\substack{x,z,w \in \mathtt{below}(t) \\ L(x)=X, L(z)\neq L(w)\neq X\neq Y \\ t(x,z,w)=x|z,w}} \mathtt{I}(x,z,w) \cdot \exp\Big(\sum_{\substack{e\in x,t\to u \\ z,w\to v}} -\mathtt{l}(e)\Big) \cdot \prod_{e\in u\to v} (1-\mathtt{s}(e)) \tag{B.17}$$

where u and v are anchor nodes of the quartet of $t, x|z,w$ *and we assume u is closer to x than v. Unlike the case of doublets,* $\mathtt{w}_{X,Y}^{x|z,w}(t) \neq \mathtt{w}_{Y,X}^{x|z,w}(t)$ *because the former means that the outgroup x is labeled X and the latter means the outgroup x is labeled Y (in either case, the ingroup* z,w *must satisfy* $L(z) \neq L(w) \neq X \neq Y$*). Lastly, observe that* $\mathtt{w}_{X,Y}^{x|z,w}(t)$ *is equal for all* $Y \in \mathscr{S}$*; we use* $\mathtt{w}_{X,0}^{x|z,w}(t)$ *to denote the bad triplet auxiliary value for any* $Y \in \mathscr{S}$*.*

Lemma B.5 (bad triplets auxiliary values)**.** $\mathtt{w}_{X,Y}^{x|z,w}(t)$ *can be computed according to the following recurrence:*

$$\mathtt{w}_{X,Y}^{x|z,w}(t) = \begin{cases} 0 & \text{if } t \text{ is a leaf in } T \\ \exp(-\mathtt{l}(t.l)) \cdot \mathtt{w}_{X,Y}^{x|z,w}(t.l) + \exp(-\mathtt{l}(t.r)) \cdot \mathtt{w}_{X,Y}^{x|z,w}(t.r) & \\ +\exp(-\mathtt{l}(t.l)) \cdot \mathtt{w}_X^1(t.l) \cdot (1-\mathtt{s}(t.r)) \cdot \sum_{\substack{Z,W\in\mathscr{S}\cup\mathscr{A} \\ Z\neq W\neq X\neq Y}} \mathtt{w}_{Z,W}^2(t.r) & \\ +\exp(-\mathtt{l}(t.r)) \cdot \mathtt{w}_X^1(t.r) \cdot (1-\mathtt{s}(t.l)) \cdot \sum_{\substack{Z,W\in\mathscr{S}\cup\mathscr{A} \\ Z\neq W\neq X\neq Y}} \mathtt{w}_{Z,W}^2(t.l) & \text{otherwise} \end{cases} \tag{B.18}$$

where the summation enumerates all unordered *pairs of taxa* $Z,W \in \mathscr{S} \cup \mathscr{A}$ *such that* $X \neq Y \neq$

$Z \neq W$ *to avoid double counting. After computing* $\mathtt{w}^1_X$ *(Lemma B.1) and* $\mathtt{w}^2_{Z,W}$ *for all pairs of taxa* $Z,W \in \mathscr{S} \cup \mathscr{A}$ *(Lemma B.3), it takes* $O((a+b)^4 bn)$ *time to compute* $\mathtt{w}^{x|z,w}_{X,Y}(t)$ *for all vertices* $t \in V(T)$ *and for all unordered pairs of taxa* $X,Y \in \mathscr{S} \cup \mathscr{A}$*. Likewise, it takes* $O((a+b)^2 n)$ *space to access the bad triplet auxiliary values later. The time and storage complexity drops to* $O((a+b)^2 b^2 n)$ *and* $O(b^2 n)$*, respectively, when restricting* X *to be an artificial taxa (i.e.,* $X \in \mathscr{A}$ *and* $Y \in \mathscr{A} \cup \{0\}$*).*

Proof. In the base case, $\mathtt{w}^{x|z,w}_{X,Y}(t) = 0$ because t is a leaf. Otherwise, we must consider the positions of x, z, and w in the subtree below t. Since the triplet is part of a quartet contributing a bad edge between X and Y, z and w must be in the same subtree after deleting edges on the path $x \to t$ and their endpoints from T. Thus, it is impossible that $z \in \mathtt{below}(t.l)$ and $w \in \mathtt{below}(t.r)$ (or vise versa) and we only have four cases to consider:

(a) $x,z,w \in \mathtt{below}(t.l)$

(b) $x \in \mathtt{below}(t.l)$ and $z,w \in \mathtt{below}(t.r)$

(c) $x \in \mathtt{below}(t.r)$ and $z,w \in \mathtt{below}(t.l)$ (symmetric with case (b)

(d) $x,z,w \in \mathtt{below}(t.r)$ (symmetric with case (a))

We split $\mathtt{w}^{x|z,w}_{X,Y}(t)$ into four terms according to the cases above:

$$\begin{aligned}
\mathtt{w}^{x|z,w}_{X,Y}(t) &= \sum_{\substack{x,z,w \in \mathtt{below}(t) \\ L(z) \neq L(w) \neq X,Y \\ t(x,z,w) = x|z,w}} \mathtt{I}(x,z,w) \cdot \Delta\mathtt{W}(x,t|z,w) \\
&= \sum_{\substack{x,z,w \in \mathtt{below}(t.l) \\ L(z) \neq L(w) \neq X,Y \\ t(x,z,w) = x|z,w}} \mathtt{I}(x,z,w) \cdot \Delta\mathtt{W}(x,t|z,w) + \sum_{\substack{x,z,w \in \mathtt{below}(t.r) \\ L(z) \neq L(w) \neq X,Y \\ t(x,z,w) = x|z,w}} \mathtt{I}(x,z,w) \cdot \Delta\mathtt{W}(x,t|z,w) \\
&+ \sum_{\substack{x \in \mathtt{below}(t.l), \\ z,w \in \mathtt{below}(t.r) \\ L(z) \neq L(w) \neq X,Y \\ t(x,z,w) = x|z,w}} \mathtt{I}(x,z,w) \cdot \Delta\mathtt{W}(x,t|z,w) + \sum_{\substack{z,w \in \mathtt{below}(t.l), \\ x \in \mathtt{below}(t.r) \\ L(z) \neq L(w) \neq X,Y \\ t(x,z,w) = x|z,w}} \mathtt{I}(x,z,w) \cdot \Delta\mathtt{W}(x,t|z,w).
\end{aligned}$$

We only prove the recurrence of the cases (a) and (b) as the other two cases are similar due to symmetry. In the **case (a)**, we have

$$\begin{aligned}
&\sum_{\substack{x,z,w\in\texttt{below}(t.l)\\ L(z)\neq L(w)\neq X,Y\\ t(x,z,w)=x|z,w}} \texttt{I}(x,z,w)\cdot\Delta\texttt{W}(x,t|z,w)\\
=&\sum_{\substack{x,z,w\in\texttt{below}(t.l)\\ L(z)\neq L(w)\neq X,Y\\ t(x,z,w)=x|z,w}} \texttt{I}(x,z,w)\cdot\texttt{exp}(\sum_{\substack{e\in x,t\to u\\ z,w\to v}}-\texttt{l}(e))\cdot\prod_{e\in u\to v}(1-\texttt{s}(e))\\
=&\texttt{exp}(-\texttt{l}(t.l))\cdot\sum_{\substack{x,z,w\in\texttt{below}(t.l)\\ L(z)\neq L(w)\neq X,Y\\ t(x,z,w)=x|z,w}} \texttt{I}(x,z,w)\cdot\texttt{exp}(\sum_{\substack{e\in x,t.l\to u\\ z,w\to v}}-\texttt{l}(e))\cdot\prod_{e\in u\to v}(1-\texttt{s}(e))\\
=&\texttt{exp}(-\texttt{l}(t.l))\cdot\texttt{w}_{X,Y}^{x|z,w}(t.l).
\end{aligned}$$

In the **case (b)**, the topology must be $x|z,w$ so we have

$$
\begin{aligned}
&\sum_{\substack{x\in\texttt{below}(t.l),\\ z,w\in\texttt{below}(t.r)\\ L(z)\neq L(w)\neq X,Y\\ t(x,z,w)=x|z,w}} \texttt{I}(x,z,w)\cdot\Delta\texttt{W}(x,t|z,w)\\
=&\sum_{\substack{x\in\texttt{below}(t.l),\\ z,w\in\texttt{below}(t.r)\\ L(z)\neq L(w)\neq X,Y}} \texttt{I}(x,z,w)\cdot\texttt{exp}(\sum_{\substack{e\in x\to t\\ z,w\to v}}-\texttt{l}(e))\cdot\prod_{e\in v\to t}(1-\texttt{s}(e))\\
=&\sum_{\substack{x\in\texttt{below}(t.l),\\ z,w\in\texttt{below}(t.r)\\ L(z)\neq L(w)\neq X,Y}} \texttt{I}(x)\cdot\texttt{exp}(-\sum_{e\in x\to t}\texttt{l}(e))\cdot\texttt{I}(z,w)\cdot\texttt{exp}(\sum_{e\in z,w\to v}-\texttt{l}(e))\cdot\prod_{e\in v\to t}(1-\texttt{s}(e))\\
=&\Bigg(\sum_{x\in\texttt{below}(t.l)}\texttt{I}(x)\cdot\texttt{exp}(-\sum_{e\in x\to t}\texttt{l}(e))\Bigg)\\
&\cdot\Bigg(\sum_{\substack{z,w\in\texttt{below}(t.r)\\ L(z)\neq L(w)\neq X,Y}}\texttt{I}(z,w)\cdot\texttt{exp}(\sum_{e\in z,w\to v}-\texttt{l}(e))\cdot\prod_{e\in v\to t}(1-\texttt{s}(e))\Bigg)\\
=&\texttt{exp}(-\texttt{l}(t.l))\cdot\texttt{w}^1_X(t.l)\cdot(1-\texttt{s}(t.r))\\
&\cdot\sum_{\substack{Z,W\in\mathscr{S}\cup\mathscr{A}\\ Z\neq W\neq X\neq Y}}\sum_{L(z)=Z,L(w)=W}\texttt{I}(z,w)\cdot\texttt{exp}(\sum_{e\in z,w\to v}-\texttt{l}(e))\cdot\prod_{e\in v\to t}(1-\texttt{s}(e))\\
=&\texttt{exp}(-\texttt{l}(t.l))\cdot\texttt{w}^1_X(t.l)\cdot(1-\texttt{s}(t.r))\cdot\sum_{\substack{Z,W\in\mathscr{S}\cup\mathscr{A}\\ Z\neq W\neq X\neq Y}}\texttt{w}^2_{Z,W}(t.r)
\end{aligned}
$$

where the summation enumerates all unordered pairs of taxa Z,W from the subproblem such that $X\neq Y\neq Z\neq W$ to avoid double counting. Adding the weights of the four cases together gives the

final recurrence:

$$\begin{aligned}
\mathtt{w}^{x|z,w}_{X,Y}(t) =& \mathtt{exp}(-\mathtt{l}(t.l)) \cdot \mathtt{w}^{x|z,w}_{X,Y}(t.l) + \mathtt{exp}(-\mathtt{l}(t.r)) \cdot \mathtt{w}^{x|z,w}_{X,Y}(t.r) \\
&+ \mathtt{exp}(-\mathtt{l}(t.l)) \cdot \mathtt{w}^{1}_{X}(t.l) \cdot (1-\mathtt{s}(t.r)) \cdot \frac{1}{2} \cdot \sum_{\substack{Z,W \in \mathscr{S} \cup \mathscr{A} \\ Z \neq W \neq X \neq Y}} \mathtt{w}^{2}_{Z,W}(t.r) \\
&+ \mathtt{exp}(-\mathtt{l}(t.r)) \cdot \mathtt{w}^{1}_{X}(t.r) \cdot (1-\mathtt{s}(t.l)) \cdot \frac{1}{2} \cdot \sum_{\substack{Z,W \in \mathscr{S} \cup \mathscr{A} \\ Z \neq W \neq X \neq Y}} \mathtt{w}^{2}_{Z,W}(t.l).
\end{aligned}$$

After precomputation, $\mathtt{w}^{x|z,w}_{X,Y}(t)$ can be computed by traversing the $O(n)$ vertices of in a postorder traversal. Evaluating the recurrence for vertex t and pair of taxa X,Y takes $O((a+b)^2)$ time and storing the resulting auxiliary value requires $O(1)$ space. Repeating for all $O((a+b)^2)$ unordered pairs of taxa $X,Y \in \mathscr{S} \cup \mathscr{A}$ gives the time and storage complexity. □

Definition B.8 (good triplet auxiliary values)**.** *Let $\mathtt{w}^{x,z|w}_{X,Y}(t)$ be the sum of the weights from three leaves x, z, and w in the subtree below t, where the topology of the triplet of (x,z,w) has one of the ingroups (w.l.o.g. x) labeled X, and the other two taxa satisfy $L(z) \neq L(w) \neq X \neq Y$. Thus, the triplet corresponds to a good quartet when selecting a leaf labeled Y above t. The good triplet auxiliary value below t is defined as*

$$\mathtt{w}^{x,z|w}_{X,Y}(t) = \sum_{\substack{x,z,w \in \mathtt{below}(t) \\ L(x)=X, L(z)\neq L(w)\neq X\neq Y \\ t(x,z,w)=x,z|w}} \mathtt{I}(x,z,w) \cdot \mathtt{exp}\Big(\sum_{\substack{e \in x,z \to u \\ t,w \to v}} -\mathtt{l}(e) \Big) \cdot \prod_{e \in u \to v} (1-\mathtt{s}(e)) \tag{B.19}$$

where u and v are anchor nodes of the quartet of $x,z|w,t$ and we assume u is the closer to x than v. Similar to bad triplet auxiliary values, $\mathtt{w}^{x,z|w}_{X,Y}(t) \neq \mathtt{w}^{x,z|w}_{Y,X}(t)$ and we use $\mathtt{w}^{x,z|w}_{X,0}(t)$ to denote the quantity for any $Y \in \mathscr{S}$.

Lemma B.6 (good triplet auxiliary values)**.** *$\mathtt{w}^{x,z|w}_{X,Y}(t)$ can be computed according to the following*

recurrence:

$$\mathtt{w}^{x,z|w}_{X,Y}(t) = \begin{cases} 0 & \text{if } t \text{ is a leaf in } T \\ \exp(-\mathtt{l}(t.l)) \cdot \mathtt{w}^{x,z|w}_{X,Y}(t.l) + \exp(-\mathtt{l}(t.r)) \cdot \mathtt{w}^{x,z|w}_{X,Y}(t.r) & \\ +(1-\mathtt{s}(t.l)) \cdot \exp(-\mathtt{l}(t.r)) & \\ \quad \cdot \sum_{\substack{Z \in \mathscr{S} \cup \mathscr{A} \\ Z \neq X,Y}} \sum_{\substack{W \in \mathscr{S} \cup \mathscr{A} \\ W \neq X,Y,Z}} \mathtt{w}^2_{X,Z}(t.l) \cdot \mathtt{w}^1_W(t.r) & \\ +(1-\mathtt{s}(t.r)) \cdot \exp(-\mathtt{l}(t.l)) & \\ \quad \cdot \sum_{\substack{Z \in \mathscr{S} \cup \mathscr{A} \\ Z \neq X,Y}} \sum_{\substack{W \in \mathscr{S} \cup \mathscr{A} \\ W \neq X,Y,Z}} \mathtt{w}^2_{X,Z}(t.r) \cdot \mathtt{w}^1_W(t.l) & \text{otherwise} \end{cases} \tag{B.20}$$

where pairs of taxa Z, W are ordered, as indicated by the nested summations, unlike when computing bad triplet auxiliary values. After computing $\mathtt{w}^1_W$ for all taxa $W \in \mathscr{S} \cup \mathscr{A}$ (Lemma B.1) and $\mathtt{w}^2_{X,Z}$ for all taxa $Z \in \mathscr{S} \cup \mathscr{A}$ (Lemma B.3), the time and space complexity for $\mathtt{w}^{x,z|w}_{X,Y}$ is the same as for $\mathtt{w}^{x|z,w}_{X,Y}$ (Lemma B.5).

Proof. In the base case, $\mathtt{w}^{x,z|w}_{X,Y}(t) = 0$ because t is a leaf. Otherwise, we must consider the positions of x, z, and w in the subtree below t. Since the triplet is part of a quartet contributing a good edge between X and Y, z and w cannot be in the same subtree after deleting edges on the path $x \to t$ and their endpoints from T. Thus, we only have four cases to consider:

(a) $x, z, w \in \mathtt{below}(t.l)$

(b) $x, z \in \mathtt{below}(t.l)$ and $w \in \mathtt{below}(t.r)$

(c) $w \in \mathtt{below}(t.l)$ and $x, z \in \mathtt{below}(t.r)$ (symmetric with case (b))

(d) $x, z, w \in \mathtt{below}(t.r)$ (symmetric with case (a))

We split $\mathtt{w}_{X,Y}^{x,z|w}(t)$ into four parts according to the cases above:

$$\begin{aligned}
\mathtt{w}_{X,Y}^{x,z|w}(t) &= \sum_{\substack{x,z,w\in\mathtt{below}(t)\\ L(z)\neq L(w)\neq X,Y\\ t(x,z,w)=x,z|w}} \mathtt{I}(x,z,w)\cdot\Delta\mathtt{W}(x,z|w,t) \\
&= \sum_{\substack{x,z,w\in\mathtt{below}(t.l)\\ L(z)\neq L(w)\neq X,Y\\ t(x,z,w)=x,z|w}} \mathtt{I}(x,z,w)\cdot\Delta\mathtt{W}(x,z|w,t) + \sum_{\substack{x,z,w\in\mathtt{below}(t.r)\\ L(z)\neq L(w)\neq X,Y\\ t(x,z,w)=x,z|w}} \mathtt{I}(x,z,w)\cdot\Delta\mathtt{W}(x,z|w,t) \\
&+ \sum_{\substack{x,z\in\mathtt{below}(t.l)\\ w\in\mathtt{below}(t.r)\\ L(z)\neq L(w)\neq X,Y\\ t(x,z,w)=x,z|w}} \mathtt{I}(x,z,w)\cdot\Delta\mathtt{W}(x,z|w,t) + \sum_{\substack{w\in\mathtt{below}(t.l)\\ x,z\in\mathtt{below}(t.r)\\ L(z)\neq L(w)\neq X,Y\\ t(x,z,w)=x,z|w}} \mathtt{I}(x,z,w)\cdot\Delta\mathtt{W}(x,z|w,t),
\end{aligned}$$

We only prove the cases (a) and (b) since the other two cases are similar due to symmetry. In the **case (a)**, all three leaves comes from the left subtree, so we recurse on $t.l$ and append the edge from t to $t.l$ to all triplets in $t.l$:

$$\begin{aligned}
&\sum_{\substack{x,z,w\in\mathtt{below}(t.l)\\ L(z)\neq L(w)\neq X,Y\\ t(x,z,w)=x,z|w}} \mathtt{I}(x,z,w)\cdot\Delta\mathtt{W}(x,z|w,t) \\
=&\sum_{\substack{x,z,w\in\mathtt{below}(t.l)\\ L(z)\neq L(w)\neq X,Y\\ t(x,z,w)=x,z|w}} \mathtt{I}(x,z,w)\cdot\mathtt{exp}\Big(\sum_{\substack{e\in x,z\to u\\ t,w\to v}} -\mathtt{l}(e)\Big)\cdot\prod_{e\in u\to v}(1-\mathtt{s}(e)) \\
=&\mathtt{exp}(-\mathtt{l}(t.l))\cdot\sum_{\substack{x,z,w\in\mathtt{below}(t.l)\\ L(z)\neq L(w)\neq X,Y\\ t(x,z,w)=x,z|w}} \mathtt{I}(x,z,w)\cdot\mathtt{exp}\Big(\sum_{\substack{e\in x,z\to u\\ t.l,w\to v}} -\mathtt{l}(e)\Big)\cdot\prod_{e\in u\to v}(1-\mathtt{s}(e)) \\
=&\mathtt{exp}(-\mathtt{l}(t.l))\cdot\sum_{\substack{x,z,w\in\mathtt{below}(t.l)\\ L(z)\neq L(w)\neq X,Y\\ t(x,z,w)=x,z|w}} \mathtt{I}(x,z,w)\cdot\Delta\mathtt{W}(x,z|w,t.l) \\
=&\mathtt{exp}(-\mathtt{l}(t.l))\cdot\mathtt{w}_{X,Y}^{x,z|w}(t.l)
\end{aligned}$$

In the **case (b)** where $x,z\in\mathtt{below}(t.l)$ and $w\in\mathtt{below}(t.r)$ so the topology of the triplet must be

$x,z|w$, the lowest common ancestor of x,z and w is t, i.e., t is the anchor for w,y in the quartet $x,z|w,y$. We then have

$$\sum_{\substack{x,z\in\texttt{below}(t.l)\\ w\in\texttt{below}(t.r)\\ L(z)\neq L(w)\neq X,Y\\ t(x,z,w)=x,z|w}} \texttt{I}(x,z,w)\cdot\Delta\texttt{W}(x,z|w,t)$$

$$=\sum_{\substack{x,z\in\texttt{below}(t.l)\\ w\in\texttt{below}(t.r)\\ L(z)\neq L(w)\neq X,Y}} \texttt{I}(x,z,w)\cdot\texttt{exp}(\sum_{\substack{e\in x,z\to u\\ w\to t}}-\texttt{l}(e))\cdot\prod_{e\in u\to t}(1-\texttt{s}(e))$$

$$=(1-\texttt{s}(t.l))\cdot\texttt{exp}(-\texttt{l}(t.r))\cdot\sum_{\substack{x,z\in\texttt{below}(t.l)\\ w\in\texttt{below}(t.r)\\ L(z)\neq L(w)\neq X,Y}} \texttt{I}(x,z,w)\cdot\texttt{exp}(\sum_{\substack{e\in x,z\to u\\ w\to t.r}}-\texttt{l}(e))\cdot\prod_{e\in u\to t.l}(1-\texttt{s}(e))$$

where

$$
\begin{aligned}
&\sum_{\substack{x,z\in\texttt{below}(t.l)\\ w\in\texttt{below}(t.r)\\ L(z)\neq L(w)\neq X,Y}} \texttt{I}(x,z,w)\cdot\texttt{exp}(\sum_{\substack{e\in x,z\to u\\ w\to t.r}}-\texttt{l}(e))\cdot\prod_{e\in u\to t.l}(1-\texttt{s}(e)) \\
=&\sum_{\substack{x,z\in\texttt{below}(t.l)\\ w\in\texttt{below}(t.r)\\ L(z)\neq L(w)\neq X,Y}} \left(\texttt{I}(x,z)\cdot\texttt{exp}(\sum_{e\in x,z\to u}-\texttt{l}(e))\cdot\prod_{e\in u\to t.l}(1-\texttt{s}(e))\right) \\
&\cdot\left(\texttt{I}(w)\cdot\texttt{exp}(\sum_{e\in w\to t.r}-\texttt{l}(e))\right) \\
=&\sum_{\substack{Z\in\mathscr{S}\cup\mathscr{A}\\ Z\neq X,Y}}\sum_{\substack{W\in\mathscr{S}\cup\mathscr{A}\\ W\neq X,Y,Z}}\sum_{\substack{x,z\in\texttt{below}(t.l)\\ L(x)=X,L(z)=Z}}\sum_{\substack{w\in\texttt{below}(t.r)\\ L(w)=W}}\left(\texttt{I}(x,z)\cdot\texttt{exp}(\sum_{e\in x,z\to u}-\texttt{l}(e))\cdot\prod_{e\in u\to t.l}(1-\texttt{s}(e))\right) \\
&\cdot\left(\texttt{I}(w)\cdot\texttt{exp}(\sum_{e\in w\to t.r}-\texttt{l}(e))\right) \\
=&\sum_{\substack{Z\in\mathscr{S}\cup\mathscr{A}\\ Z\neq X,Y}}\sum_{\substack{W\in\mathscr{S}\cup\mathscr{A}\\ W\neq X,Y,Z}}\left(\sum_{\substack{x,z\in\texttt{below}(t.l)\\ L(x)=X,L(z)=Z}}\texttt{I}(x,z)\cdot\texttt{exp}(\sum_{e\in x,z\to u}-\texttt{l}(e))\cdot\prod_{e\in u\to t.l}(1-\texttt{s}(e))\right) \\
&\cdot\left(\sum_{\substack{w\in\texttt{below}(t.r)\\ L(w)=W}}\texttt{I}(w)\cdot\texttt{exp}(\sum_{e\in w\to t.r}-\texttt{l}(e))\right) \\
=&\sum_{\substack{Z\in\mathscr{S}\cup\mathscr{A}\\ Z\neq X,Y}}\sum_{\substack{W\in\mathscr{S}\cup\mathscr{A}\\ W\neq X,Y,Z}}\texttt{w}^2_{X,Z}(t.l)\cdot\texttt{w}^1_W(t.r)
\end{aligned}
$$

Note we can write the sum of products as the products of sums in the proof above because $Z\neq W$. Also, it is guaranteed that z and w are from different subtrees so there are no double counting

problems. Adding the weights of the four cases together gives the final recurrence:

$$\begin{aligned}
\mathtt{w}^{x,z|w}_{X,Y}(t) =& \exp(-\mathtt{l}(t.l)) \cdot \mathtt{w}^{x,z|w}_{X,Y}(t.l) + \exp(-\mathtt{l}(t.r)) \cdot \mathtt{w}^{x,z|w}_{X,Y}(t.r) \\
&+ (1-\mathtt{s}(t.l)) \cdot \exp(-\mathtt{l}(t.r)) \cdot \sum_{\substack{Z \in \mathscr{S} \cup \mathscr{A} \\ Z \neq X,Y}} \sum_{\substack{W \in \mathscr{S} \cup \mathscr{A} \\ W \neq X,Y,Z}} \mathtt{w}^2_{X,Z}(t.l) \cdot \mathtt{w}^1_W(t.r) \\
&+ (1-\mathtt{s}(t.r)) \cdot \exp(-\mathtt{l}(t.l)) \cdot \sum_{\substack{Z \in \mathscr{S} \cup \mathscr{A} \\ Z \neq X,Y}} \sum_{\substack{W \in \mathscr{S} \cup \mathscr{A} \\ W \neq X,Y,Z}} \mathtt{w}^2_{X,Z}(t.r) \cdot \mathtt{w}^1_W(t.l)
\end{aligned}$$

After precomputation, $\mathtt{w}^{x,z|w}_{X,Y}(t)$ can be computed by traversing the $O(n)$ vertices of gene tree T in a postorder traversal. The remainder of the complexity analysis for $\mathtt{w}^{x,z|w}_{X,Y}(t)$ is the same as for $\mathtt{w}^{x|z,w}_{X,Y}(t)$ (Lemma B.5). □

B.3.2 Quartet Graph Construction from Auxiliary Values w

We are now ready to define equations for the bad edges $\Delta\mathbb{B}_t(X,Y)$ and good edges $\Delta\mathbb{G}_t(X,Y)$ based on the six auxiliary values computed for gene tree T.

Theorem B.1 (bad edges below t)**.** *The weight of bad edges between taxa X,Y below vertex t in gene tree T can be computed from the auxiliary values according to the following equation:*

$$\begin{aligned}
\Delta\mathbb{B}_t(X,Y) =& \mathtt{w}_Y(t.r) \cdot \exp(-\mathtt{l}(t.r) \cdot \mathtt{w}^{x|z,w}_{X,Y}(t.l) \cdot \exp(-\mathtt{l}(t.l)) \\
&+ \exp(-\mathtt{l}(t.l)) \cdot \mathtt{w}_X(t.l) \cdot \exp(-\mathtt{l}(t.r)) \cdot \mathtt{w}_Y(t.r) \cdot \sum_{\substack{Z,W \in \mathscr{S} \cup \mathscr{A} \\ Z \neq W \neq X \neq Y}} \overline{\mathtt{w}}_{Z,W}(t) \\
&+ \mathtt{w}_X(t.l) \cdot \exp(-\mathtt{l}(t.l) \cdot \mathtt{w}^{x|z,w}_{Y,X}(t.r) \cdot \exp(-\mathtt{l}(t.r)) \qquad \text{(B.21)}
\end{aligned}$$

where the summation enumerates all unordered *pairs of taxa $Z,W \in \mathscr{S} \cup \mathscr{A}$ such that $X \neq Y \neq Z \neq W$.*

Proof. From Definition B.1, we consider quartets $x,y|z,w$ such that $x \in \mathtt{below}(t.l)$, $y \in \mathtt{below}(t.r)$, $L(x) = X$, $L(y) = Y$, and $L(x) \neq L(y) \neq L(z) \neq L(w)$. Depending on the positions of z and w, we

divide the computation into several cases:

(a) $z, w \in \texttt{below}(t.l)$

(b) $z \in \texttt{below}(t.l)$ and $w \in \texttt{above}(t)$

(c) $z, w \in \texttt{above}(t)$

(d) $z \in \texttt{above}(t)$ and $w \in \texttt{below}(t.r)$ (symmetric with case (b))

(e) $z, w \in \texttt{below}(t.r)$ (symmetric with case (a))

(f) $z \in \texttt{below}(t.l)$ and $w \in \texttt{below}(t.r)$

Only case (a), case (c), and case (e) result in bad edges, as both z and w must be from the same subtree after deleting edges on the path $x \to y$ and their endpoints from T. We introduce how to compute the weight of the cases (a) and (c) and the algorithm for the case (e) is similar to the case (a) according to symmetry. In the **case (a)** where $z, w \in \texttt{below}(t.l)$, we have

$$
\begin{aligned}
&\Delta\mathbb{B}_t^{(a)}(X,Y) \\
=& \sum_{\substack{x\in\texttt{below}(t.l)\\ y\in\texttt{below}(t.r)\\ L(x)=X,L(y)=Y}} \sum_{\substack{z,w\in\texttt{below}(t.l)\\ L(z)\neq L(w)\neq X,Y\\ q(x,y,z,w)=x,y|z,w}} \Delta\texttt{W}(q)\cdot\texttt{I}(q) \\
=& \sum_{\substack{x\in\texttt{below}(t.l)\\ y\in\texttt{below}(t.r)\\ L(x)=X,L(y)=Y}} \sum_{\substack{z,w\in\texttt{below}(t.l)\\ L(z)\neq L(w)\neq X,Y\\ q(x,y,z,w)=x,y|z,w}} \texttt{exp}\Bigg(\sum_{\substack{e\in x,y\to u,\\ z,w\to v}} -\texttt{l}(e)\Bigg)\cdot\Bigg(\prod_{e\in u\to v}(1-\texttt{s}(e))\Bigg)\cdot\texttt{I}(x,y,z,w) \\
=& \Bigg(\sum_{\substack{y\in\texttt{below}(t.r)\\ L(y)=Y}} \texttt{I}(y)\cdot\texttt{exp}(\sum_{e\in y\to t} -\texttt{l}(e))\Bigg) \\
&\cdot\Bigg(\sum_{\substack{x,z,w\in\texttt{below}(t.l)\\ L(z)\neq L(w)\neq X,Y\\ t(x,z,w)=x|z,w}} \texttt{I}(x,z,w)\cdot\texttt{exp}(\sum_{\substack{e\in x,t\to u\\ z,w\to v}} -\texttt{l}(e))\cdot\prod_{e\in u\to v}(1-\texttt{s}(e))\Bigg) \\
=& \Bigg(\texttt{w}_Y^{\texttt{l}}(t.r)\cdot\texttt{exp}(-\texttt{l}(t.r)\Bigg)\cdot\Bigg(\texttt{w}_{X,Y}^{x|z,w}(t.l)\cdot\texttt{exp}(-\texttt{l}(t.l))\Bigg),
\end{aligned}
$$

where $\mathtt{w}_Y^1(t.r)$ counts the total weight of singlets below $t.r$ (which can be computed via the recurrence in Lemma B.1) and $\mathtt{w}_{X,Y}^{x|z,w}(t.l)$ counts the total weight of the triplets below $t.l$ whose topology is $x|z,w$ (which can be computed via the recurrence in Lemma B.5). In the **case (c)**, z and w are in the subtree above t.

$$
\begin{aligned}
&\Delta\mathbb{B}_t^{(c)}(X,Y)\\
=&\sum_{\substack{x\in\mathtt{below}(t.l)\\ y\in\mathtt{below}(t.r)\\ L(x)=X,L(y)=Y}}\ \sum_{\substack{z,w\in\mathtt{above}(t)\\ L(z)\neq L(w)\neq X,Y\\ q(x,y,z,w)=x,y|z,w}} \Delta\mathtt{W}(q)\cdot\mathtt{I}(q)\\
=&\sum_{\substack{x\in\mathtt{below}(t.l)\\ y\in\mathtt{below}(t.r)\\ L(x)=X,L(y)=Y}}\ \sum_{\substack{z,w\in\mathtt{above}(t)\\ L(z)\neq L(w)\neq X,Y\\ q(x,y,z,w)=x,y|z,w}} \mathtt{exp}\Big(\sum_{\substack{e\in x,y\to t,\\ z,w\to u}}-\mathtt{l}(e)\Big)\cdot\Big(\prod_{e\in u\to v}(1-\mathtt{s}(e))\Big)\cdot\mathtt{I}(x,y,z,w)\\
=&\Big(\sum_{\substack{x\in\mathtt{below}(t.l)\\ L(x)=X}}\mathtt{I}(x)\cdot\mathtt{exp}(\sum_{e\in x\to t}-\mathtt{l}(e))\Big)\cdot\Big(\sum_{\substack{y\in\mathtt{below}(t.r)\\ L(y)=Y}}\mathtt{I}(y)\cdot\mathtt{exp}(\sum_{e\in y\to t}-\mathtt{l}(e))\Big)\\
&\cdot\Big(\sum_{\substack{z,w\in\mathtt{above}(t)\\ L(z)\neq L(w)\neq X,Y}}\mathtt{I}(z,w)\cdot\mathtt{exp}(\sum_{e\in z,w\to v}-\mathtt{l}(e))\cdot\prod_{e\in v\to t}(1-\mathtt{s}(e))\Big)\\
=&\mathtt{exp}(-\mathtt{l}(t.l))\cdot\mathtt{w}_X(t.l)\cdot\mathtt{exp}(-\mathtt{l}(t.r))\cdot\mathtt{w}_Y(t.r)\cdot\\
&\Big(\sum_{\substack{Z,W\in\mathscr{S}\cup\mathscr{A}\\ Z\neq W\neq X\neq Y}}\ \sum_{L(z)=Z,L(w)=W}\mathtt{I}(z,w)\cdot\mathtt{exp}(\sum_{e\in z,w\to v}-\mathtt{l}(e))\cdot\prod_{e\in v\to t}(1-\mathtt{s}(e))\Big)\\
=&\mathtt{exp}(-\mathtt{l}(t.l))\cdot\mathtt{w}_X(t.l)\cdot\mathtt{exp}(-\mathtt{l}(t.r))\cdot\mathtt{w}_Y(t.r)\cdot\sum_{\substack{Z,W\in\mathscr{S}\cup\mathscr{A}\\ Z\neq W\neq X\neq Y}}\overline{\mathtt{w}}_{Z,W}(t).
\end{aligned}
$$

where $\overline{\mathtt{w}}_{X,Y}(t)$ counts the total weight of doublets in the subtree above t (which can be computed via the recurrence in Lemma B.3). Adding up weights from all three cases

$$\Delta\mathbb{B}_t(X,Y)=\sum_{\gamma\in\{a,c,e\}}\Delta\mathbb{B}_t^{(\gamma)}(X,Y).$$

gives us the final recurrence. □

Theorem B.2 (good edges below t)**.** *The weight of good edges between taxa X, Y below vertex t in gene tree T can be derived from auxiliary values using the following equation:*

$$\begin{aligned}
\Delta\mathbb{G}_t(X,Y) = {} & \mathtt{w}^1_Y(t.r)\cdot\exp(-\mathtt{l}(t.r))\cdot\mathtt{w}^{x,z|w}_{X,Y}(t.l)\cdot\exp(-\mathtt{l}(t.l)) \\
& + \exp(-\mathtt{l}(t.r))\cdot\mathtt{w}^1_Y(t.r)\cdot(1-\mathtt{s}(t.l))\cdot\textstyle\sum_{\substack{Z\in\mathscr{S}\cup\mathscr{A}\\ Z\neq X,Y}}\sum_{\substack{W\in\mathscr{S}\cup\mathscr{A}\\ W\neq X,Y,Z}}\mathtt{w}^2_{X,Z}(t.l)\cdot\overline{\mathtt{w}}^1_W(t) \\
& + (1-\mathtt{s}(t.l))\cdot(1-\mathtt{s}(t.r))\cdot\textstyle\sum_{\substack{Z\in\mathscr{S}\cup\mathscr{A}\\ Z\neq X,Y}}\sum_{\substack{W\in\mathscr{S}\cup\mathscr{A}\\ W\neq X,Y,Z}}\mathtt{w}^2_{X,Z}(t.l)\cdot\mathtt{w}^2_{Y,W}(t.r) \\
& + \exp(-\mathtt{l}(t.l))\cdot\mathtt{w}^1_X(t.l)\cdot(1-\mathtt{s}(t.r))\cdot\textstyle\sum_{\substack{Z\in\mathscr{S}\cup\mathscr{A}\\ Z\neq X,Y}}\sum_{\substack{W\in\mathscr{S}\cup\mathscr{A}\\ W\neq X,Y,Z}}\mathtt{w}^2_{Y,Z}(t.r)\cdot\overline{\mathtt{w}}^1_W(t) \\
& + \mathtt{w}^1_X(t.l)\cdot\exp(-\mathtt{l}(t.l))\cdot\mathtt{w}^{x,z|w}_{Y,X}(t.r)\cdot\exp(-\mathtt{l}(t.r))
\end{aligned} \tag{B.22}$$

where Z, W are ordered, as indicated by the nested summations, unlike when computing bad edges $\Delta\mathbb{B}_t(X,Y)$.

Proof. From Definition B.1, we consider quartets $x,z|y,w$ such that $x\in\mathtt{below}(t.l)$, $y\in\mathtt{below}(t.r)$, $L(x)=X$, $L(y)=Y$, and $L(x)\neq L(y)\neq L(z)\neq L(w)$. Depending on the positions of z and w, we divide the computation into several cases:

(a) $z,w\in\mathtt{below}(t.l)$

(b) $z\in\mathtt{below}(t.l)$ and $w\in\mathtt{above}(t)$

(c) $z,w\in\mathtt{above}(t)$

(d) $z\in\mathtt{above}(t)$ and $w\in\mathtt{below}(t.r)$ (symmetric with case (b))

(e) $z,w\in\mathtt{below}(t.r)$ (symmetric with case (a))

(f) $z\in\mathtt{below}(t.l)$ and $w\in\mathtt{below}(t.r)$

Case (c) can be eliminated because when z,w are from above $t.p$ (and x,y are from below t), the quartet must have topology $x,y|z,w$, which corresponds to a bad edge between X,Y. Cases (a) and

(e) are similar to each other and so are cases (b) and (d) according to the symmetry. We therefore only elaborate on the cases (a), (b), and (f). In the **case (a)**, both z and w are from $below(t.l)$ so x, z, and w form a triplet in $below(t.l)$, whose topology has to be $x,z|w$. All triplets of (x,z,w) in $below(t.l)$ with a topology of $x,z|w$ is able to form a valid quartet of topology $x,z|y,w$ with $y \in \texttt{below}(t.r)$ as long as $L(z), L(w) \neq Y$. In this case, the value of $\Delta\mathbb{G}_t^{(a)}(X,Y)$ is

$$
\begin{aligned}
&\Delta\mathbb{G}_t^{(a)}(X,Y) \\
&= \sum_{\substack{x\in\texttt{below}(t.l)\\ y\in\texttt{below}(t.r)\\ L(x)=X,L(y)=Y}} \sum_{\substack{z,w\in\texttt{below}(t.l)\\ L(z)\neq L(w)\neq X,Y\\ q(x,y,z,w)=x,z|y,w}} \Delta\texttt{W}(q)\cdot\texttt{I}(q) \\
&= \sum_{\substack{x\in\texttt{below}(t.l)\\ y\in\texttt{below}(t.r)\\ L(x)=X,L(y)=Y}} \sum_{\substack{z,w\in\texttt{below}(t.l)\\ L(z)\neq L(w)\neq X,Y\\ q(x,y,z,w)=x,z|y,w}} \texttt{exp}\left(\sum_{\substack{e\in x,z\to u,\\ y,w\to v}} -\texttt{l}(e)\right)\cdot\left(\prod_{e\in u\to v}(1-\texttt{s}(e))\right)\cdot\texttt{I}(x,y,z,w) \\
&= \left(\sum_{\substack{y\in\texttt{below}(t.r)\\ L(y)=Y}} \texttt{I}(y)\cdot\texttt{exp}(\sum_{e\in y\to t} -\texttt{l}(e))\right) \\
&\quad\cdot\left(\sum_{\substack{x,z,w\in\texttt{below}(t.l)\\ L(z)\neq L(w)\neq X,Y\\ t(x,z,w)=x,z|w}} \texttt{I}(x,z,w)\cdot\texttt{exp}(\sum_{\substack{e\in x,z\to u\\ t,w\to v}} -\texttt{l}(e))\cdot\prod_{e\in u\to v}(1-\texttt{s}(e))\right) \\
&= \left(\texttt{w}_Y^1(t.r)\cdot\texttt{exp}(-\texttt{l}(t.r))\right)\cdot\left(\texttt{w}_{X,Y}^{x,z|w}(t.l)\cdot\texttt{exp}(-\texttt{l}(t.l))\right)
\end{aligned}
$$

where $\texttt{w}_Y^1(t.r)$ counts the total weight of singlets below t_r (which can be computed via the recurrence in Lemma B.1) and $\texttt{w}_{X,Y}^{x,z|w}(t.l)$ counts the total weight of triplets below $t.l$ whose topology is $x,z|w$ (which can be computed via the recurrence in Lemma B.6). In the **case (b)**, one of the leaves in the quartet is from the subtree above the node t so the anchor node close to y and w is t. In this

case, the total weight of the quartets is:

$$
\begin{aligned}
&\Delta\mathbb{G}_t^{(b)}(X,Y) \\
=&\sum_{\substack{x\in\texttt{below}(t.l)\\ y\in\texttt{below}(t.r)\\ L(x)=X,L(y)=Y}}\ \sum_{\substack{z\in\texttt{below}(t.l)\\ w\in\texttt{above}(t)\\ L(z)\neq L(w)\neq X,Y\\ q(x,y,z,w)=x,z|y,w}} \Delta\texttt{W}(q)\cdot\texttt{I}(q) \\
=&\sum_{\substack{x\in\texttt{below}(t.l)\\ y\in\texttt{below}(t.r)\\ L(x)=X,L(y)=Y}}\ \sum_{\substack{z\in\texttt{below}(t.l)\\ w\in\texttt{above}(t)\\ L(z)\neq L(w)\neq X,Y\\ q(x,y,z,w)=x,z|y,w}} \texttt{exp}\Bigg(\sum_{\substack{e\in x,z\to u,\\ y,w\to t}}-\texttt{l}(e)\Bigg)\cdot\Bigg(\prod_{e\in u\to t}(1-\texttt{s}(e))\Bigg)\cdot\texttt{I}(x,y,z,w) \\
=&\Bigg(\sum_{\substack{y\in\texttt{below}(t.r)\\ L(y)=Y}}\texttt{I}(y)\cdot\texttt{exp}(\sum_{e\in y\to t}-\texttt{l}(e))\Bigg) \\
&\quad\cdot\Bigg(\sum_{\substack{x,z\in\texttt{below}(t.l),w\in\texttt{above}(t)\\ L(x)=X,L(z)\neq L(w)\neq X,Y}}\texttt{I}(x,z,w)\cdot\Delta\texttt{W}(x,z|t)\cdot\texttt{exp}(\sum_{e\in w\to t}-\texttt{l}(e))\Bigg) \\
=&\texttt{exp}(-\texttt{l}(t.r))\cdot\texttt{w}_Y^1(t.r)\cdot(1-\texttt{s}(t.l)) \\
&\quad\cdot\Bigg(\sum_{\substack{x,z\in\texttt{below}(t.l),w\in\texttt{above}(t)\\ L(x)=X,L(z)\neq L(w)\neq X,Y}}\texttt{I}(x,z,w)\cdot\Delta\texttt{W}(x,z|t.l)\cdot\texttt{exp}(\sum_{e\in w\to t}-\texttt{l}(e))\Bigg)
\end{aligned}
$$

where

$$
\begin{aligned}
&\sum_{\substack{x,z\in\texttt{below}(t.l),w\in\texttt{above}(t)\\ L(x)=X,L(z)\neq L(w)\neq X,Y}} \texttt{I}(x,z,w)\cdot\Delta\texttt{W}(x,z|t.l)\cdot\texttt{exp}(\sum_{e\in w\to t}-\texttt{l}(e))\\
&=\sum_{\substack{Z\in\mathscr{S}\cup\mathscr{A}\\ Z\neq X,Y}}\sum_{\substack{W\in\mathscr{S}\cup\mathscr{A}\\ W\neq X,Y,Z}}\sum_{\substack{x,z\in\texttt{below}(t.l)\\ L(x)=X,L(z)=Z}}\sum_{\substack{w\in\texttt{above}(t)\\ L(w)=W}}\texttt{I}(x,z,w)\cdot\Delta\texttt{W}(x,z|t.l)\cdot\texttt{exp}(\sum_{e\in w\to t}-\texttt{l}(e))\\
&=\sum_{\substack{Z\in\mathscr{S}\cup\mathscr{A}\\ Z\neq X,Y}}\sum_{\substack{W\in\mathscr{S}\cup\mathscr{A}\\ W\neq X,Y,Z}}\sum_{\substack{x,z\in\texttt{below}(t.l)\\ L(x)=X,L(z)=Z}}\sum_{\substack{w\in\texttt{above}(t)\\ L(w)=W}}\left(\texttt{I}(x,z)\cdot\texttt{exp}(\sum_{e\in x,z\to u}-\texttt{l}(e))\cdot(\prod_{e\in u\to t.l}(1-\texttt{s}(e)))\right)\\
&\quad\cdot\left(\texttt{I}(w)\cdot\texttt{exp}(\sum_{e\in w\to t}-\texttt{l}(e))\right)\\
&=\sum_{\substack{Z\in\mathscr{S}\cup\mathscr{A}\\ Z\neq X,Y}}\sum_{\substack{W\in\mathscr{S}\cup\mathscr{A}\\ W\neq X,Y,Z}}\left(\sum_{\substack{x,z\in\texttt{below}(t.l)\\ L(x)=X,L(z)=Z}}\texttt{I}(x,z)\cdot\texttt{exp}(\sum_{e\in x,z\to u}-\texttt{l}(e))\cdot(\prod_{e\in u\to t.l}(1-\texttt{s}(e)))\right)\\
&\quad\cdot\left(\sum_{\substack{w\in\texttt{above}(t)\\ L(w)=W}}\texttt{I}(w)\cdot\texttt{exp}(\sum_{e\in w\to t}-\texttt{l}(e))\right)\\
&=\sum_{\substack{Z\in\mathscr{S}\cup\mathscr{A}\\ Z\neq X,Y}}\sum_{\substack{W\in\mathscr{S}\cup\mathscr{A}\\ W\neq X,Y,Z}}\texttt{w}^2_{X,Z}(t.l)\cdot\overline{\texttt{w}}^1_W(t)
\end{aligned}
$$

where $\overline{\texttt{w}}^1_W(t)$ is the total weight of singlets above t (which can be computed via the recurrence in Lemma B.2) and $\texttt{w}^2_{X,Z}(t.l)$ is the total weight of doublets below $t.l$ (which can be computed via the recurrence in Lemma B.3). As a result,

$$\Delta\mathbb{G}^{(c)}_t(X,Y)=\texttt{exp}(-\texttt{l}(t.r))\cdot\texttt{w}^1_Y(t.r)\cdot(1-\texttt{s}(t.l))\cdot\sum_{\substack{Z\in\mathscr{S}\cup\mathscr{A}\\ Z\neq X,Y}}\sum_{\substack{W\in\mathscr{S}\cup\mathscr{A}\\ W\neq X,Y,Z}}\texttt{w}^2_{X,Z}(t.l)\cdot\overline{\texttt{w}}^1_W(t)$$

Lastly, in **case (f)**, x and z are from the left subtree while y and w are from the right subtree. In other words, we select a doublet (x,z) from $v.l$ and a doublet (y,w) from $v.r$ to form the quartet. As a result, we compute the total weight by multiplying the doublet weight sums in the subtrees while excluding the weights when $L(z)=L(w)$. Case (f) is special because the topology of the quartet

must be $x,z|y,w$ so we have

$$
\begin{aligned}
&\Delta\mathbb{G}_t^{(f)}(X,Y)\\
=&\sum_{\substack{x,z\in\texttt{below}(t.l),y,w\in\texttt{below}(t.r)\\ L(x)=X,L(y)=Y,L(z)\neq L(w)\neq X,Y\\ q(x,y,z,w)=x,z|y,w}} \Delta\mathtt{W}(q)\cdot\mathtt{I}(q)\\
=&\sum_{\substack{x,z\in\texttt{below}(t.l),y,w\in\texttt{below}(t.r)\\ L(x)=X,L(y)=Y,L(z)\neq L(w)\neq X,Y}} \exp\Bigg(\sum_{\substack{e\in x,z\to u\\ y,w\to v}}-\mathtt{l}(e)\Bigg)\cdot\Bigg(\prod_{e\in u\to v}(1-\mathtt{s}(e))\Bigg)\cdot\mathtt{I}(x,y,z,w)\\
=&\sum_{\substack{x,z\in\texttt{below}(t.l),y,w\in\texttt{below}(t.r)\\ L(x)=X,L(y)=Y,L(z)\neq L(w)\neq X,Y}} \Bigg((1-\mathtt{s}(t.l))\cdot\mathtt{I}(x,z)\cdot\exp(\sum_{e\in x,z\to u}-\mathtt{l}(e))\cdot\prod_{e\in u\to t.l}(1-\mathtt{s}(e))\Bigg)\\
&\cdot\Bigg((1-\mathtt{s}(t.r))\cdot\mathtt{I}(y,w)\cdot\exp(\sum_{e\in y,w\to v}-\mathtt{l}(e))\cdot\prod_{e\in v\to t.r}(1-\mathtt{s}(e))\Bigg)\\
=&(1-\mathtt{s}(t.l))\cdot(1-\mathtt{s}(t.r))\cdot\Bigg(\sum_{\substack{Z\in\mathscr{S}\cup\mathscr{A}\\ Z\neq X,Y}}\sum_{\substack{W\in\mathscr{S}\cup\mathscr{A}\\ W\neq X,Y,Z}}\mathtt{w}^2_{X,Z}(t.l)\cdot\mathtt{w}^2_{Y,Z}(t.r)\Bigg).
\end{aligned}
$$

where $\mathtt{w}^2_{X,Z}(t.l)$ counts the total weight of doublets below $t.l$ (which can be computed via the recurrence in Lemma B.3). Adding up weights from all five cases

$$\Delta\mathbb{G}_t(X,Y)=\sum_{\gamma\in\{a,b,d,e,f\}}\Delta\mathbb{G}_t^{(\gamma)}(X,Y)$$

gives us the final recurrence. □

Theorem B.3. *After precomputation of the six auxiliary values, $\Delta\mathbb{B}(X,Y)$ and $\Delta\mathbb{G}(X,Y)$ can be computed for all pairs of taxa $X,Y\in\mathscr{S}\cup\mathscr{A}$ in $O((a+b)^4n)$ time and $O((a+b)^2$ space.*

Proof. Bad and good edges between taxa X,Y can be computed as

$$\Delta\mathbb{B}(X,Y)=\sum_{t\in V(T)}\Delta\mathbb{B}_t(X,Y)\text{ and }\Delta\mathbb{G}(X,Y)=\sum_{t\in V(T)}\Delta\mathbb{G}_t(X,Y)\tag{B.23}$$

where $\Delta\mathbb{B}_t(X,Y)$ and $\Delta\mathbb{G}_t(X,Y)$ are each solved in $O((a+b)^2)$ time with Equation B.21 (The-

orem B.1) and Equation B.22 (Theorem B.2), respectively, after computing and saving the six auxiliary values. Repeating this computation for all pairs of taxa $X, Y \in \mathscr{S} \cup \mathscr{A}$ gives the time complexity result. We do not need to store $\Delta\mathbb{B}_t(X,Y)$, as it can be added to $\Delta\mathbb{B}(X,Y)$, which gives us the storage complexity result. □

B.3.3 Time and Space Efficient Algorithms

We just introduced algorithms to compute $\Delta\mathbb{B}(X,Y)$ and $\Delta\mathbb{G}(X,Y)$. The algorithms relies heavily on precomputation of auxiliary values, which treats all pairs of taxa X, Y in the same way regardless of whether they are singletons or artificial taxa. This is undesirable because we conjecture that the number a of singletons will much greater than the number b of artificial taxa in practice, and as we will show, singleton taxa can be treated more efficiently than artificial taxa. In the remainder of this section, we introduce time and space efficient algorithms, picking up from Section "Efficient Algorithm for Computing Weighted Bad Edges and Good for Subproblem" in Appendix B in the main text.

Aggregation of singlets and doublets across all singleton taxa

In this section, we pick up from paragraph, "Reducing the time complexity of triplet auxiliary values by aggregating singletons," in the main text. We begin by aggregating the singlet and doublet auxiliary values across all singleton taxa, introducing special taxon label `1` to represent all singletons. We first consider extending the definition of singlet auxiliary values $\mathtt{w}^1_X(t)$ (Definition B.3) and $\overline{\mathtt{w}}^1_X(t)$ (Definition B.4) in the natural way.

Definition B.9 (singlet auxiliary values for aggregated singletons)**.** $\mathtt{w}^1_1(t)$ *is the sum of weight from* singletons *in the subtree* below *vertex* t *in gene tree* T*. Similarly,* $\overline{\mathtt{w}}^1_1(t)$ *is the sum of weight from* singletons *in the subtree above* t*. More precisely, the singlet weights of singletons* below *and* above

t are defined as

$$\mathtt{w}_1^1(t) = \sum_{\substack{x \in \mathtt{below}(t) \\ L(x) \in \mathscr{S}}} \mathtt{I}(x) \cdot \mathtt{exp}\Big(\sum_{e \in x \to t} -\mathtt{l}(e) \Big) \text{ and} \tag{B.24}$$

$$\overline{\mathtt{w}}_1^1(t) = \sum_{\substack{x \in \mathtt{above}(t) \\ L(x) \in \mathscr{S}}} \mathtt{I}(x) \cdot \mathtt{exp}\Big(\sum_{e \in x \to t} -\mathtt{l}(e) \Big), \text{ respectively.} \tag{B.25}$$

Corollary B.1 (singlet auxiliary values for aggregated singletons)**.** $\mathtt{w}_1^1(t)$ *and* $\overline{\mathtt{w}}_1^1(t)$ *can be computed according to the following recurrences:*

$$\mathtt{w}_1^1(t) = \begin{cases} \mathtt{I}(t) & \text{if } t \text{ is a leaf in } T \\ \mathtt{exp}(-\mathtt{l}(t.l)) \cdot \mathtt{w}_1^1(t.l) + \mathtt{exp}(-\mathtt{l}(t.r)) \cdot \mathtt{w}_1^1(t.r) & \text{otherwise} \end{cases} \tag{B.26}$$

$$\overline{\mathtt{w}}_1^1(t) = \begin{cases} 0 & \text{if } t \text{ is the root of } T \\ \mathtt{exp}(-\mathtt{l}(t)) \cdot \overline{\mathtt{w}}_1^1(t.p) + \mathtt{exp}(-\mathtt{l}(t.s)) \cdot \mathtt{w}_1^1(t.s) & \text{otherwise} \end{cases} \tag{B.27}$$

It takes $O(n)$ *time to compute* $\mathtt{w}_1^1(t)$ *for all vertices* $t \in V(T)$*. Likewise, it takes* $O(n)$ *space to access the aggregated singlet auxiliary values later. The time and storage complexity are the same for* $\overline{\mathtt{w}}_X^1(t)$*, after computing* $\mathtt{w}_1^1(t)$*.*

Proof. Equation B.26 and Equation B.27 follow from Equation B.9 (Lemma B.1) and Equation B.11 (Lemma B.2). The "below" singlet is computed by traversing the $O(n)$ vertices of T in a postorder fashion; after which the "above" singlet is computed by traversing the $O(n)$ vertices of T in a preorder fashion. Evaluating the two recurrences for vertex t takes $O(1)$ time and then storing the two auxiliary values requires $O(1)$ space. Putting this together gives the time and storage complexity. □

We continue in this fashion for doublet auxiliary values $\mathtt{w}_{X,Y}^2(t)$ and $\overline{\mathtt{w}}_{X,Y}^2(t)$, although now we could replace one or both of X and Y with the special taxon label 1. First, we consider the

aggregating doublets when both X and Y are singletons.

Definition B.10 (doublet auxiliary values for two aggregated singletons)**.** $\mathtt{w}^2_{1,1}(t)$ *is the sum of the weight from leaf pairs x and y in the subtree* below *t in gene tree T, where x and y are labeled by singletons. Similarly,* $\overline{\mathtt{w}}^2_{1,1}(t)$ *is the sum of weight from leaf pairs in the subtree* above *t. The doublet weights of pairs of singletons* below *and* above *t are defined as*

$$\mathtt{w}^2_{1,1}(t) = \sum_{\substack{x,y\in\mathtt{below}(t),x\neq y\\ L(x),L(y)\in\mathscr{S}}} \mathtt{I}(x,y)\cdot\mathtt{exp}\Big(\sum_{e\in x,y\to u} -\mathtt{l}(e)\Big)\cdot\prod_{e\in u\to t}(1-\mathtt{s}(e)) \text{ and} \tag{B.28}$$

$$\overline{\mathtt{w}}^2_{1,1}(t) = \sum_{\substack{x,y\in\mathtt{above}(t),x\neq y\\ L(x),L(y)\in\mathscr{S}}} \mathtt{I}(x,y)\cdot\mathtt{exp}\Big(\sum_{e\in x,y\to u} -\mathtt{l}(e)\Big)\cdot\prod_{e\in u\to t}(1-\mathtt{s}(e)), \tag{B.29}$$

respectively, where u is the lowest common ancestor of x and y for $\mathtt{w}^2_{1,1}$ *and u is* $lca(x,y)$*,* $lca(x,t)$*, or* $lca(y,t)$ *for* $\overline{\mathtt{w}}^2_{1,1}$ *depending on which is farthest from the root of T.*

Corollary B.2 (doublet auxiliary values for two aggregated singletons)**.** $\mathtt{w}^2_{1,1}(t)$ *and* $\overline{\mathtt{w}}^2_{1,1}(t)$ *can be computed according to the following recurrences:*

$$\mathtt{w}^2_{1,1}(t) = \begin{cases} 0 & \text{if } t \text{ is a leaf of } T \\ (1-\mathtt{s}(t.l))\cdot\mathtt{w}^2_{1,1}(t.l) + (1-\mathtt{s}(t.r))\cdot\mathtt{w}^2_{1,1}(t.r) & \\ +\mathtt{exp}(-\mathtt{l}(t.l))\cdot\mathtt{w}^1_1(t.l)\cdot\mathtt{exp}(-\mathtt{l}(t.r))\cdot\mathtt{w}^1_1(t.r) & \text{otherwise} \end{cases} \tag{B.30}$$

$$\overline{\mathtt{w}}^2_{1,1}(t) = \begin{cases} 0 & \text{if } t \text{ is the root of } T \\ (1-\mathtt{s}(t))\cdot\overline{\mathtt{w}}^2_{1,1}(t.p) + (1-\mathtt{s}(t))\cdot(1-\mathtt{s}(t.s))\cdot\mathtt{w}^2_{1,1}(t.s) & \\ +(1-\mathtt{s}(t))\cdot\mathtt{exp}(-\mathtt{l}(t.s))\cdot\overline{\mathtt{w}}^1_1(t.p)\cdot\mathtt{w}^1_1(t.s) & \text{otherwise} \end{cases} \tag{B.31}$$

It takes $O(n)$ *time to compute* $\mathtt{w}^2_{1,1}(t)$ *for all vertices* $t\in V(T)$*. Likewise, it takes* $O(n)$ *space to access the aggregated doublet auxiliary values later. The time and storage complexity is the same*

for $\overline{\mathtt{w}}^2_{1,1}(t)$*, after computing* $\mathtt{w}^1_1$ *(Corollary B.1).*

Proof. Equation B.30 and Equation B.31 are the similar as Equation B.14 (Lemma B.3) and Equation B.16 (Lemma B.16), respectively, but now instead of summing over all ways of picking two different leaves x, y labeled by X, Y, we now sum over all ways of picking two different leaves labeled by singletons. This requires some small modifications to avoid double counting singleton taxa. The "below" doublet is computed by traversing the $O(n)$ vertices of T in a postorder fashion; after which the "above" doublet is computed by traversing the $O(n)$ vertices of T in a preorder fashion. Evaluating the two recurrences for vertex t takes $O(1)$ time and then storing the two auxiliary values requires $O(1)$ space. Putting this together gives the time and storage complexity. □

Second, we consider the aggregating doublets when Y is a singleton but not X.

Definition B.11 (doublet auxiliary values for one aggregated singletons)**.** $\mathtt{w}^2_{X,1}(t) = \mathtt{w}^2_{1,X}(t)$ *is the sum of the weight from leaf pairs in the subtree* below *t in gene tree T, where one leaf is labeled by a singleton and the other is labeled by an artificial taxon X. Similarly,* $\overline{\mathtt{w}}^2_{X,1}(t) = \overline{\mathtt{w}}^2_{1,X}(t)$ *is the sum of weight from leaf pairs in the subtree* above *t. The doublet weights of pairs of singletons* below *and* above *t are defined as*

$$\mathtt{w}^2_{X,1}(t) = \mathtt{w}^2_{1,X}(t) = \sum_{\substack{x,y \in \mathtt{below}(t) \\ L(x)=X, L(y)\in\mathscr{S}}} \mathtt{I}(x,y) \cdot \mathtt{exp}\Big(\sum_{e\in x,y\to u} -\mathtt{l}(e)\Big) \cdot \prod_{e\in u\to t} (1-\mathtt{s}(e)) \text{ and} \tag{B.32}$$

$$\overline{\mathtt{w}}^2_{X,1}(t) = \overline{\mathtt{w}}^2_{1,X}(t) = \sum_{\substack{x,y \in \mathtt{above}(t) \\ L(x)=X, L(y)\in\mathscr{S}}} \mathtt{I}(x,y) \cdot \mathtt{exp}\Big(\sum_{e\in x,y\to u} -\mathtt{l}(e)\Big) \cdot \prod_{e\in u\to t} (1-\mathtt{s}(e)), \tag{B.33}$$

respectively, where u is the lowest common ancestor of x and y for $\mathtt{w}^2_{X,1}$ *and u is $lca(x,y)$, $lca(x,t)$, or $lca(y,t)$ for* $\overline{\mathtt{w}}^2_{X,1}$ *depending on which is farthest from the root of T.*

Corollary B.3 (doublet auxiliary values for one aggregated singletons)**.** $\mathtt{w}^2_{X,1}(t)$ *and* $\overline{\mathtt{w}}^2_{X,1}(t)$ *can be*

computed according to the following recurrences:

$$\mathtt{w}^2_{X,1}(t) = \mathtt{w}^2_{1,X}(t) = \begin{cases} 0 & \textit{if } t \textit{ is a leaf} \\ (1-\mathtt{s}(t.l)) \cdot \mathtt{w}^2_{X,1}(t.l) + (1-\mathtt{s}(t.r)) \cdot \mathtt{w}^2_{X,1}(t.r) & \\ +\mathtt{exp}(-\mathtt{l}(t.l)) \cdot \mathtt{w}^1_X(t.l) \cdot \mathtt{exp}(-\mathtt{l}(t.r)) \cdot \mathtt{w}^1_1(t.r) & \\ +\mathtt{exp}(-\mathtt{l}(t.l)) \cdot \mathtt{w}^1_1(t.l) \cdot \mathtt{exp}(-\mathtt{l}(t.r)) \cdot \mathtt{w}^1_X(t.r) & \textit{otherwise} \end{cases} \tag{B.34}$$

$$\overline{\mathtt{w}}^2_{X,1}(t) = \overline{\mathtt{w}}^2_{1,X}(t) = \begin{cases} 0 & \textit{if } t \textit{ is the root} \\ (1-\mathtt{s}(t)) \cdot \overline{\mathtt{w}}^2_{X,1}(t.p) & \\ +(1-\mathtt{s}(t)) \cdot (1-\mathtt{s}(t.s)) \cdot \mathtt{w}^2_{X,1}(t.s) & \\ +(1-\mathtt{s}(t)) \cdot \mathtt{exp}(-\mathtt{l}(t.s)) \cdot \overline{\mathtt{w}}^1_X(t.p) \cdot \mathtt{w}^1_1(t.s) & \\ +(1-\mathtt{s}(t)) \cdot \mathtt{exp}(-\mathtt{l}(t.s)) \cdot \overline{\mathtt{w}}^1_X(t.s) \cdot \mathtt{w}^1_1(t.p) & \textit{otherwise} \end{cases} \tag{B.35}$$

After precomputation, it takes $O(bn)$ time to compute $\mathtt{w}^2_{X,1}(t)$ and $\overline{\mathtt{w}}^2_{X,1}(t)$ for all vertices $t \in V(T)$ and for all taxa $X \in \mathscr{A}$. Likewise, it takes $O(bn)$ space to access the aggregated doublet auxiliary values later.

Proof. Equation B.34 and Equation B.35 follow from Equation B.14 (Lemma B.3) and Equation B.16 (Lemma B.16). The "below" doublet is computed by traversing the $O(n)$ vertices of T in a postorder fashion; after which the "above" doublet is computed by traversing the $O(n)$ vertices of T in a preorder fashion. Evaluating the two recurrences for vertex t and artificial taxon X takes $O(1)$ time and then storing the two auxiliary values requires $O(1)$ space. Putting this together gives the time and storage complexity. □

Efficient algorithms for bad triplets based on aggregated singletons Computing the auxiliary values for bad triplets is more complicated than singlets or doublets because the triplet recurrences enumerate all ways of selecting two other taxa Z,W such that $Z \neq W \neq X \neq Y$. This leads to a

quadratic term in the evaluation of the recurrences. To count the weights of bad triplets $\mathtt{w}_{X,Y}^{x|z,w}(t)$ via the recurrence in Equation B.18 (Lemma B.5), we need to evaluate the term

$$\sum_{\substack{Z,W\in\mathscr{S}\cup\mathscr{A}\\ Z\neq W\neq X\neq Y}} \mathtt{w}_{Z,W}^2(\cdot)$$

at both the left and right child of t. We speed up the computation for bad triplets by leveraging the doublet auxiliary quantities aggregated across singletons.

Lemma B.7. *The quantity below can be computed in $O(b^2)$ time with the following equation:*

$$\sum_{\substack{Z,W\in\mathscr{S}\cup\mathscr{A}\\ Z\neq W\neq X,Y}} \mathtt{w}_{Z,W}^2(t) = \mathtt{w}_{1,1}^2(t) + \sum_{\substack{W\in\mathscr{A}\\ W\neq X,Y}} \mathtt{w}_{1,W}^2(t) + \sum_{\substack{Z,W\in\mathscr{A}\\ Z\neq W\neq X,Y}} \mathtt{w}_{Z,W}^2(t)$$

after computing the doublet auxiliary values.

Proof. We expand the computation of $\sum_{\substack{Z,W\in\mathscr{S}\cup\mathscr{A}\\ Z\neq W\neq X,Y}} \mathtt{w}_{Z,W}^2(t)$ into three cases: (1) both Z and W are both singleton taxa (i.e., $Z,W\in\mathscr{S}$), (2) one of Z and W is an artificial taxon and the other is a singleton taxon (w.l.o.g. $Z\in\mathscr{S}$ and $W\in\mathscr{A}$), and (3) Z and W are both artificial taxa (i.e., $Z,W\in\mathscr{A}$), where leaves labeled Z and W come from the same subtree by definition of $\mathtt{w}_{Z,W}^2(t)$. In case (1), both Z and W are singletons so the total weight of the doublets is $\mathtt{w}_{1,1}(t)$. In case (2), Z is a singleton but W is an artificial taxon, so the total weight is the sum of the $\mathtt{w}_{1,W}^2(t)$ for all W in $\mathscr{A}$. In case (3), both Z and W are artificial so we still enumerate all unordered pairs of artificial

taxa. Adding the weights of three cases together, we have

$$\begin{aligned}\sum_{\substack{Z,W\in\mathscr{S}\cup\mathscr{A}\\ Z\neq W\neq X,Y}} \mathtt{w}^2_{Z,W}(t) &= \sum_{\substack{Z,W\in\mathscr{S}\\ Z\neq W\neq X,Y}} \mathtt{w}^2_{Z,W}(t) + \sum_{\substack{Z\in\mathscr{S},W\in\mathscr{A}\\ Z\neq W\neq X,Y}} \mathtt{w}^2_{Z,W}(t) + \sum_{\substack{Z,W\in\mathscr{A}\\ Z\neq W\neq X,Y}} \mathtt{w}^2_{Z,W}(t)\\ &= \sum_{\substack{Z,W\in\mathscr{S}\\ Z\neq W\neq X,Y}} \mathtt{w}^2_{Z,W}(t) + \sum_{\substack{W\in\mathscr{A}\\ W\neq X,Y}}\sum_{\substack{Z\in\mathscr{S}\\ Z\neq X,Y}} \mathtt{w}^2_{Z,W}(t) + \sum_{\substack{Z,W\in\mathscr{A}\\ Z\neq W\neq X,Y}} \mathtt{w}^2_{Z,W}(t)\\ &= \mathtt{w}^2_{1,1}(t) + \sum_{\substack{W\in\mathscr{A}\\ W\neq X,Y}} \mathtt{w}^2_{1,W}(t) + \sum_{\substack{Z,W\in\mathscr{A}\\ Z\neq W\neq X,Y}} \mathtt{w}^2_{Z,W}(t)\end{aligned}$$

After precomputation, the time complexity of the first, second, and third terms is $O(1)$, $O(b)$, and $O(b^2)$, giving us the time complexity result. □

The proof above for $\mathtt{w}^2_{Z,W}(t)$ works for other function evaluations related to bad edges. We therefore define a function $\mathbb{P}^{\mathbb{B}}_{X,Y}[\mathtt{F}_{Z,Y}](t)$ that computes these nested summations efficiently, for example setting $\mathtt{F}_{Z,W} = \mathtt{w}^2_{Z,W}$ (which is evaluated at vertex t). This gives us the corollary below.

Corollary B.4. *Let* $\mathbf{P}^{\mathbb{B}}_{X,Y}[\mathtt{F}_{Z,W}](t)$ *be* $\sum_{\substack{Z,W\in\mathscr{S}\cup\mathscr{A}\\ Z\neq W\neq X,Y}} \mathtt{F}_{Z,W}(t)$*, where* $\mathtt{F}_{Z,W}$ *is a mapping from a 3-tuple of two artificial taxa and a tree vertex to a weight value, i.e.,*

$$\mathtt{F}_{Z,W} : (\mathscr{S}\cup\mathscr{A}) \times (\mathscr{S}\cup\mathscr{A}) \times V(T) \to \mathbb{R},$$

which could be $\mathtt{w}^2_{Z,W}(t)$ *or* $\overline{\mathtt{w}}^2_{Z,W}(t)$ *in our algorithms. Then,* $\mathbf{P}^{\mathbb{B}}_{X,Y}[\mathtt{F}_{Z,W}](t)$ *can be computed in* $O(b^2)$ *time using the following equation:*

$$\mathbf{P}^{\mathbb{B}}_{X,Y}[\mathtt{F}_{Z,W}](t) = \sum_{\substack{Z,W\in\mathscr{S}\cup\mathscr{A}\\ Z\neq W\neq X,Y}} \mathtt{F}_{Z,W}(t) = \mathtt{F}_{1,1}(t) + \sum_{\substack{W\in\mathscr{A}\\ W\neq X,Y}} \mathtt{F}_{1,W}(t) + \sum_{\substack{Z,W\in\mathscr{A}\\ Z\neq W\neq X,Y}} \mathtt{F}_{Z,W}(t) \tag{B.36}$$

after computing the relevant auxiliary values.

Using Corollary B.4, we can reduce the cost of computing a bad triplet for X,Y below t from

$O((a+b)^2)$ to $O(b^2)$ by computing auxiliary values and aggregating the computation from singletons. The bottleneck now is evaluating the term enumerating the pairs of artificial taxa Z and W in the subtree t, i.e., $\mathbf{P}^{\mathbb{B}}_{X,Y}[\mathtt{F}_{Z,W}](t)$, which gives us the $O(b^2)$ term. In the following, we show how to reduce the time cost to $O(b)$.

Lemma B.8. $\mathbf{P}^{\mathbb{B}}_{X,Y}[\mathtt{w}^2_{Z,W}](t)$ *can be computed in* $O(b)$ *time, after computing the doublet auxiliary values.*

Proof. According to Lemma B.7, we have

$$\mathbf{P}^{\mathbb{B}}_{X,Y}[\mathtt{w}^2_{Z,W}](t) = \sum_{\substack{Z,W \in \mathscr{S} \cup \mathscr{A} \\ Z \neq W \neq X,Y}} \mathtt{w}^2_Z(t) = \mathtt{w}^2_{1,1}(t) + \sum_{\substack{W \in \mathscr{A} \\ W \neq X,Y}} \mathtt{w}^2_{1,W}(t) + \sum_{\substack{Z,W \in \mathscr{A} \\ Z \neq W \neq X,Y}} \mathtt{w}^2_{Z,W}(t)$$

where the first two terms $\mathtt{w}^2_{1,1}(t) + \sum_{\substack{W \in \mathscr{A} \\ W \neq X,Y}} \mathtt{w}^2_{1,W}(t)$ can be computed in $O(b)$ time. We rewrite the third term as

$$\sum_{\substack{Z,W \in \mathscr{A} \\ Z \neq W \neq X,Y}} \mathtt{w}^2_{Z,W}(t) = \sum_{Z,W \in \mathscr{A}, Z \neq W} \mathtt{w}^2_{Z,W}(t) - \sum_{Z \in \mathscr{A}} \mathtt{w}^2_{Z,X}(t) - \sum_{Z \in \mathscr{A}} \mathtt{w}^2_{Z,Y}(t) + \mathtt{w}^2_{X,Y}(t), \tag{B.37}$$

The time complexity of the first, second, third, and fourth terms is $O(b^2)$, $O(b)$, $O(b)$, and $O(1)$, after precomputation. However, the first term does not depend on the selection of X, Y; therefore, we can compute this quantity during the precomputation phase after doublet auxiliary values but before bad triplet auxiliary values. The precomputation of the first term takes $O(b^2 n)$ time and space because there are $O(b^2)$ pairs of Z, W of artificial taxa and $O(n)$ vertices t. Thus, it does not exceed the time and space complexity from precomputation of doublet auxiliary values even when only computing them for pairs of artificial taxa (Lemma B.3).

Lastly, we do not have to exclude X and/or Y if they singletons. This simplifies the computation

of the third term in $\mathbf{P}^{\mathbb{B}}_{X,Y}[\mathtt{w}^2_{Z,W}](t)$ to

$$\sum_{\substack{Z,W\in\mathscr{A}\\ Z\neq W\neq X,Y}} \mathtt{w}^2_{Z,W}(t) = \sum_{Z,W\in\mathscr{A},Z\neq W} \mathtt{w}^2_{Z,W}(t) - \sum_{Z\in\mathscr{A}} \mathtt{w}^2_{Z,Y}(t) \tag{B.38}$$

if X but not Y is a singleton,

$$\sum_{\substack{Z,W\in\mathscr{A}\\ Z\neq W\neq X,Y}} \mathtt{w}^2_{Z,W}(t) = \sum_{Z,W\in\mathscr{A},Z\neq W} \mathtt{w}^2_{Z,W}(t) - \sum_{Z\in\mathscr{A}} \mathtt{w}^2_{Z,X}(t) \tag{B.39}$$

if Y but not X is a singleton, and

$$\sum_{\substack{Z,W\in\mathscr{A}\\ Z\neq W\neq X,Y}} \mathtt{w}^2_{Z,W}(t) = \sum_{Z,W\in\mathscr{A},Z\neq W} \mathtt{w}^2_{Z,W}(t) \tag{B.40}$$

if both X and Y are singletons. It does not change the other computations because if X and/or Y are singletons we do not need to include them when drawing taxa from the set $\mathscr{A}$. □

Once again, the proof above for $\mathtt{w}^2_{Z,W}(t)$ works for other function evaluations related to bad edges. This gives us the following corollary.

Corollary B.5. $\mathbf{P}^{\mathbb{B}}_{X,Y}[\mathtt{F}_{Z,W}](t)$ *can be computed in* $O(b)$ *time, after computing the relevant auxiliary values.*

Putting this all together enables us to reduce the time complexity of computing the bad triplet auxiliary values $\mathtt{w}^{x|z,w}_{X,Y}(t)$.

Corollary B.6 (efficient bad triplet auxiliary values)**.** $\mathtt{w}^{x|z,w}_{X,Y}(t)$ *can be computed according to the*

following recurrence

$$\mathtt{w}_{X,Y}^{x|z,w}(t) = \begin{cases} 0 & \text{if } t \text{ is a leaf in } T \\ \exp(-\mathtt{l}(t.l)) \cdot \mathtt{w}_{X,Y}^{x|z,w}(t.l) + \exp(-\mathtt{l}(t.r)) \cdot \mathtt{w}_{X,Y}^{x|z,w}(t.r) & \\ +\exp(-\mathtt{l}(t.l)) \cdot \mathtt{w}_{X}^{1}(t.l) \cdot (1-\mathtt{s}(t.r)) \cdot \mathbf{P}_{X,Y}^{\mathbb{B}}[\mathtt{w}_{Z,W}^{2}](t.r) & \\ +\exp(-\mathtt{l}(t.r)) \cdot \mathtt{w}_{X}^{1}(t.r) \cdot (1-\mathtt{s}(t.l)) \cdot \mathbf{P}_{X,Y}^{\mathbb{B}}[\mathtt{w}_{Z,W}^{2}](t.l) & \text{otherwise} \end{cases} \tag{B.41}$$

After computing singlet and doublet auxiliary values, it takes $O((a+b)^2bn)$ time to compute $\mathtt{w}_{X,Y}^{x|z,w}(t)$ for all vertices $t \in V(T)$ and for all unordered pairs of taxa $X,Y \in \mathscr{S} \cup \mathscr{A}$. Likewise, it takes $O((a+b)^2n)$ space to access the bad triplet auxiliary values later. The time and storage complexity drops to $O(b^3n)$ and $O(b^2n)$, respectively when restricting X to be an artificial taxa (i.e., $X \in \mathscr{A}$ and $Y \in \mathscr{A} \cup \{0\}$).

Proof. This correctness of the recurrence follows from Lemma B.5 and Lemma B.8. After precomputation, $\mathtt{w}_{X,Y}^{x|z,w}(t)$ can be computed by traversing the $O(n)$ vertices of gene tree T in a postorder traversal. Evaluating the recurrence for vertex t and pair of taxa X,Y takes $O(b)$ time by Lemma B.8, and storing the resulting auxiliary value requires $O(1)$ space. Repeating for all $O((a+b)^2)$ unordered pairs of taxa $X,Y \in \mathscr{S} \cup \mathscr{A}$ gives the time and storage complexity. □

Efficient algorithms for good triplets based on aggregated singletons

The recurrence for good triplets also enumerates pairs of taxa, i.e., $\sum_{\substack{Z,W \in \mathscr{S} \cup \mathscr{A} \\ Z \neq W \neq X \neq Y}} \cdots$ and $\sum_{\substack{Z \in \mathscr{S} \cup \mathscr{A} \\ Z \neq X,Y}} \sum_{\substack{W \in \mathscr{S} \cup \mathscr{A} \\ W \neq X,Y,Z}} \cdots$ but with nested summations. We speed up the computation for good triplets by leveraging the doublet auxiliary quantities aggregated across singletons.

Lemma B.9. *The quantity below can be computed in $O(b^2)$ time with the following equation:*

$$\sum_{\substack{Z\in\mathscr{S}\cup\mathscr{A}\\ Z\neq X,Y}}\sum_{\substack{W\in\mathscr{S}\cup\mathscr{A}\\ W\neq X,Y,Z}} \mathtt{w}^2_{X,Z}(t.l)\cdot\mathtt{w}^1_W(t.r) = \mathtt{w}^2_{X,1}(t.l)\cdot\mathtt{w}^1_1(t.r) + \sum_{\substack{W\in\mathscr{A}\\ W\neq X,Y}} \mathtt{w}^2_{X,1}(t.l)\cdot\mathtt{w}^1_W(t.r)$$
$$+ \sum_{\substack{Z\in\mathscr{A}\\ Z\neq X,Y}} \mathtt{w}^2_{X,Z}(t.l)\cdot\mathtt{w}^1_1(t.r) + \sum_{\substack{Z\in\mathscr{A}\\ Z\neq X,Y}}\sum_{\substack{W\in\mathscr{A}\\ W\neq X,Y,Z}} \mathtt{w}^2_{X,Z}(t.l)\cdot\mathtt{w}^1_W(t.r)$$

after computing the singlet and doublet auxiliary values.

Proof. We expand the computation of $\sum_{\substack{Z\in\mathscr{S}\cup\mathscr{A}\\ Z\neq X,Y}}\sum_{\substack{W\in\mathscr{S}\cup\mathscr{A}\\ W\neq X,Y,Z}} \mathtt{w}^2_{X,Z}(t.l)\cdot\mathtt{w}^1_W(t.r)$, where Z into four cases: (1) $Z,W\in\mathscr{A}$, (2) $Z\in\mathscr{S}$ and $W\in\mathscr{A}$, (3) $Z\in\mathscr{A}$ and $W\in\mathscr{S}$, and (4) $Z,W\in\mathscr{S}$, where leaves labeled Z and W are from different subtrees by definition of $\mathtt{w}^1_W(t.r)$ and $\mathtt{w}^2_{X,Z}(t.l)$. Thus, cases (2) and (3) are not equivalent unlike the case of bad triplets (but they are similar due to symmetry). Breaking the computation into these cases gives us the following

$$\begin{aligned}
&\sum_{\substack{Z\in\mathscr{S}\cup\mathscr{A}\\ Z\neq X,Y}}\sum_{\substack{W\in\mathscr{S}\cup\mathscr{A}\\ W\neq X,Y,Z}} \mathtt{w}^2_{X,Z}(t.l)\cdot\mathtt{w}^1_W(t.r)\\
=&\sum_{\substack{Z\in\mathscr{S}\\ Z\neq X,Y}}\sum_{\substack{W\in\mathscr{S}\\ W\neq X,Y,Z}} \mathtt{w}^2_{X,Z}(t.l)\cdot\mathtt{w}^1_W(t.r) + \sum_{\substack{Z\in\mathscr{S}\\ Z\neq X,Y}}\sum_{\substack{W\in\mathscr{A}\\ W\neq X,Y,Z}} \mathtt{w}^2_{X,Z}(t.l)\cdot\mathtt{w}^1_W(t.r)\\
&+\sum_{\substack{Z\in\mathscr{A}\\ Z\neq X,Y}}\sum_{\substack{W\in\mathscr{S}\\ W\neq X,Y,Z}} \mathtt{w}^2_{X,Z}(t.l)\cdot\mathtt{w}^1_W(t.r) + \sum_{\substack{Z\in\mathscr{A}\\ Z\neq X,Y}}\sum_{\substack{W\in\mathscr{A}\\ W\neq X,Y,Z}} \mathtt{w}^2_{X,Z}(t.l)\cdot\mathtt{w}^1_W(t.r)\\
=&\left(\sum_{Z\in\mathscr{S}} \mathtt{w}^2_{X,Z}(t.l)\right)\cdot\left(\sum_{W\in\mathscr{S}} \mathtt{w}^1_W(t.r)\right) + \sum_{\substack{W\in\mathscr{A}\\ W\neq X,Y}}\left(\sum_{Z\in\mathscr{S}} \mathtt{w}^2_{X,Z}(t.l)\right)\cdot\mathtt{w}^1_W(t.r)\\
&+\sum_{\substack{Z\in\mathscr{A}\\ Z\neq X,Y}} \mathtt{w}^2_{X,Z}(t.l))\cdot\left(\sum_{W\in\mathscr{S}} \mathtt{w}^1_W(t.r)\right) + \sum_{\substack{Z\in\mathscr{A}\\ Z\neq X,Y}}\sum_{\substack{W\in\mathscr{A}\\ W\neq X,Y,Z}} \mathtt{w}^2_{X,Z}(t.l)\cdot\mathtt{w}^1_W(t.r)\\
=&\mathtt{w}^2_{X,1}(t.l)\cdot\mathtt{w}^1_1(t.r) + \sum_{\substack{W\in\mathscr{A}\\ W\neq X,Y}} \mathtt{w}^2_{X,1}(t.l)\cdot\mathtt{w}^1_W(t.r)\\
&+\sum_{\substack{Z\in\mathscr{A}\\ Z\neq X,Y}} \mathtt{w}^2_{X,Z}(t.l)\cdot\mathtt{w}^1_1(t.r) + \sum_{\substack{Z\in\mathscr{A}\\ Z\neq X,Y}}\sum_{\substack{W\in\mathscr{A}\\ W\neq X,Y,Z}} \mathtt{w}^2_{X,Z}(t.l)\cdot\mathtt{w}^1_W(t.r)
\end{aligned}$$

After computing the singlet and doublet auxiliary values, the time complexity of the first, second, third, and fourth term is $O(1)$, $O(b)$, $O(b)$, and $O(b^2)$, giving us the time complexity result. $\square$

The proof for $\mathtt{w}^2_{X,Z}(t.l)$ and $\mathtt{w}^1_W(t.r)$ works for other pairs of function evaluations related to good edges. We therefore define a function $\mathbf{P}^{\mathbb{G}}_{X,Y}[\mathtt{G}_Z, \mathtt{H}_W](t_1,t_2)$ that computes these nested summations efficiently, for example setting $\mathtt{G}_Z = \mathtt{w}^2_{X,Z}$ (which will be evaluated at vertex $t1$) and $\mathtt{H}_W = \mathtt{w}^1_W$ (which will be evaluated at vertex $t2$). This gives us the corollary below.

Corollary B.7. *Let* $\mathbf{P}^{\mathbb{G}}_{X,Y}[\mathtt{G}_Z, \mathtt{H}_W](t_1,t_2)$ *denote* $\sum_{\substack{Z \in \mathscr{S}\cup\mathscr{A} \\ Z\neq X,Y}} \sum_{\substack{W \in \mathscr{S}\cup\mathscr{A} \\ W\neq X,Y,Z}} \mathtt{G}_Z(t_1) \cdot \mathtt{H}_W(t_2)$*, where* $\mathtt{G}_Z(t_1)$ *and* $\mathtt{H}_W(t_2)$ *are both mappings from a 2-tuple of an artificial taxon and a tree vertex to a weight value, i.e.,*

$$\mathtt{G}_Z, \mathtt{H}_W : (\mathscr{S} \cup \mathscr{A}) \times V(T) \to \mathbb{R},$$

which could be $\mathtt{w}^1_Z(\cdot)$*,* $\overline{\mathtt{w}}^1_Z(\cdot)$*,* $\mathtt{w}^2_{C,Z}(\cdot)$*,* $\overline{\mathtt{w}}^2_{C,Z}(\cdot)$*, or* $\mathtt{p}^2_{C,Z}(\cdot)$ *in our algorithms (we treat taxon C as a constant). Then,* $\mathbf{P}^{\mathbb{G}}_{X,Y}[\mathtt{G}_Z, \mathtt{H}_W](t_1,t_2)$ *can be computed in* $O(b^2)$ *time using the following equation:*

$$\begin{aligned}\mathbf{P}^{\mathbb{G}}_{X,Y}[\mathtt{G}_Z, \mathtt{H}_W](t_1,t_2) = \sum_{\substack{Z \in \mathscr{S}\cup\mathscr{A} \\ Z\neq X,Y}} \sum_{\substack{W \in \mathscr{S}\cup\mathscr{A} \\ W\neq X,Y,Z}} \mathtt{G}_1(t_1) \cdot \mathtt{H}_1(t_2) = \mathtt{G}_1(t_1) \cdot \mathtt{H}_1(t_2) + \sum_{\substack{W \in \mathscr{A} \\ W\neq X,Y}} \mathtt{G}_1(t_1) \cdot \mathtt{H}_W(t_2) \\ + \sum_{\substack{Z \in \mathscr{A} \\ Z\neq X,Y}} \mathtt{G}_Z(t_1) \cdot \mathtt{H}_1(t_2) + \sum_{\substack{Z \in \mathscr{A} \\ Z\neq X,Y}} \sum_{\substack{W \in \mathscr{A} \\ W\neq X,Y,Z}} \mathtt{G}_Z(t_1) \cdot \mathtt{H}_W(t_2) \end{aligned} \tag{B.42}$$

after computing the relevant auxiliary values.

Using Corollary B.7, we can reduce the cost of computing a good triplet for X,Y below t from $O((a+b)^2)$ to $O(b^2)$ by computing auxiliary values and aggregating the computation from singletons. The bottleneck now is evaluating the term enumerating all pairs of artificial taxa Z and W in the subtree below t, which gives us the $O(b^2)$ terms. In the following, we show how to reduce the time cost to $O(b)$.

Lemma B.10. $\mathbf{P}^{\mathbb{G}}_{X,Y}[\mathtt{w}^2_{X,Z}, \mathtt{w}^1_W](t_1,t_2)$ *can be computed in* $O(b)$ *time, after computing the singlet and doublet auxiliary values.*

Proof. According to Lemma B.9, we have

$$\begin{aligned}
&\mathbf{P}^{\mathbb{G}}_{X,Y}[\mathtt{w}^2_{X,Z},\mathtt{w}^1_W](t_1,t_2)\\
&= \sum_{\substack{Z\in\mathscr{S}\cup\mathscr{A}\\ Z\neq X,Y}} \sum_{\substack{W\in\mathscr{S}\cup\mathscr{A}\\ W\neq X,Y,Z}} \mathtt{w}^2_{X,Z}(t_1)\cdot\mathtt{w}^1_1(t_2) = \mathtt{w}^2_{1,1}(t_1)\cdot\mathtt{w}^1_1(t_2) + \sum_{\substack{W\in\mathscr{A}\\ W\neq X,Y}} \mathtt{w}^2_{X,1}(t_1)\cdot\mathtt{w}^1_W(t_2)\\
&+ \sum_{\substack{Z\in\mathscr{A}\\ Z\neq X,Y}} \mathtt{w}^2_{X,Z}(t_1)\cdot\mathtt{w}^1_1(t_2) + \sum_{\substack{Z\in\mathscr{A}\\ Z\neq X,Y}} \sum_{\substack{W\in\mathscr{A}\\ W\neq X,Y,Z}} \mathtt{w}^2_{X,Z}(t_1)\cdot\mathtt{w}^1_W(t_2) \qquad \text{(B.43)}
\end{aligned}$$

where the first three terms can be computed in $O(b)$ time. For the last term, we rewrite it as

$$\begin{aligned}
&\sum_{\substack{Z\in\mathscr{A}\\ Z\neq X,Y}} \sum_{\substack{W\in\mathscr{A}\\ W\neq X,Y,Z}} \mathtt{w}^2_{X,Z}(t_1)\cdot\mathtt{w}^1_W(t_2)\\
&= \sum_{Z\in\mathscr{A},Z\neq X,Y} \mathtt{w}^2_{X,Z}(t_1)\cdot \sum_{Z\in\mathscr{A},Z\neq X,Y} \mathtt{w}^1_Z(t_2) - \sum_{Z\in\mathscr{A},Z\neq X,Y} \mathtt{w}^2_{X,Z}(t_1)\cdot\mathtt{w}^1_Z(t_2), \qquad \text{(B.44)}
\end{aligned}$$

where all three sums in the equation can be computed in $O(b)$ time. If X or Y is a singleton, then $Z \neq X$ or $Z \neq Y$ is always true but this does not affect the computation. □

Once again, the proof for $\mathbf{P}^{\mathbb{G}}_{X,Y}[\mathtt{w}^2_{X,Z},\mathtt{w}^1_W](t_1,t_2)$ works for other pairs of function evaluations related to good edges besides $\mathtt{w}^2_{X,Z}(t1)$ and $\mathtt{w}^1_W(t2)$. This gives us the following corollary.

Corollary B.8. $\mathbf{P}^{\mathbb{G}}_{X,Y}[\mathtt{G}_Z,\mathtt{H}_W](t_1,t_2)$ *can be computed in* $O(b)$ *time, after computing the relevant auxiliary values.*

Putting this all together enables us to reduce the time complexity of computing the good triplet weights $\mathtt{w}^{x,z|w}_{X,Y}(t)$.

Corollary B.9 (efficient good triplet auxiliary values)**.** $\mathtt{w}^{x,z|w}_{X,Y}(t)$ *can be computed according to the*

following recurrence:

$$\mathtt{w}_{X,Y}^{x,z|w}(t) = \begin{cases} 0 & \textit{if } t \textit{ is a leaf in } T \\ \mathtt{exp}(-\mathtt{l}(t.l)) \cdot \mathtt{w}_{X,Y}^{x,z|w}(t.l) + \mathtt{exp}(-\mathtt{l}(t.r)) \cdot \mathtt{w}_{X,Y}^{x,z|w}(t.r) & \\ +(1-\mathtt{s}(t.l)) \cdot \mathtt{exp}(-\mathtt{l}(t.r)) \cdot \mathbf{P}_{X,Y}^{\mathbb{G}}[\mathtt{w}_{X,Z}^{2}, \mathtt{w}_{W}^{1}](t.r, t.l) & \\ +(1-\mathtt{s}(t.r)) \cdot \mathtt{exp}(-\mathtt{l}(t.l)) \cdot \mathbf{P}_{X,Y}^{\mathbb{G}}[\mathtt{w}_{X,Z}^{2}, \mathtt{w}_{W}^{1}](t.l, t.r) & \textit{otherwise} \end{cases} \tag{B.45}$$

After computing singlet and doublet auxiliary values, the time and space complexity for good triplets $\mathtt{w}_{X,Y}^{x,z|w}(t)$ *is the same as for bad triplets* $\mathtt{w}_{X,Y}^{x|z,w}(t)$ *(Corollary B.5).*

Proof. This correctness of the recurrence follows Lemma B.6 and Lemma B.10. After computing the singlet and doublet auxiliary values, $\mathtt{w}_{X,Y}^{x,z|w}(t)$ can be computed by traversing the $O(n)$ leaves of T in a postorder fashion. The remainder of the time and space complexity analysis for $\mathtt{w}_{X,Y}^{x,z|w}(t)$ is the same as for bad triplets $\mathtt{w}_{X,Y}^{x|z,w}(t)$ (Corollary B.5) □

Reducing the storage and time of auxiliary values for singleton taxa

In this section, we pick up from paragraph, "Reducing the storage and time complexity of auxiliary values for singleton taxa," in the main text. We reduce the storage of auxiliary values for singleton taxa, breaking the computation into three cases:

(i) both X and Y are artificial taxa,

(ii) one of X and Y is an artificial taxa (w.l.o.g. we take X to be the singleton and Y to be the artificial taxon), and

(ii) both X and Y are singleton taxa.

As described in the main text, these cases allow us to reduce the storage and time complexity of auxiliary values for singleton taxa. We now give recurrences for computing the simplified

recurrences for auxiliary values for singleton taxa, which are denoted by $\mathtt{p}$ instead of $\mathtt{w}$.

Corollary B.10 (simplified singlet auxiliary value for singleton taxa)**.** *Let X be a singleton. Then, $\mathtt{w}^1_X(t) = \mathtt{p}^1_X(t)$ can be computed according to the following recurrence:*

$$\mathtt{p}^1_X(t) = \begin{cases} \mathtt{I}(t) & \text{if } t \text{ is a leaf in } T \text{ with } L(t) = X \in \mathscr{S} \\ \exp(-\mathtt{l}(t.l)) \cdot \mathtt{p}^1_X(t.l) & \text{if } X \in \mathtt{below}(t.l) \\ \exp(-\mathtt{l}(t.r)) \cdot \mathtt{p}^1_X(t.r) & \text{if } X \in \mathtt{below}(t.r) \\ \textit{undefined} & \textit{otherwise} \end{cases}. \tag{B.46}$$

It takes $O(ah)$ time and $O(a)$ space to compute $\mathtt{p}^1_X(t)$ within the computation of $\Delta\mathbb{G}(X,Y)$ and $\Delta\mathbb{B}(X,Y)$ (Algorithm B.4).

Proof. The recurrence of $\mathtt{p}^1_X(t)$ follows from the recurrence for $\mathtt{w}^1_X(t)$ (Equation B.9 in Lemma B.1) but (1) replacing $\mathtt{w}^1_X(t)$ with $\mathtt{p}^1_X(t)$ and (2) simplifying based on whether $X \in \mathtt{below}(t.l)$ and $X \in \mathtt{below}(t.r)$. The recurrence for $\mathtt{p}^1_X(t)$ can be computed by traversing the $O(h)$ vertices on the path from the unique leaf labeled X to the root of gene tree T. Evaluating the recurrence for vertex t and taxon X takes $O(1)$ time and $O(1)$ space. Repeating for all $O(a)$ taxa $X \in \mathscr{S}$ gives time and storage complexity of $O(ah)$. We reduce the space complexity to $O(a)$ because we only need to maintain $\mathtt{p}^1_X(t)$ at one vertex t at a time during the postorder traversal of T when also computing $\Delta\mathbb{G}(X,Y)$ and $\Delta\mathbb{B}(X,Y)$. □

Corollary B.11 (doublet auxiliary value for at least one singleton)**.** *Let X be a singleton and Y be an artificial taxon. Then, $\mathtt{w}^2_{X,Y}(t) = \mathtt{p}^2_{X,Y}(t) = \mathtt{p}^2_{Y,X}(t)$ can be computed according to the following*

recurrence:

$$\mathtt{p}^2_{X,Y}(t) = \begin{cases} 0 & \text{if } L(t) = X \in \mathscr{S} \\ (1-\mathtt{s}(t.l)) \cdot \mathtt{p}^2_{X,Y}(t.l) & \\ +\mathtt{exp}(-\mathtt{l}(t.l)) \cdot \mathtt{p}^1_X(t.l) \cdot \mathtt{exp}(-\mathtt{l}(t.r)) \cdot \mathtt{w}^1_Y(t.r) & \text{if } X \in \mathtt{below}(t.l) \\ (1-\mathtt{s}(t.r)) \cdot \mathtt{p}^2_{X,Y}(t.r) & \\ +\mathtt{exp}(-\mathtt{l}(t.r)) \cdot \mathtt{p}^1_X(t.r) \cdot \mathtt{exp}(-\mathtt{l}(t.l)) \cdot \mathtt{w}^1_Y(t.l) & \text{if } X \in \mathtt{below}(t.r) \\ \textit{undefined} & \textit{otherwise} \end{cases}, \tag{B.47}$$

After precomputation, it takes $O(abh)$ time and $O(ab)$ space to compute $\mathtt{p}^2_{X,Y}(t)$ within the computation of $\Delta\mathbb{G}(X,Y)$ and $\Delta\mathbb{B}(X,Y)$ (Algorithm B.4). If Y is also a singleton, the time and space complexity are $O(a^2h)$ time and $O(a^2)$, respectively.

Proof. The recurrence for $\mathtt{p}^2_{X,Y}(t)$ follows from the recurrence for $\mathtt{w}^2_{X,Y}(t)$ (Equation B.14 in Lemma B.3) but (1) replacing $\mathtt{w}^1_X(t)$ with $\mathtt{p}^1_X(t)$, (2) replacing $\mathtt{w}^2_{X,Y}(t)$ with $\mathtt{p}^2_{X,Y}(t)$, and (3) simplifying based on whether $X \in \mathtt{below}(t.l)$ and $X \in \mathtt{below}(t.r)$ (etc). After precomputation, the recurrence can be computed by traversing the $O(h)$ vertices on the path from the unique labeled X to the root of gene tree T. Evaluating the recurrence for vertex t and a pair of taxa X,Y takes $O(1)$ time and $O(1)$ space. Repeating for all $O(a)$ taxa $X \in \mathscr{S}$ and for all taxa $Y \in \mathscr{A}$ gives time and space complexity of $O(abh)$. We reduce the space complexity to $O(ab)$ because we only need to maintain $\mathtt{p}^2_{X,Y}(t)$ at one vertex of t at a time during the postorder traversal of T when also computing $\Delta\mathbb{G}(X,Y)$ and $\Delta\mathbb{B}(X,Y)$. If Y is a singleton, then we simply change "for all taxa $Y \in \mathscr{A}$" to "for all taxa $Y \in \mathscr{S}$" in the analysis above. □

Corollary B.12 (efficient bad triplet auxiliary value for one singleton and one artificial taxa)**.** *Let X be a singleton taxa and Y be an artificial taxa. Then, $\mathtt{w}^{x|z,w}_{X,Y}(t) = \mathtt{p}^{x|z,w}_{X,Y}(t)$ can be computed*

according the following recurrence:

$$\mathtt{p}_{X,Y}^{x|z,w}(t) = \begin{cases} 0 & \text{if } L(t) = X \\ \exp(-\mathtt{l}(t.l)) \cdot \mathtt{p}_{X,Y}^{x|z,w}(t.l) & \\ \quad + \exp(-\mathtt{l}(t.l)) \cdot \mathtt{p}_X^1(t.l) \cdot (1 - \mathtt{s}(t.r)) \cdot \mathbf{P}_{X,Y}^{\mathbb{B}}[\mathtt{w}_{Z,W}^2](t.r) & \text{if } X \in \mathtt{below}(t.l) \\ \exp(-\mathtt{l}(t.r)) \cdot \mathtt{p}_{X,Y}^{x|z,w}(t.r) & \\ \quad + \exp(-\mathtt{l}(t.r)) \cdot \mathtt{p}_X^1(t.r) \cdot (1 - \mathtt{s}(t.l)) \cdot \mathbf{P}_{X,Y}^{\mathbb{B}}[\mathtt{w}_{Z,W}^2](t.l) & \text{if } X \in \mathtt{below}(t.r) \\ \textit{undefined} & \textit{otherwise} \end{cases} \tag{B.48}$$

After precomputation, it takes $O(ab^2h)$ time and $O(ab)$ space to compute $\mathtt{p}_{X,Y}^{x|z,w}(t)$ within the computation of $\Delta\mathbb{B}(X,Y)$ (Algorithm B.4).

Proof. The recurrence for $\mathtt{p}_{X,Y}^{x|z,w}$ follows Equation B.41 (Corollary B.6) but (1) replacing $\mathtt{w}_X^1(t)$ with $\mathtt{p}_X^1(t)$, (2) replacing $\mathtt{w}_{X,Y}^{x|z,w}(t)$ with $\mathtt{p}_{X,Y}^{x|z,w}(t)$, and (3) simplifying based on whether $X \in \mathtt{below}(t.l)$ and $X \in \mathtt{below}(t.r)$ (etc). After precomputation, the recurrence can be computed by traversing the $O(h)$ vertices on the path from the unique labeled X to the root of gene tree T. Evaluating the recurrence for vertex t and taxon X takes $O(b)$ time and $O(1)$ space. Repeating for all $O(a)$ taxa $X \in \mathscr{S}$ and for all taxa $Y \in \mathscr{A}$ gives time and space complexity of $O(ab^2h)$ and $O(abh)$, respectively. We reduce the space complexity to $O(ab)$ because we only need to maintain $\mathtt{p}_{X,Y}^{x|z,w}$ at one vertex of t at a time during the postorder traversal when also computing $\Delta\mathbb{G}(X,Y)$ and $\Delta\mathbb{B}(X,Y)$. □

Corollary B.13 (efficient good triplet auxiliary values for one singleton and one artificial taxa)**.** *Let X be a singleton taxa and Y be an artificial taxa. Then, $\mathtt{w}_{X,Y}^{x,z|w}(t) = \mathtt{p}_{X,Y}^{x,z|w}(t)$ can be computed*

using the following recurrence:

$$\mathtt{p}^{x,z|w}_{X,Y}(t) = \begin{cases} 0 & \text{if } L(t) = X \\ \mathtt{exp}(-\mathtt{l}(t.l)) \cdot \mathtt{p}^{x,z|w}_{X,Y}(t.l) & \\ +(1-\mathtt{s}(t.l)) \cdot \mathbf{P}^{\mathbb{G}}_{X,Y}[\mathtt{p}^{2}_{X,Z}, \mathtt{w}^{1}_{W}](t.l,t.r) & \text{if } X \in \mathtt{below}(t.l) \\ \mathtt{exp}(-\mathtt{l}(t.r)) \cdot \mathtt{p}^{x,z|w}_{X,Y}(t.r) & \\ +(1-\mathtt{s}(t.r)) \cdot \mathbf{P}^{\mathbb{G}}_{X,Y}[\mathtt{p}^{2}_{X,Z}, \mathtt{w}^{1}_{W}](t.r,t.l) & \text{if } X \in \mathtt{below}(t.r) \\ \textit{undefined} & \textit{otherwise} \end{cases} \tag{B.49}$$

After precomputation, the time and storage complexity for $\mathtt{p}^{x,z|w}_{X,Y}(t)$ *is the same as for* $\mathtt{p}^{x|z,w}_{X,Y}(t)$ *(Corollary B.12).*

Proof. The recurrence for $\mathtt{p}^{x,z|w}_{X,Y}$ follows Equation B.45 (Corollary B.9) but (1) replacing $\mathtt{w}^{1}_{X}(t)$ with $\mathtt{p}^{1}_{X}(t)$, (2) replacing $\mathtt{w}^{x,z|w}_{X,Y}(t)$ with $\mathtt{p}^{x,z|w}_{X,Y}(t)$, and (3) simplifying based on whether $X \in \mathtt{below}(t.l)$ and $X \in \mathtt{below}(t.r)$ (etc). The time and space cost analysis is the same as the auxiliary values for bad triplets (Corollary B.12). □

Corollary B.14 (efficient bad and good triplet auxiliary values for two singletons)**.** *Let X and Y be singletons. Observe that good triplets auxiliary values* $\mathtt{p}^{x,z|w}_{X,Y}(t)$ *are equal for all* $Y \in \mathscr{S}$ *because Y does not need to be excluded from the computation (Definition B.20). We use* $\mathtt{p}^{x,z|w}_{X,0}(t)$ *to denote the bad triplet auxiliary value for any* $Y \in \mathscr{S}$*; it can be computed in* $O(abh)$ *time and* $O(a)$ *space. The same holds for bad triplet auxiliary values* $\mathtt{p}^{x|z,w}_{X,Y}(t)$.

Proof. The corollary follows from Corollary B.13 but reduces the time and storage complexity by a factor of (b) since we no longer need to evaluate the good triplet $\mathtt{p}^{x,z|w}_{X,Y}(t)$ for all $Y \in \mathscr{A}$ (we can select one singleton arbitrarily, denoted 0). The same holds for the bad triplet $\mathtt{p}^{x,z|w}_{X,Y}(t)$ but following Corollary B.12. □

Lastly, recall that for bad triplets $\mathrm{p}^{x|z,w}_{X,Y}(t)$ does not equal $\mathrm{p}^{x|z,w}_{Y,X}(t)$ because they differ by whether the outgroup x is labeled X or Y; however, we can use the same recurrence due to symmetry swapping the ordering of the labels. The same is holds for good triplets.

Efficient algorithms for computing $\Delta\mathbb{B}_t$ and $\Delta\mathbb{G}_t$

Finally, we incorporate our algorithmic improvements into the computation of $\Delta\mathbb{G}_t(X,Y)$ and $\Delta\mathbb{B}_t(X,Y)$, splitting it into three cases: (i) both X and Y are artificial taxa, (ii) one of X and Y is a singleton and the other is artificial (w.l.o.g. we take X to be the singleton and Y to be the artificial taxa), and (ii) both X and Y are singletons. Then we have the following corollaries:

Corollary B.15. *$\Delta\mathbb{B}_t(X,Y)$ can be computed in $O(b)$ time according to the following equation:*

$$\Delta\mathbb{B}_t(X,Y) = \begin{cases} \mathtt{w}^1_Y(t.r)\cdot\exp(-\mathtt{l}(t.r)\cdot \mathtt{w}^{x|z,w}_{X,Y}(t.l)\cdot\exp(-\mathtt{l}(t.l)) \\ +\exp(-\mathtt{l}(t.l))\cdot\mathtt{w}^1_X(t.l)\cdot\exp(-\mathtt{l}(t.r))\cdot\mathtt{w}^1_Y(t.r)\cdot\mathbf{P}^{\mathbb{B}}_{X,Y}[\overline{\mathtt{w}}^2_{Z,W}](t) \\ +\mathtt{w}^1_X(t.l)\cdot\exp(-\mathtt{l}(t.l)\cdot \mathtt{w}^{x|z,w}_{Y,X}(t.r)\cdot\exp(-\mathtt{l}(t.r)) \\ \qquad\qquad \textit{if both } X \textit{ and } Y \textit{ are artificial} \\ \mathtt{w}^1_Y(t.r)\cdot\exp(-\mathtt{l}(t.r)\cdot \mathtt{p}^{x|z,w}_{X,Y}(t.l)\cdot\exp(-\mathtt{l}(t.l)) \\ +\exp(-\mathtt{l}(t.l))\cdot\mathtt{p}^1_X(t.l)\cdot\exp(-\mathtt{l}(t.r))\cdot\mathtt{w}^1_Y(t.r)\cdot\mathbf{P}^{\mathbb{B}}_{X,Y}[\overline{\mathtt{w}}^2_{Z,W}](t) \\ +\mathtt{p}^1_X(t.l)\cdot\exp(-\mathtt{l}(t.l)\cdot \mathtt{w}^{x|z,w}_{Y,0}(t.r)\cdot\exp(-\mathtt{l}(t.r)) \\ \qquad\qquad \textit{if } X \textit{ is a singleton and } Y \textit{ is artificial} \\ \mathtt{p}^1_Y(t.r)\cdot\exp(-\mathtt{l}(t.r)\cdot \mathtt{p}^{x|z,w}_{X,0}(t.l)\cdot\exp(-\mathtt{l}(t.l)) \\ +\exp(-\mathtt{l}(t.l))\cdot\mathtt{p}^1_X(t.l)\cdot\exp(-\mathtt{l}(t.r))\cdot\mathtt{p}^1_Y(t.r)\cdot\mathbf{P}^{\mathbb{B}}_{X,Y}[\overline{\mathtt{w}}^2_{Z,W}](t) \\ +\mathtt{p}^1_X(t.l)\cdot\exp(-\mathtt{l}(t.l)\cdot \mathtt{p}^{x|z,w}_{Y,0}(t.r)\cdot\exp(-\mathtt{l}(t.r)) \\ \qquad\qquad \textit{if both } X \textit{ and } Y \textit{ are singletons} \end{cases} \tag{B.50}$$

Proof. The corollary follows from Theorem B.1 but (1) replacing $\mathtt{w}_A^1$ with $\mathtt{p}_A^1$ if A is a singleton (Corollary B.10), (2) replacing $\mathtt{w}_A^1$ with $\mathtt{p}_A^1$ if A is a singleton (Corollary B.11), (3) replacing $\mathtt{w}_{A,B}^{x|z,w}$ with $\mathtt{w}_{B,0}^{x|z,w}$ if A is artificial and B is singleton (Definition B.7), (4) replacing $\mathtt{w}_{A,B}^{x|z,w}$ with $\mathtt{p}_{A,B}^{x|z,w}$ if A is singleton and B is artificial (Corollary B.12), and (5) replacing with $\mathtt{w}_{A,B}^{x|z,w}$ with $\mathtt{p}_{A,0}^{x|z,w}$ if A and B are both singletons (Corollary B.14). After precomputation, the previous quadratic term, now given by the function $\mathbf{P}_{X,Y}^{\mathbb{B}}$ can be computed according to Corollary B.5 so the time cost to evaluate $\Delta\mathbb{B}_t(X,Y)$ is $O(b)$. □

Corollary B.16. *$\Delta\mathbb{G}_t(X,Y)$ can be computed in $O(b)$ time according to the following equation:*

$$\Delta\mathbb{G}_t(X,Y) = \begin{cases} \mathtt{w}^1_Y(t.r)\cdot\exp(-\mathtt{l}(t.r))\cdot\mathtt{w}^{x,z|w}_{X,Y}(t.l)\cdot\exp(-\mathtt{l}(t.l)) \\ +\exp(-\mathtt{l}(t.r))\cdot\mathtt{w}^1_Y(t.r)\cdot(1-\mathtt{s}(t.l))\cdot\mathbf{P}^{\mathbb{G}}_{X,Y}[\mathtt{w}^2_{X,Z},\overline{\mathtt{w}}^1_W](t.l,t) \\ +(1-\mathtt{s}(t.l))\cdot(1-\mathtt{s}(t.r))\cdot\mathbf{P}^{\mathbb{G}}_{X,Y}[\mathtt{w}^2_{X,Z},\mathtt{w}^2_{Y,W}](t.l,t.r) \\ +\exp(-\mathtt{l}(t.l))\cdot\mathtt{w}^1_X(t.l)\cdot(1-\mathtt{s}(t.r))\cdot\mathbf{P}^{\mathbb{G}}_{X,Y}[\mathtt{w}^2_{Y,Z},\overline{\mathtt{w}}^1_W](t.r,t) \\ +\mathtt{w}^1_X(t.l)\cdot\exp(-\mathtt{l}(t.l))\cdot\mathtt{w}^{x,z|w}_{Y,X}(t.r)\cdot\exp(-\mathtt{l}(t.r)) \\ \qquad\qquad \textit{if both } X \textit{ and } Y \textit{ are artificial} \\ \mathtt{w}^1_Y(t.r)\cdot\exp(-\mathtt{l}(t.r))\cdot\mathtt{p}^{x,z|w}_{X,Y}(t.l)\cdot\exp(-\mathtt{l}(t.l)) \\ +\exp(-\mathtt{l}(t.r))\cdot\mathtt{w}^1_Y(t.r)\cdot(1-\mathtt{s}(t.l))\cdot\mathbf{P}^{\mathbb{G}}_{X,Y}[\mathtt{p}^2_{X,Z},\overline{\mathtt{w}}^1_W](t.l,t) \\ +(1-\mathtt{s}(t.l))\cdot(1-\mathtt{s}(t.r))\cdot\mathbf{P}^{\mathbb{G}}_{X,Y}[\mathtt{p}^2_{X,Z},\mathtt{w}^2_{Y,W}](t.l,t.r) \\ +\exp(-\mathtt{l}(t.l))\cdot\mathtt{p}^1_X(t.l)\cdot(1-\mathtt{s}(t.r))\cdot\mathbf{P}^{\mathbb{G}}_{X,Y}[\mathtt{w}^2_{Y,Z},\overline{\mathtt{w}}^1_W](t.r,t) \\ +\mathtt{p}^1_X(t.l)\cdot\exp(-\mathtt{l}(t.l))\cdot\mathtt{w}^{x,z|w}_{Y,0}(t.r)\cdot\exp(-\mathtt{l}(t.r)) \\ \qquad\qquad \textit{if } X \textit{ is a singleton and } Y \textit{ is artificial} \\ \mathtt{p}^1_Y(t.r)\cdot\exp(-\mathtt{l}(t.r))\cdot\mathtt{p}^{x,z|w}_{X,0}(t.l)\cdot\exp(-\mathtt{l}(t.l)) \\ +\exp(-\mathtt{l}(t.r))\cdot\mathtt{p}^1_Y(t.r)\cdot(1-\mathtt{s}(t.l))\cdot\mathbf{P}^{\mathbb{G}}_{X,Y}[\mathtt{p}^2_{X,Z},\overline{\mathtt{w}}^1_W](t.l,t) \\ +(1-\mathtt{s}(t.l))\cdot(1-\mathtt{s}(t.r))\cdot\mathbf{P}^{\mathbb{G}}_{X,Y}[\mathtt{p}^2_{X,Z},\mathtt{p}^2_{Y,W}](t.l,t.r) \\ +\exp(-\mathtt{l}(t.l))\cdot\mathtt{p}^1_X(t.l)\cdot(1-\mathtt{s}(t.r))\cdot\mathbf{P}^{\mathbb{G}}_{X,Y}[\mathtt{p}^2_{Y,Z},\overline{\mathtt{w}}^1_W](t.r,t) \\ +\mathtt{p}^1_X(t.l)\cdot\exp(-\mathtt{l}(t.l))\cdot\mathtt{p}^{x,z|w}_{Y,0}(t.r)\cdot\exp(-\mathtt{l}(t.r)) \\ \qquad\qquad \textit{if both } X \textit{ and } Y \textit{ are singletons} \end{cases} \tag{B.51}$$

Proof. The corollary follows from Theorem B.2 but (1) replacing $\mathtt{w}^1_A$ with $\mathtt{p}^1_A$ if A is a singleton

(Corollary B.10), (2) replacing $\mathtt{w}_A^1$ with $\mathtt{p}_A^1$ if A is a singleton (Corollary B.11), (3) replacing $\mathtt{w}_{A,B}^{x,z|w}$ with $\mathtt{w}_{B,0}^{x,z|w}$ if A is artificial and B is singleton (Definition B.8), (4) replacing $\mathtt{w}_{A,B}^{x,z|w}$ with $\mathtt{p}_{A,B}^{x,z|w}$ if A is singleton and B is artificial (Corollary B.13), and (5) replacing with $\mathtt{w}_{A,B}^{x,z|w}$ with $\mathtt{p}_{A,0}^{x,z|w}$ if A and B are both singletons (Corollary B.14). After precomputation, the previous quadratic term, now given by the function $\mathbf{P}_{X,Y}^{\mathbb{G}}$ can be computed according to Corollary B.8 so the time cost to evaluate $\Delta\mathbb{G}_t(X,Y)$ is $O(b)$. □

B.3.4 Pseudocode

Algorithm B.2: Weighted Quartet Graph Construction

Input: an input gene tree T
Output: the quartet graph represented by matrices $\mathbb{G}$ and $\mathbb{B}$

$T_{\mathtt{l}} \leftarrow$ a copy of T with all branch support values being 0
arbitrarily resolve T and $T_{\mathtt{l}}$ (and set support and length values of new edges to 0)
$\mathbb{G}_{\mathtt{l}}$, $\mathbb{B}_{\mathtt{l}} \leftarrow$ `edge-weight(`$T_{\mathtt{l}}$`)`
$\Delta\mathbb{G}$, $\Delta\mathbb{B} \leftarrow$ `edge-weight(`T`)`
return $\mathbb{G}_{\mathtt{l}} - \Delta\mathbb{G}$, $\mathbb{B}_{\mathtt{l}} - \Delta\mathbb{B}$

Algorithm B.3: Precomputation of auxiliary values for artificial taxa

Input: an input gene tree T

Output: all auxiliary values: $\mathtt{w}^1, \mathtt{w}^2, \overline{\mathtt{w}}^1, \overline{\mathtt{w}}^2, \mathtt{w}^{x|z,w}, \mathtt{w}^{x,z|w}$ and the precomputed *term* for evaluating $\mathbf{P}^{\mathbb{B}}$ efficiently according to Corollary B.5

for *each node t in a postordering of T* **do**
  $\mathtt{w}_1^1(t) \leftarrow$ Equation B.26
  for *each X in $\mathscr{A}$* **do**
    $\mathtt{w}_X^1(t) \leftarrow$ Equation B.9
  $\mathtt{w}_{1,1}^2(t) \leftarrow$ Equation B.30
  for *each X in $\mathscr{A}$* **do**
    $\mathtt{w}_{1,X}^2 = \mathtt{w}_{X,1}^2 \leftarrow$ Equation B.34
  for *each (X,Y) in $\mathscr{A} \times \mathscr{A}$* **do**
    if $X \neq Y$ **then** $\mathtt{w}_{X,Y}^2(t) \leftarrow$ Equation B.14
for *each vertex t in a preorder traversal of T* **do**
  $\overline{\mathtt{w}}_1^1(t) \leftarrow$ Equation B.27
  for *each X in $\mathscr{A}$* **do**
    $\overline{\mathtt{w}}_X^1(t) \leftarrow$ Equation B.11
  $\overline{\mathtt{w}}_{1,1}^2(t) \leftarrow$ Equation B.31
  for *each X in $\mathscr{A}$* **do**
    $\overline{\mathtt{w}}_{1,X}^2 = \overline{\mathtt{w}}_{X,1}^2 \leftarrow$ Equation B.35
  for *each (X,Y) in $\mathscr{A} \times \mathscr{A}$* **do**
    if $X \neq Y$ **then** $\overline{\mathtt{w}}_{X,Y}^2(t) \leftarrow$ Equation B.16
for *each vertex t in a postorder traversal of T* **do**
  $term(t) \leftarrow 0$ **for** *each (X,Y) in $\mathscr{A} \times \mathscr{A} \cup \{0\}$* **do**
    if $X \neq Y$ **then**
      $\mathtt{w}_{X,Y}^{x|z,w}(t) \leftarrow$ Equation B.41
      $\mathtt{w}_{X,Y}^{x,z|w}(t) \leftarrow$ Equation B.45
      if $Y \neq 0$ **then** $term(t) \leftarrow term(t) + w_{X,Y}^2(t)$
return $\mathtt{w}^1, \mathtt{w}^2, \overline{\mathtt{w}}^1, \overline{\mathtt{w}}^2, \mathtt{w}^{x|z,w}, \mathtt{w}^{x,z|w}, term$

Algorithm B.4: Good and Bad Edge Weight Computation. Recall that the auxiliary quantities simplified for singleton taxa, denoted, $\mathtt{p}$ and are computed via a postorder traversal, along with $\Delta\mathbb{B}$ and $\Delta\mathbb{G}$. The goal is to reduce storage complexity for auxiliary values. At the leaves, there is one $\mathtt{p}$ auxiliary value for every singleton, and they are moved up tree during the postorder traversal. As an example, when we compute $\mathtt{p}_X^1(t)$, we can either overwrite $\mathtt{p}_X^1(t.l)$ or $\mathtt{p}_X^1(t.r)$, depending on whether $X \in \{L(x) : x \in \mathtt{below}(t.l)\}$ or $X \in \{L(x) : x \in \mathtt{below}(t.r)\}$ (only one is possible).

Input: an input gene tree T

Output: weights of good and bad edges, stored in the matrices $\Delta\mathbb{G}$ and $\Delta\mathbb{B}$

$\mathtt{w}^1, \mathtt{w}^2, \overline{\mathtt{w}}^1, \overline{\mathtt{w}}^2, \mathtt{w}^{x|z,w}, \mathtt{w}^{x,z|w}, term \leftarrow \mathtt{pre\text{-}processing}(T)$
for *each vertex t in a postorder traversal of T* **do**
  for *each (X,Y) in* $\{L(x) : x \in \mathtt{below}(t.l)\} \times \{L(x) : x \in \mathtt{below}(t.r)\}$ **do**
    if $X \neq Y$ **then**
      $\Delta\mathbb{G}_t(X,Y) \leftarrow$ Equation B.51
      $\Delta\mathbb{B}_t(X,Y) \leftarrow$ Equation B.50
      $\Delta\mathbb{G}(X,Y) \leftarrow \Delta\mathbb{G}(X,Y) + \Delta\mathbb{G}_t(X,Y)$
      $\Delta\mathbb{B}(X,Y) \leftarrow \Delta\mathbb{B}(X,Y) + \Delta\mathbb{B}_t(X,Y)$
  for *each X in* $\{L(x) : x \in \mathtt{below}(t.l) \cup \mathtt{below}(t.r)\}$ **do**
    if $X \in \mathscr{S}$ **then**
      $\mathtt{p}_X^1(t) \leftarrow$ Equation B.46, overwrite $\mathtt{p}_X^1(t.l)$ or $\mathtt{p}_X^1(t.r)$
      for *each $Y \in \mathscr{A} \cup \{1\}$* **do**
        $\mathtt{p}_{X,Y}^2(t) \leftarrow$ Equation B.47, overwrite $\mathtt{p}_{X,Y}^2(t.l)$ or $\mathtt{p}_{X,Y}^2(t.r)$
      $\mathtt{p}_{X,0}^{x|z,w}(t) \leftarrow$ Equation B.48, overwrite $\mathtt{p}_{X,0}^{x|z,w}(t.l)$ or $\mathtt{p}_{X,0}^{x|z,w}(t.r)$
      $\mathtt{p}_{X,0}^{x,z|w}(t) \leftarrow$ Equation B.49, overwrite $\mathtt{p}_{X,0}^{x,z|w}(t.l)$ or $\mathtt{p}_{X,0}^{x,z|w}(t.r)$
      for *each $Y \in \mathscr{A}$* **do**
        $\mathtt{p}_{X,Y}^{x|z,w}(t) \leftarrow$ Equation B.48, overwrite $\mathtt{p}_{X,Y}^{x|z,w}(t.l)$ or $\mathtt{p}_{X,Y}^{x|z,w}(t.r)$
        $\mathtt{p}_{X,Y}^{x,z|w}(t) \leftarrow$ Equation B.49, overwrite $\mathtt{p}_{X,Y}^{x,z|w}(t.l)$ or $\mathtt{p}_{X,Y}^{x,z|w}(t.r)$
return $\Delta\mathbb{G}$, $\Delta\mathbb{B}$

B.3.5 Final time and space complexity results

Theorem B.4. *Let n be the number leaves in the gene tree T, which is rooted arbitrarily with height h. Let $s = a + b$ be the number of leaf labels for the current subproblem, where a is the number of singletons and b is the number of non-singletons (or artificial taxa). Then, we can compute $\mathbb{G}$ and $\mathbb{B}$ correctly in $O(a^2b + ab^2n + b^3n)$ time and $O(a^2 + ab + b^2n)$ space using Algorithm B.2.*

Proof. Correctness follows from Corollaries B.15 and B.16 because the modifications to the equations only change the computational efficiency not the results.

To construct the quartet graph from gene tree T (Algorithm B.2), we apply Algorithm B.4 to the gene tree two times and the combines the results into the final graph . Therefore, we analyze the space and time complexity of Algorithm B.4.

Algorithm B.4 first computes the auxiliary values during a precomputation phase (Algorithm B.3). Specifically, the auxiliary values $\mathtt{w}^1_X(t)$, $\mathtt{w}^2_{X,Y}(t)$, $\overline{\mathtt{w}}^1_X(t)$, $\overline{\mathtt{w}}^2_{X,Y}(t)$, $\mathtt{w}^{x|z,w}_{X,Y}(t)$, and $\mathtt{w}^{x,z|w}_{X,Y}(t)$ are computed for artificial taxa (plus the special taxa 0 and 1) and for all vertex $t \in V(T)$. We split the cost of the procomputation into three cases. In each case, the cost is determined by the number of values computed and the time cost to compute each value, i.e., the time complexity is $O(A) \times O(B) = O(AB)$ if there are $O(A)$ values and each value takes $O(B)$ time to compute. The cost of each auxiliary value is

- $\mathtt{w}^1_X(t)$ and $\overline{\mathtt{w}}^1_X(t)$: $O(bn) \times O(1) = O(bn)$, according to Lemmas B.1–B.2 and Corollaries B.1,
- $\mathtt{w}^2_{X,Y}(t)$ and $\overline{\mathtt{w}}^2_{X,Y}(t)$: $O(b^2n) \times O(1) = O(b^2n)$, according to Lemmas B.3–B.4 and Corollaries B.2–B.3
- $\mathtt{w}^{x|z,w}_{X,Y}(t)$ and $\mathtt{w}^{x,z|w}_{X,Y}(t)$: $O(b^2n) \times O(b) = O(b^3n)$, according to Corollaries B.6 and B.9.

Putting this together, the time and space complexity of precomputation for auxiliary values $\mathtt{w}$ is $O(b^3n)$ and $O(b^2n)$, respectively.

After the precomputation phase, Algorithm B.4 computes $\Delta\mathbb{G}_t(X,Y)$ and $\Delta\mathbb{B}_t(X,Y)$, which are defined such that X labels leaves below $t.l$ and Y labels leaves below $t.r$. The time to compute $\Delta\mathbb{G}_t(X,Y)$ and $\Delta\mathbb{B}_t(X,Y)$ is determined by the number of (X,Y,t) 3-tuples and the cost to process each 3-tuple. The number of (X,Y,t) 3-tuples is $O(a^2+abh+b^2n)$, where h is the height of T. To show this, we split the problem into three cases and consider the number of t:

- t is unique if both X and Y are singletons so the number of 3-tuples is $O(a^2)\times O(1)=O(a^2)$,
- the number of t is $O(h)$ if one of X and Y is artificial so the number of 3-tuples is $O(ab)\times O(h)=O(abh)$,
- the number of t is $O(n)$ if both X and Y are artificial so the number of 3-tuple is $O(b^2)\times O(n)=O(b^2n)$.

Adding the three cases up, we show the total number of 3-tuples is $O(a^2+abh+b^2n)$. Given a 3-tuple of (X,Y,t), both $\Delta\mathbb{G}_t(X,Y)$ and $\Delta\mathbb{B}_t(X,Y)$ takes $O(b)$ time to compute according to Corollaries B.16-B.15, so the total time complexity of the procedure is $O(a^2+abh+b^2n)\times O(b)=O(a^2b+ab^2h+b^3n)$. No extra space is needed because the quartet graph itself $O((a+b)^2)=O(a^2+ab+b^2)$ because we update the weights of the good and bad edges in place.

Lastly, while computing $\Delta\mathbb{G}$ and $\Delta\mathbb{B}$ in a postorder traversal of T, we also compute $\mathtt{p}^1_X(t)$, $\mathtt{p}^2_{X,Y}(t)$, $\mathtt{p}^{x|z,w}_{X,Y}(t)$, and $\mathtt{p}^{x,z|w}_{X}(t)$. The cost of each quantity is

- $\mathtt{p}^1_X(t)$: $O(a)\times O(h)=O(a)$, according to Corollary B.10,
- $\mathtt{p}^2_{X,Y}(t)$: $O(ab)\times O(h)=O(abh)$, according to Corollary B.11,
- $\mathtt{p}^{x|z,w}_{X,Y}(t)$ and $\mathtt{p}^{x,z|w}_{X,Y}(t)$: $O(ab)\times O(bh)=O(ab^2h)$, according to Corollaries B.12 and B.13.

The storage cost of the $\mathtt{p}$ values is no greater than $O(ab)$ because they are updated on the fly. Consequently, computing the auxiliary values for singletons $\mathtt{p}(\cdot)$ does not affect the time or space complexity of Algorithm B.4.

The final time complexity of Algorithm B.2 is $O(a^2b + ab^2h + b^3n)$ by putting the cost of precomputation and edge weight computation together. The final space time complexity is $O(a^2 + ab + b^2n)$ because we combine the storage for the quartet graph $O((a+b)^2) = O(a^2 + ab + b^2)$ with the storage from precomputation $O(b^2n)$

Although they do not impact the storage complexity, for completeness we mention storage for the subproblem forest data structure $O(n)$, which must be maintained across subproblems on the stack (for total of $O(n^2)$ which is the same as the quartet graph when $a = n$) and storage for the input gene trees $O(nk)$. □

Theorem B.5. *Let n be the number of species, and let k be the number of gene trees. weighted TREE-QMC has time complexity of $O(n^4k)$ if subproblems are produced in a perfectly balanced fashion, and moreover a time complexity of $O(n^2 \log n \cdot k)$ if we assume the number of artificial taxa in each subproblem is a constant.*

Proof. At each step in the divide phase of the algorithm, we compute a bipartition and then recurse on two subproblems. Let s be the number of taxa in the subproblem, i.e., $s = a + b$. The work per subproblem has time complexity $O(s^3nk + s^2) = O(s^3nk)$ where the first term comes from constructing the quartet graph by applying our algorithm to all gene trees ($O(a^2b + ab^2h + b^3n) = O(s^3n)$ by Theorem B.4) and the second term comes from applying our approach for seeking a max-cut (Theorem 2 in [45]).

To proceed with the analysis, we assume the subproblem decomposition is perfectly balanced, as in Theorem 3 in [45]. Under this assumption, $T(s) = 2T(s/2) + O(s^3nk)$, and by the master theorem we get by we have $T(s) = O(s^3nk)$ for divide-and-conquer recurrence. This evaluates to $O(n^4k)$ if $s = n$. If we also assume the number of artificial taxa b is a constant, the work per subproblem becomes $O((a^2 + ah + n)k + s^2) = O(s^2k + snk)$. Then we have $T(s) = 2T(s/2) + O(s^2k + snk)$. By splitting the $T(s)$ into $T_1(s) = 2T_1(s/2) + O(s^2k)$ and $T_2(s) = 2T_2(s/2) + O(snk)$, we obtain $T_1(s) = O(s^2k)$ and $T_2(s) = O(s \log s \cdot nk)$ by the master theorem so the total time is

$T_1(s) + T_2(s) = O(s^2k + s\log s \cdot nk)$. This evaluates to $O(n^2 \log n \cdot k)$ when $s = n$.

□

B.4 Experimental Study

B.4.1 Properties of Simulated Data

Table B.1: **Properties of Asteroid simulated data from [86].** Data were downloaded from `https://cme.h-its.org/exelixis/material/asteroid_data.tar.gz` in December 2022. For each replicate, ILS is the fraction of branches in true species tree missing from the true gene tree, averaged across all gene trees. GTEE is the fraction of branches in true gene tree that are missing from the estimated gene tree, averaged across all gene trees. AD is the fraction of branches in the true species tree that are missing from the estimated gene tree, averaged across all gene trees. The values in the table are the average ($\pm$ standard deviation) across all replicates. Dup indicate the row is duplicated.

		ILS	GTEE	AD	Proportion missing taxa
Varying population size	10	0.0000 ± 0.0000	0.3419 ± 0.0453	0.3419 ± 0.0453	0.8115 ± 0.0107
	50000000	0.0610 ± 0.0297	0.3530 ± 0.0483	0.3616 ± 0.0496	0.8106 ± 0.0118
	100000000	0.1119 ± 0.0359	0.3598 ± 0.0440	0.3836 ± 0.0478	0.8109 ± 0.0115
	500000000	0.3854 ± 0.0722	0.3904 ± 0.0413	0.5428 ± 0.0582	0.8131 ± 0.0099
	1000000000	0.5576 ± 0.0626	0.4071 ± 0.0395	0.6579 ± 0.0501	0.8118 ± 0.0089
Varying number of taxa	25	0.0492 ± 0.0314	0.3008 ± 0.0456	0.3085 ± 0.0485	0.7607 ± 0.0104
	75	0.0626 ± 0.0225	0.3872 ± 0.0382	0.3960 ± 0.0388	0.8255 ± 0.0088
	50	0.0610 ± 0.0297	0.3530 ± 0.0483	0.3616 ± 0.0496	0.8106 ± 0.0118 (dup)
	100	0.0733 ± 0.0270	0.4125 ± 0.0442	0.4220 ± 0.0454	0.8320 ± 0.0076
	125	0.0835 ± 0.0274	0.4384 ± 0.0397	0.4487 ± 0.0417	0.8340 ± 0.0073
	150	0.0841 ± 0.0284	0.4524 ± 0.0362	0.4625 ± 0.0371	0.8370 ± 0.0073
Varying number of genes	250	0.0602 ± 0.0256	0.3504 ± 0.0487	0.3584 ± 0.0500	0.8127 ± 0.0100
	500	0.0592 ± 0.0264	0.3527 ± 0.0435	0.3603 ± 0.0448	0.8110 ± 0.0094
	1000	0.0610 ± 0.0297	0.3530 ± 0.0483	0.3616 ± 0.0496	0.8106 ± 0.0118 (dup)
	2000	0.0618 ± 0.0256	0.3496 ± 0.0464	0.3582 ± 0.0478	0.8116 ± 0.0101
Varying sequence length	50	0.0618 ± 0.0264	0.4578 ± 0.0452	0.4628 ± 0.0455	0.8113 ± 0.0093
	100	0.0610 ± 0.0297	0.3530 ± 0.0483	0.3616 ± 0.0496	0.8106 ± 0.0118 (dup)
	200	0.0586 ± 0.0282	0.2514 ± 0.0454	0.2634 ± 0.0480	0.8098 ± 0.0100
	500	0.0614 ± 0.0273	0.1571 ± 0.0374	0.1779 ± 0.0423	0.8135 ± 0.0100
Varying branch length scaler	0.05	0.0600 ± 0.0274	0.5530 ± 0.0610	0.5559 ± 0.0612	0.8135 ± 0.0093
	0.10	0.0608 ± 0.0282	0.4686 ± 0.0555	0.4722 ± 0.0550	0.8118 ± 0.0090
	1.00	0.0610 ± 0.0297	0.3530 ± 0.0483	0.3616 ± 0.0496	0.8106 ± 0.0118 (dup)
	10.00	0.0600 ± 0.0262	0.4280 ± 0.0412	0.4347 ± 0.0414	0.8125 ± 0.0097
	100.00	0.0599 ± 0.0270	0.7035 ± 0.0382	0.7050 ± 0.0378	0.8126 ± 0.0102
	200.00	0.0604 ± 0.0258	0.7664 ± 0.0322	0.7674 ± 0.0319	0.8109 ± 0.0098
Varying missingness parameter	0.50	0.0767 ± 0.0281	0.3818 ± 0.0430	0.3936 ± 0.0441	0.7373 ± 0.0111
	0.55	0.0659 ± 0.0265	0.3644 ± 0.0467	0.3744 ± 0.0481	0.7792 ± 0.0124
	0.60	0.0610 ± 0.0297	0.3530 ± 0.0483	0.3616 ± 0.0496	0.8106 ± 0.0118 (dup)
	0.65	0.0548 ± 0.0250	0.3340 ± 0.0449	0.3418 ± 0.0459	0.8378 ± 0.0088
	0.70	0.0471 ± 0.0250	0.3200 ± 0.0462	0.3263 ± 0.0468	0.8617 ± 0.0093
	0.75	0.0447 ± 0.0267	0.3023 ± 0.0538	0.3066 ± 0.0539	0.8789 ± 0.0048

Table B.2: **Properties of S100 simulated data from [157] and also [156].** Number of refinements is the number of refinements needed to make FastTrees binary (they can be non-binary due to identical sequences).

Sequence length	ILS	GTEE	AD	# of refinements
200	0.4575 ± 0.0601	0.5546 ± 0.0792	0.6634 ± 0.0652	2.6337 ± 3.9622
400	0.4575 ± 0.0601	0.4229 ± 0.0823	0.5894 ± 0.0653	0.8408 ± 1.5477
800	0.4575 ± 0.0601	0.3115 ± 0.0789	0.5385 ± 0.0628	0.2706 ± 0.6069
1600	0.4575 ± 0.0601	0.2258 ± 0.0735	0.5070 ± 0.0611	0.0972 ± 0.2746

Table B.3: **Properties of 200-taxon ASTRAL-II simulated data from [78] and also [157].**

ILS level (species tree height)	speciation	# of taxa	ILS	GTEE	AD
high (0.5X)	deep	200	0.68 ± 0.02	0.44 ± 0.14	0.74 ± 0.03
high (0.5X)	shallow	200	0.69 ± 0.02	0.44 ± 0.12	0.74 ± 0.03
medium (1X)	shallow	10	0.17 ± 0.06	0.19 ± 0.09	0.28 ± 0.08
medium (1X)	shallow	50	0.31 ± 0.04	0.26 ± 0.11	0.42 ± 0.08
medium (1X)	shallow	100	0.33 ± 0.02	0.26 ± 0.09	0.44 ± 0.05
medium (1X)	deep	200	0.34 ± 0.02	0.34 ± 0.12	0.47 ± 0.08
medium (1X)	shallow	200	0.34 ± 0.02	0.27 ± 0.12	0.44 ± 0.07
medium (1X)	shallow	500	0.34 ± 0.01	0.28 ± 0.11	0.45 ± 0.07
medium (1X)	shallow	1000	0.35 ± 0.01	0.30 ± 0.11	0.47 ± 0.08
low (5X)	deep	200	0.09 ± 0.01	0.28 ± 0.11	0.31 ± 0.10
low (5X)	shallow	200	0.21 ± 0.02	0.21 ± 0.13	0.33 ± 0.09

B.4.2 Software and Data availability

The scripts used to run methods and analyze the results are also available on Github: `https://github.com/molloy-lab/tree-qmc-study/tree/main/han2024wtreeqmc`. We analyzed a large number of analyses varying method parameters, so we encourage you to look at the scripts. We give a summary of commands below.

B.4.3 Gene Tree Branch Support Estimation Commands

Some data sets included gene trees with branch supported estimated with the abayes technique [5], as implemented in ith IQ-TREE [76]. For the trees without abayes support (i.e., the ASTRAL-II data sets with 10, 50, 100, 500, and 1000 taxan), we estimated abayes support with 10, 50, 100, 500, and 1000 taxa using the following command:

```
iqtree2 \
    -s <input alignment> \
    -te <input gene tree> \
    -m GTR+G \
    -abayes \
    -pre <prefix for output file> \
    -T 1
```

Note that IQtree refines polytomies and gives them no support value. The default in weighted ASTRAL/ASTER [156], weighted ASTRID [65], and wTREE-QMC is to have these refined branches have no branch support when running the weighted versions of these methods. However, these (arbitrary?) refinements would be part of the input for the unweighted version of TREE-QMC (option: `--fast` or `-w f`).

B.4.4 Species Tree Estimation Commands

TREE-QMC.

TREE-QMC version 3.0.4 (commit b5bfe82) was run using the following command:

```
tree-qmc \
    -w <weighting scheme for quartets> \
    <branch support scheme> \
    <normalization scheme> \
    -i [input gene trees] \
    -o [output species tree] \
    &> [output log file]
```

The best version of TREE-QMC (denoted **TREE-QMC-wh**) uses the hybrid weighting scheme for quartets (option `-w h` or `--hybrid`) and the n2 normalization scheme for artificial taxa (option `--norm-atax 2`). If using branch support weighting, the branch support scheme set based on the method used to estimate support in the input gene trees, specifically `--lrt` for sh branch support (equivalent to `-n 0 -x 1 -d 0`), `--bootstrap` for bootstrap support (equivalent to `-n 0 -x 100 -d 0`), and `--bayes` for abayes branch support (equivalent to `-n 0.333 -x 1 -d 0.333`). We ran TREE-QMC with a variety of normalization and weighting schemes. We also ran TREE-QMC using the original algorithm method (no weighting) from [45] (option `--fast` or `-w f`) with the n2 normalization scheme for artificial taxa (option: `--norm-atax 2`); this method (denoted **TREE-QMC-n2**) does not support polytomies in the input gene trees and thus it internally refines any polytomies in the gene trees at random prior to species tree estimation. Note that we built TREE-QMC from source on the EPYC-7313 compute nodes.

ASTRAL-IV.

ASTRAL-IV [156] was downloaded from `https://github.com/chaoszhang/ASTER` (commit 0c61214) and built from source on an EPYC-7313 compute node. For Asteroid data sets, ASTRAL-IV (no weighting) v1.16.3.4 was run using the following command:

```
astral \
    -u 0 \
    -i [input gene trees] \
    -o [output species tree] \
    &> [output log file]
```

The `-u 0` option turns of branch support calculations to enable a fair runtime comparison. For the other data sets, weighted ASTRAL-IV (hybrid weighting) v1.16.3.4 was run using the following command:

```
astral-hybrid \
    --bayes \
    -u 0  \
    -i [input gene trees] \
    -o [output species tree] \
    &> [output log file]
```

The `--bayes` option was replaced with `--bootstrap` when branch support was estimated with bootstrapping. For the scalability study, we added flags `-t 1` or `-t 16` to indicate 1 or 16 threads, respectively.

ASTRID.

We were unable to build wASTRID from source on an EPYC-7313 compute node, so we downloaded wASTRID version 0.0.7 [156] from `https://github.com/RuneBlaze/internode/releases/tag/v0.0.7-snapshot`. wASTRID was run with the following command:

```
wastrid \
    -b <support min>-<support max>  \
    -i [input gene trees] \
    -o [output species tree] \
    &> [output log file]
```

with minimum and maximum support set based on the input gene trees as previously noted. The default mode is support weighting (i.e., `-m support`). To use length weighting, we added `-m n-length` and removed the branch support information. To use no weighting, we added `-m internode` or `--preset vanilla`) and removed the branch support information.

Asteroid.

Asteroid [86] was downloaded from `https://github.com/BenoitMorel/Asteroid` (commit f69b761) and built from source (without mpi) on an EPYC-7313 compute node. Astroid was run with the following command:

```
asteroid \
    -i [input gene trees] \
    -p [prefix for output species tree] \
    &> [output log file]
```

B.4.5 Species Tree Branch Support Estimation Commands

Quartet score and branch support with hybrid quartet weights were computed using the following command:

```
astral-hybrid \
    --scoring -u 2 -t 16 \
    -i [input gene trees] \
    -c [input species tree] \
    -o [output species tree annotated with support] \
    &> [output log file]
```

For analyses of simulated data, we replaced `astral-hybrid` with `astral` to perform the computation with unweighted quartets. For the avian biological data sets, we used astral-hybrid with the `--mode` flag to change the weighting scheme. For the plant biological data sets, ASTRAL-III (v5.7.7) was run with the following command:

```
java -Xmx36G -D"java.library.path=<path to ASTRAL>/lib" \
    -jar <path to ASTRAL>/astral.5.7.7.jar \
    -t2 \
    -q [input species tree] \
    -i [input gene trees] \
    -o [output scored species tree] &> [output log file]
```

because this version handles explicitly outputs EN and gives warnings for low EN, which is useful information when gene trees have high rates of missing taxa.

B.4.6 Scalability Study

We evaluated scalability for increasing numbers of taxa using the ASTRAL-II data sets [78] with 1000 genes, reporting the wall-clock time (i.e., the amount of time that the user waits from the beginning to the end of the computation). These computational experiments were performed on compute nodes outfitted with 32 AMD EPYC-7313 cores and 2TB of RAM. There are 28 nodes with these specifications on the CBCB cluster, we requested exclusive access of these compute nodes via slurm scheduler when performing timing experiments. We ran all methods were run with 1 thread and limited to 64 GB of memory. ASTRAL-IV-wh was also evaluated with 16 threads.

B.4.7 Replicates Excluded from ASTRAL-II Data

All comparisons are made on the same set of replicate data sets. As in [78], we excluded replicate data set from analyses whenever more than half of the 1000 gene trees had the majority of their branches unresolved (due to identical sequences). Specifically, we excluded

- 1 replicate (# 41) from the 10-taxon model condition (50, 200, and 1000 genes)
- 2 replicates (# 21, 41) from the 50-taxon model condition (50, 200, and 1000 genes)
- 2 replicates (# 8, 47) from the 100-taxon model condition (50, 200, and 1000 genes)
- 3 replicates (# 8, 15, 49) from the 200 taxon model condition with very high ILS and shallow speciation (50, 200, and 1000 genes)

We also excluded replicates because the concatenation trees (downloaded from [78]) were missing on the following data sets:

- 1 replicate (# 12) from the 10-taxon, 1000-gene model condition
- 1 replicate (# 27) from the 50-taxon model condition (50, 200, and 1000 genes)

Scalability study. For the scalability study, 3 additional replicates were excluded because ASTER-IV-wh (single-threaded) failed to complete within 20 hours for the following data sets:

- 1 replicate (# 21) from the 200 taxon, 1000-gene model condition with very high ILS and deep speciation
- 2 replicates (# 6, 38) from the 1000-taxon, 1000-gene model condition

Lastly, the concatenation tree was also missing from the 1000-taxon, 1000-gene model condition (all 50 replicates) so we don't show results for concatenation on this model condition.

B.4.8 Statistical Tests

We tested for significant differences between methods using two-sided, paired Wilcoxon signed-rank tests, as implemented in the R coin library. We tried two different ways of correcting for ties. The Wilcoxon method of correcting for ties was run with the following command:

```
pval <-  pvalue(wilcoxsign_test(mthd1 ~ mthd2,
        zero.method="Wilcoxon",
        distribution = "exact",
        paired = TRUE,
        alternative = "two.sided",
        conf.int = TRUE))
```

The Pratt method of correcting for ties was run with the following command:

```
pval <- pvalue(wilcoxsign_test(mthd1 ~ mthd2,
        zero.method="Pratt",
        distribution = "exact",
        paired = TRUE,
```

```
alternative = "two.sided",
conf.int = TRUE))
```

We took the **higher** of the two p-values to be conservative. We say the resulting p-value was significant after Bonferroni multiple comparisons correction if it was less than 0.05 divided by the number of tests for an experiment (we consider analyses of the Asteroid, S100, and ASTRAL-II data sets as separate experiments).

B.5 Additional Results on Simulated Data Sets

B.5.1 Additional Results on Asteroid data

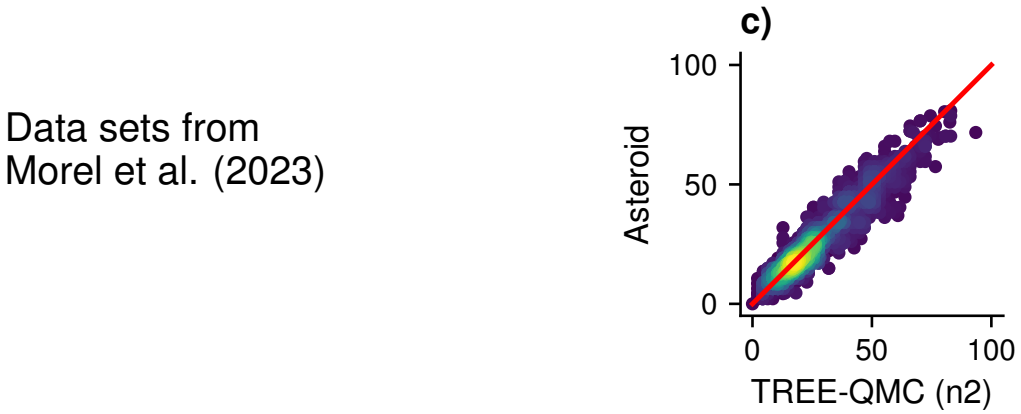


Figure B.2: **Dot plot comparing Asteroid and TREE-QMC (n2) on Morel data sets.** Continuation of figure in main text; note that the other data sets do not have missing data, so Asteroid and ASTRID should give the same results.

Table B.4: **Species tree (FP) error rate for Asteroid data.** Mean error rate is given across 50 replicates for each method.

		ASTRID	ASTRAL IV	Asteroid	TREE-QMC n2	TREE-QMC n1	TREE-QMC n0	TREE-QMC n2-shared
Varying population size	10	0.21200	0.17660	0.14430	**0.14370**	0.14630	0.17560	0.21490
	50000000	0.24600	0.20380	0.18130	**0.18120**	0.18190	0.20710	0.24580
	100000000	0.29660	0.26170	0.23150	**0.22390**	0.22740	0.25870	0.32100
	500000000	0.47530	0.46210	0.43450	**0.41560**	0.42080	0.45200	0.51840
	1000000000	0.58470	0.58810	**0.55280**	0.55330	0.55430	0.60310	0.66180
Varying number of taxa	25	0.26910	0.27540	0.22540	0.22280	**0.22160**	0.26480	0.27030
	75	0.24390	0.20890	0.18310	**0.18070**	0.18420	0.20710	0.26900
	50	0.24600	0.20380	0.18130	**0.18120**	0.18190	0.20710	0.24580 (dup)
	100	0.25070	0.21710	0.19650	**0.18560**	0.18980	0.22000	0.28340
	125	0.24670	0.21120	0.18480	**0.17650**	0.18010	0.20990	0.27040
	150	0.25580	0.21690	0.18680	**0.18160**	0.18600	0.21720	0.28880
Varying number of genes	250	0.47080	0.43140	**0.35980**	0.36450	0.36500	0.42440	0.46060
	500	0.34820	0.31340	0.26010	**0.25260**	0.25960	0.29610	0.33560
	1000	0.24600	0.20380	0.18130	**0.18120**	0.18190	0.20710	0.24580 (dup)
	2000	0.18170	0.14940	0.13110	**0.12050**	0.12630	0.14860	0.18800
Varying sequence length	50	0.29180	0.29470	**0.24580**	0.25020	0.25440	0.29120	0.34300
	100	0.24600	0.20380	0.18130	**0.18120**	0.18190	0.20710	0.24580 (dup)
	200	0.23060	0.17110	0.16430	**0.15320**	0.15450	0.17300	0.20780
	500	0.19740	0.14640	0.13230	0.11520	**0.11170**	0.13870	0.15590
Varying branch length scaler	0.05	0.51020	0.48550	0.45060	0.45540	**0.44880**	0.48060	0.58390
	0.10	0.37830	0.34000	**0.29280**	0.30920	0.30210	0.35010	0.42360
	1.00	0.24600	0.20380	0.18130	**0.18120**	0.18190	0.20710	0.24580 (dup)
	10.00	0.26130	0.24510	0.20720	**0.19000**	0.20270	0.23790	0.26610
	100.00	0.42680	0.42550	**0.35790**	0.36260	0.37720	0.42630	0.51750
	200.00	0.50080	0.49530	**0.45620**	0.48220	0.48530	0.52540	0.61540
Varying missingness parameter	0.50	0.13360	0.11790	**0.09870**	0.10280	0.10490	0.12080	0.14070
	0.55	0.17320	0.17190	0.14510	**0.14020**	0.14300	0.17010	0.20660
	0.60	0.24600	0.20380	0.18130	**0.18120**	0.18190	0.20710	0.24580 (dup)
	0.65	0.38490	0.32350	0.28050	**0.24730**	0.25690	0.30250	0.35640
	0.70	0.60670	0.49840	0.43100	**0.41460**	0.41930	0.45520	0.53320
	0.75	0.82050	0.73420	0.62240	0.61020	**0.60340**	0.65720	0.72410

Table B.5: **Species tree (FN) error rate for Asteroid data.** Mean error rate is given across 50 replicates for each method.

		ASTRID	ASTRAL IV	Asteroid	TREE-QMC n2	TREE-QMC n1	TREE-QMC n0	TREE-QMC n2-shared
Varying population size	10	0.21200	0.17660	**0.14430**	0.15750	0.16090	0.18860	0.22640
	50000000	0.24600	0.20380	**0.18130**	0.19620	0.19620	0.22040	0.25830
	100000000	0.29660	0.26170	**0.23150**	0.23960	0.24340	0.27400	0.33400
	500000000	0.47530	0.46210	0.43450	**0.42430**	0.42980	0.46080	0.52430
	1000000000	0.58470	0.58810	**0.55280**	0.56130	0.56170	0.61020	0.66890
Varying number of taxa	25	0.26910	0.27540	**0.22540**	0.24180	0.24000	0.28090	0.28820
	75	0.24390	0.20890	**0.18310**	0.19560	0.19970	0.22190	0.28420
	50	0.24600	0.20380	**0.18130**	0.19620	0.19620	0.22040	0.25830 (dup)
	100	0.25070	0.21710	**0.19650**	0.20310	0.20640	0.23570	0.29770
	125	0.24670	0.21120	**0.18480**	0.19430	0.19900	0.22590	0.28560
	150	0.25580	0.21690	**0.18680**	0.19820	0.20250	0.23100	0.30290
Varying number of genes	250	0.47080	0.43140	**0.35980**	0.41230	0.41310	0.46560	0.50060
	500	0.34820	0.31340	**0.26010**	0.28310	0.29040	0.32360	0.36020
	1000	0.24600	0.20380	**0.18130**	0.19620	0.19620	0.22040	0.25830 (dup)
	2000	0.18170	0.14940	0.13110	**0.13060**	0.13570	0.15830	0.19700
Varying sequence length	50	0.29180	0.29470	**0.24580**	0.26320	0.26790	0.30100	0.35390
	100	0.24600	0.20380	**0.18130**	0.19620	0.19620	0.22040	0.25830 (dup)
	200	0.23060	0.17110	**0.16430**	0.17530	0.17570	0.19280	0.22720
	500	0.19740	0.14640	**0.13230**	0.13740	0.13400	0.15960	0.17570
Varying branch length scaler	0.05	0.51020	0.48550	**0.45060**	0.46850	0.46300	0.48980	0.59190
	0.10	0.37830	0.34000	**0.29280**	0.32260	0.31570	0.36080	0.43360
	1.00	0.24600	0.20380	**0.18130**	0.19620	0.19620	0.22040	0.25830 (dup)
	10.00	0.26130	0.24510	**0.20720**	0.21150	0.22260	0.25570	0.28380
	100.00	0.42680	0.42550	**0.35790**	0.37230	0.38770	0.43450	0.52770
	200.00	0.50080	0.49530	**0.45620**	0.49320	0.49490	0.53280	0.62340
Varying missingness parameter	0.50	0.13360	0.11790	**0.09870**	0.10470	0.10680	0.12300	0.14260
	0.55	0.17320	0.17190	**0.14510**	0.14720	0.14940	0.17450	0.21280
	0.60	0.24600	0.20380	**0.18130**	0.19620	0.19620	0.22040	0.25830 (dup)
	0.65	0.38490	0.32350	**0.28050**	0.29030	0.29970	0.33970	0.38820
	0.70	0.60670	0.49840	**0.43100**	0.48830	0.49130	0.51890	0.58580
	0.75	0.82050	0.73420	**0.62240**	0.69620	0.69020	0.72750	0.77880

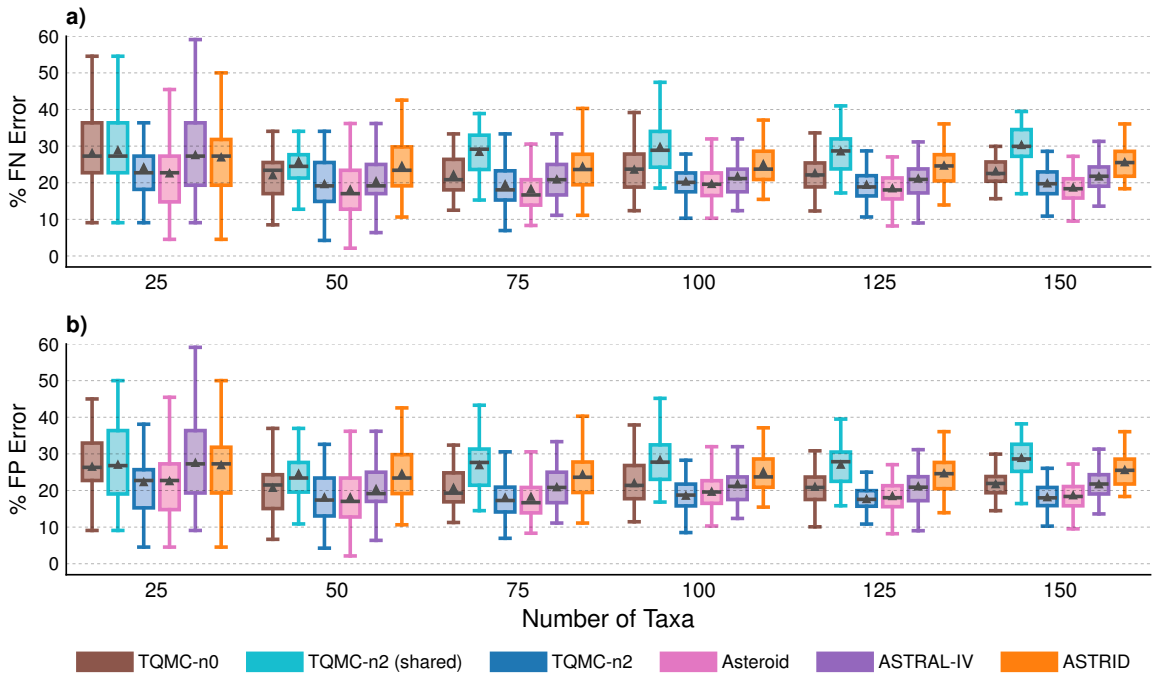


Figure B.3: **Species tree error for Asteroid data with varying numbers of taxa.** Percent species tree (FN or FP) error is shown on *y*-axis for 50 replicates in each model condition. Bars represent medians, triangles represent means, outliers are not shown.

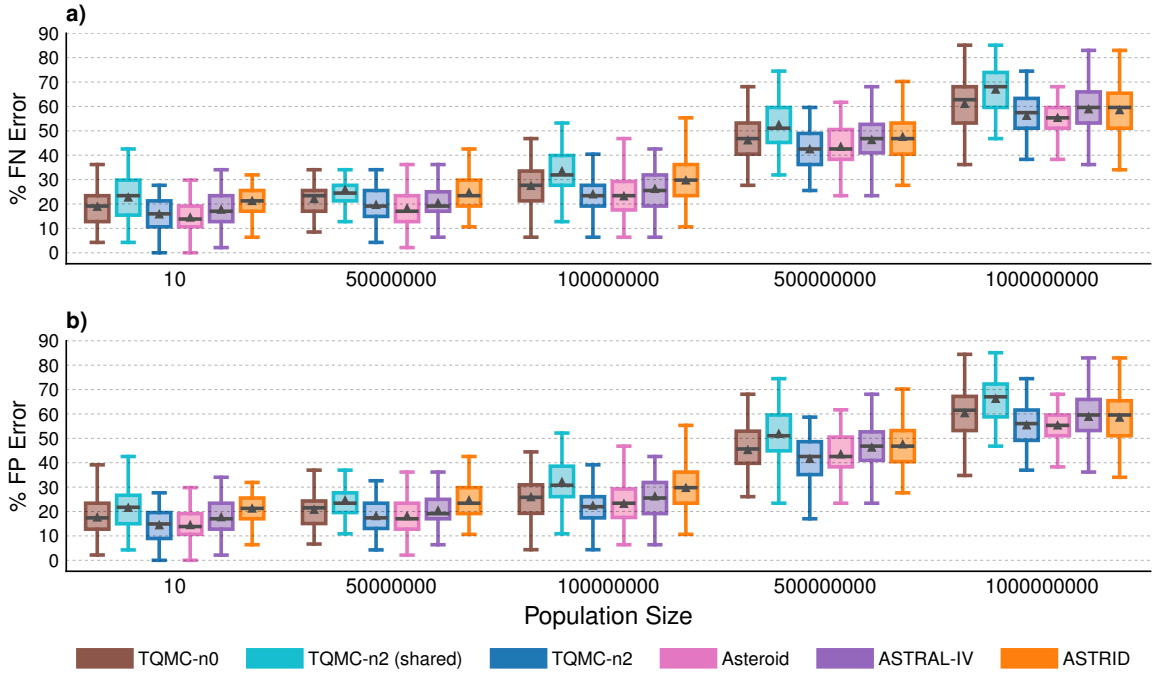


Figure B.4: **Species tree error for Asteroid data with varying population size.** Percent species tree (FN or FP) error is shown on *y*-axis for 50 replicates in each model condition. Bars represent medians, triangles represent means, outliers are not shown.

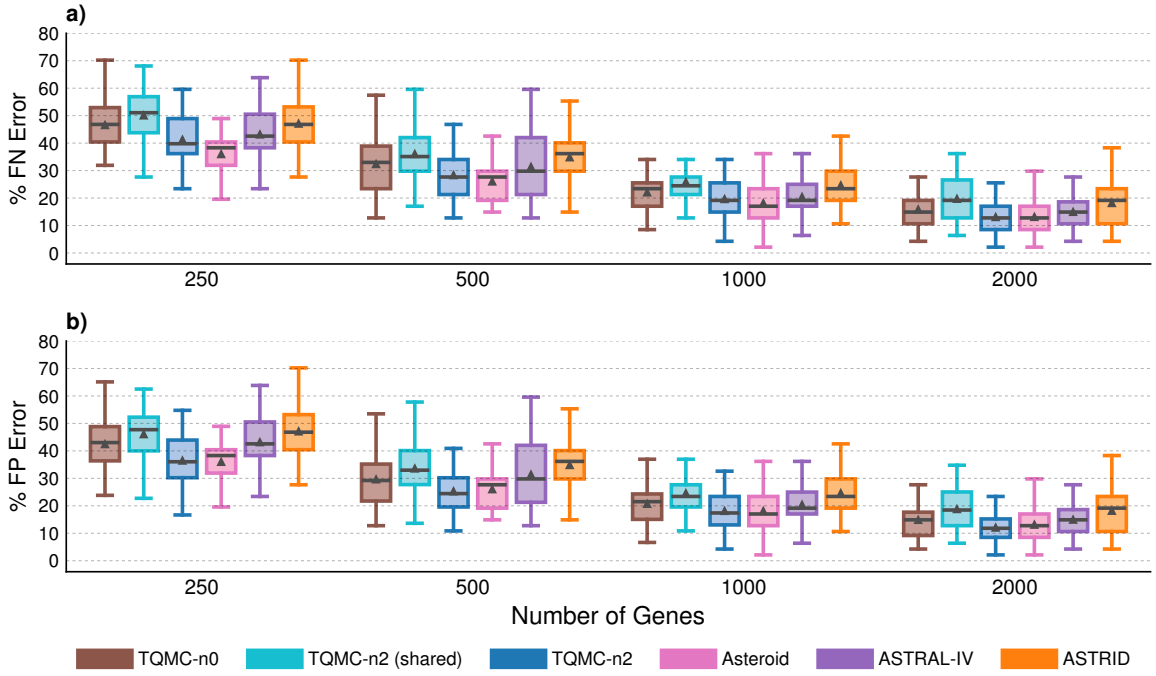


Figure B.5: **Species tree error for Asteroid data with varying numbers of genes.** Percent species tree (FN or FP) error is shown on *y*-axis for 50 replicates in each model condition. Bars represent medians, triangles represent means, outliers are not shown.

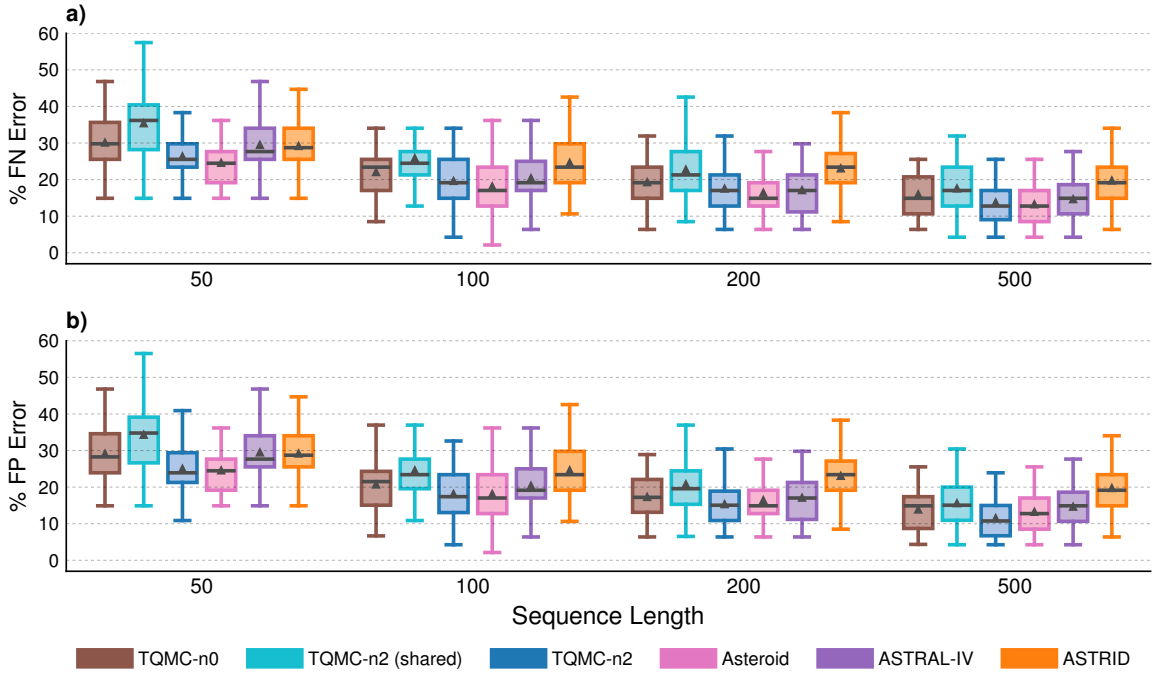


Figure B.6: **Species tree error for Asteroid data with varying sequence length.** Percent species tree (FN or FP) error is shown on *y*-axis for 50 replicates in each model condition. Bars represent medians, triangles represent means, outliers are not shown.

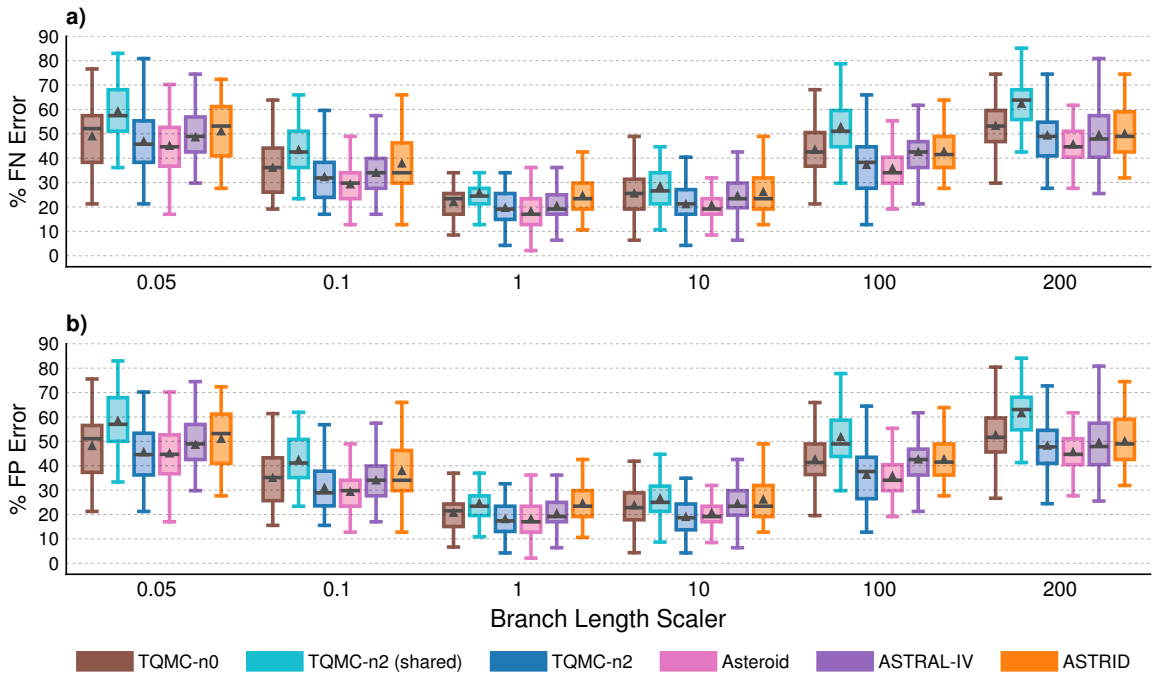


Figure B.7: **Species tree error for Asteroid data with varying branch length scalar.** Percent species tree (FN or FP) error is shown on *y*-axis for 50 replicates in each model condition. Bars represent medians, triangles represent means, outliers are not shown.

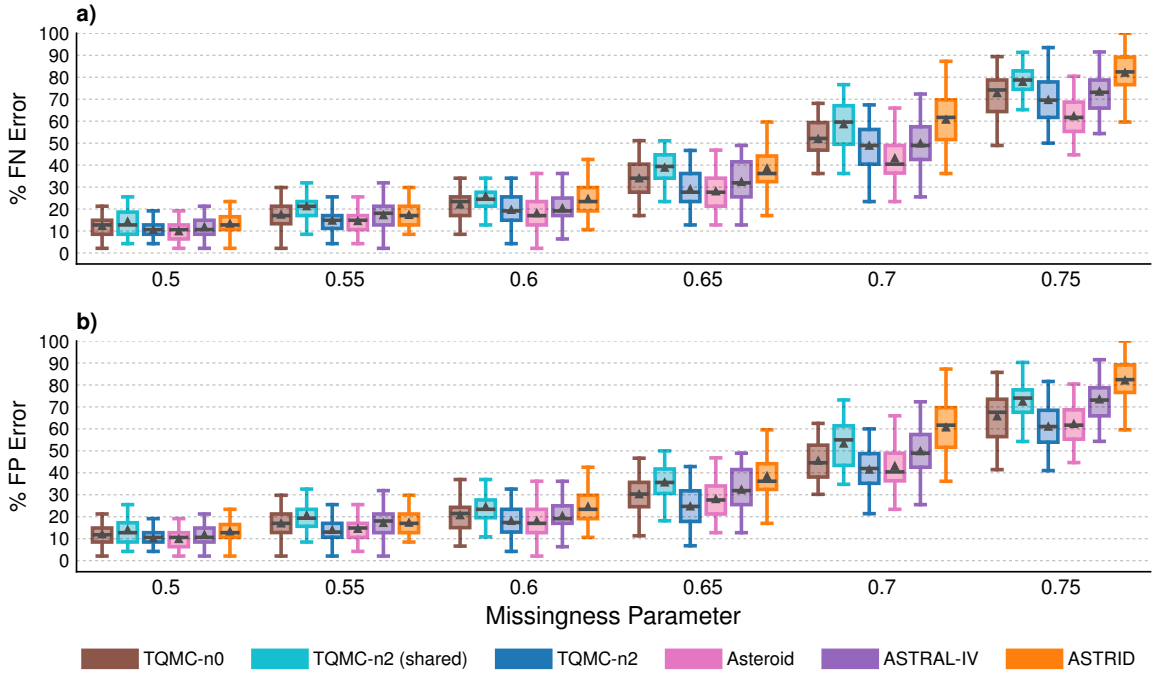


Figure B.8: **Species tree error for Asteroid data with varying missingness parameter.** Percent species tree (FN or FP) error is shown on *y*-axis for 50 replicates in each model condition. Bars represent medians, triangles represent means, outliers are not shown.

Table B.6: **Testing for differences between TREE-QMC-n2 vs Asteroid on the Asteroid data.** BET is the number of replicates for which TREE-QMC-n2 has lower species tree error and thus is better than Asteroid, and WOR is the opposite. Significance is evaluated using Wilcoxon signed-rank tests on the error rates. The symbols *, **, ***, ****, ***** indicate significance at $p < 0.5$, 0.005, and so on. MC indicates significance after Bonferroni correction, i.e., $p < 0.05$ / 312 = 2e-04 for the 312 tests made on the Asteroid data.

TREE-QMC-n2 vs Asteroid									
		BET	WOR	TIE	p-val	sig	MC	note	
False negative error rate									
Varying population size	10	15	24	11	0.04	*		ASTEROID better	
	50000000	12	27	11	0.01	*		ASTEROID better	
	100000000	20	20	10	0.4				
	500000000	26	20	4	0.3				
	1000000000	16	25	9	0.4				
Varying number of taxa	25	13	18	19	0.3				
	75	14	27	9	0.008	*		ASTEROID better	
	50	12	27	11	0.01	*		ASTEROID better	(dup)
	100	13	28	9	0.1				
	125	13	29	8	0.003	**		ASTEROID better	
	150	15	32	3	0.002	**		ASTEROID better	
Varying number of genes	250	12	36	2	3e-06	*****	MC	ASTEROID better	
	500	15	28	7	0.004	**		ASTEROID better	
	1000	12	27	11	0.01	*		ASTEROID better	(dup)
	2000	13	15	22	0.8				
Varying sequence length	50	13	28	9	0.02	*		ASTEROID better	
	100	12	27	11	0.01	*		ASTEROID better	(dup)
	200	13	24	13	0.03	*		ASTEROID better	
	500	15	20	15	0.3				
Varying branch length scaler	0.05	14	26	10	0.07				
	0.1	9	34	7	7e-04	**		ASTEROID better	
	1	12	27	11	0.01	*		ASTEROID better	(dup)
	10	18	18	14	0.7				
	100	21	25	4	0.3				
	200	12	29	9	0.002	**		ASTEROID better	
Varying missingness parameter	0.5	14	23	13	0.3				
	0.55	15	17	18	0.8				
	0.6	12	27	11	0.01	*		ASTEROID better	(dup)
	0.65	15	24	11	0.08				
	0.7	10	32	8	5e-06	*****	MC	ASTEROID better	
	0.75	7	40	3	5e-09	*****	MC	ASTEROID better	

Table B.7: **Testing for differences between TREE-QMC-n2 vs Asteroid on the Asteroid data (continued).**

TREE-QMC-n2 vs Asteroid									
		BET	WOR	TIE	p-val	sig	MC	note	
False positive error rate									
Varying population size	10	20	23	7	1				
	50000000	24	23	3	0.7				
	100000000	30	19	1	0.2				
	500000000	30	18	2	0.04	*			
	1000000000	22	24	4	0.8				
Varying number of taxa	25	20	16	14	0.7				
	75	23	25	2	0.8				
	50	24	23	3	0.7				(dup)
	100	29	20	1	0.03	*			
	125	30	19	1	0.01	*			
	150	31	19	0	0.1				
Varying number of genes	250	28	22	0	0.9				
	500	26	23	1	0.3				
	1000	24	23	3	0.7				(dup)
	2000	21	13	16	0.09				
Varying sequence length	50	19	27	4	0.3				
	100	24	23	3	0.7				(dup)
	200	27	19	4	0.2				
	500	32	11	7	8e-04	**			
Varying branch length scaler	0.05	26	23	1	0.9				
	0.1	16	29	5	0.04	*		ASTEROID better	
	1	24	23	3	0.7				(dup)
	10	31	16	3	0.02	*			
	100	25	23	2	0.7				
	200	17	29	4	0.02	*		ASTEROID better	
Varying missingness parameter	0.5	17	23	10	0.3				
	0.55	22	16	12	0.3				
	0.6	24	23	3	0.7				(dup)
	0.65	39	11	0	1e-05	****	MC		
	0.7	33	17	0	0.07				
	0.75	28	22	0	0.2				

Table B.8: **Testing for differences between TREE-QMC-n2 vs ASTRAL-IV on the Asteroid data.** BET is the number of replicates for which TREE-QMC-n2 has lower species tree error and thus is better than ASTRAL-IV, and WOR is the opposite. Significance is evaluated using Wilcoxon signed-rank tests on the error rates. The symbols *, **, ***, ****, ***** indicate significance at $p < 0.5$, 0.005, and so on. MC indicates significance after Bonferroni correction, i.e., $p < 0.05$ / 312 = 2e-04 for the 312 tests made on the Asteroid data.

TREE-QMC-n2 vs ASTRAL-IV								
		BET	WOR	TIE	p-val	sig	MC	note
False negative error rate								
Varying population size	10	30	13	7	0.005	*		
	50000000	24	19	7	0.3			
	100000000	26	16	8	0.02	*		
	500000000	32	12	6	5e-04	***		
	1000000000	32	16	2	0.005	*		
Varying number of taxa	25	29	9	12	8e-04	**		
	75	26	18	6	0.1			
	50	24	19	7	0.3			(dup)
	100	30	15	5	0.01	*		
	125	31	16	3	0.003	**		
	150	38	6	6	3e-06	*****	MC	
Varying number of genes	250	28	16	6	0.1			
	500	30	14	6	0.006	*		
	1000	24	19	7	0.3			(dup)
	2000	34	9	7	0.001	**		
Varying sequence length	50	29	11	10	0.001	**		
	100	24	19	7	0.3			(dup)
	200	17	26	7	0.4			
	500	21	17	12	0.3			
Varying branch length scaler	0.05	27	17	6	0.09			
	0.1	27	15	8	0.05			
	1	24	19	7	0.3			(dup)
	10	29	13	8	6e-04	**		
	100	37	10	3	2e-06	*****	MC	
	200	25	21	4	0.6			
Varying missingness parameter	0.5	27	11	12	0.02	*		
	0.55	31	9	10	5e-04	***		
	0.6	24	19	7	0.3			(dup)
	0.65	29	7	14	2e-04	***	MC	
	0.7	22	20	8	0.4			
	0.75	28	15	7	0.01	*		

Table B.9: **Testing for differences between TREE-QMC-n2 vs ASTRAL-IV on the Asteroid data (continued).**

TREE-QMC-n2 vs ASTRAL-IV									
		BET	WOR	TIE	p-val	sig	MC	note	
False positive error rate									
Varying population size	10	36	12	2	3e-05	****	MC		
	50000000	28	18	4	0.009	*			
	100000000	36	14	0	5e-05	***	MC		
	500000000	40	10	0	1e-05	****	MC		
	1000000000	32	16	2	2e-04	***			
Varying number of taxa	25	36	7	7	8e-06	****	MC		
	75	33	16	1	3e-04	***			
	50	28	18	4	0.009	*			(dup)
	100	39	10	1	4e-08	*****	MC		
	125	41	9	0	2e-08	*****	MC		
	150	46	4	0	4e-13	*****	MC		
Varying number of genes	250	40	10	0	2e-08	*****	MC		
	500	39	11	0	3e-07	*****	MC		
	1000	28	18	4	0.009	*			(dup)
	2000	39	8	3	2e-06	*****	MC		
Varying sequence length	50	36	10	4	2e-05	****	MC		
	100	28	18	4	0.009	*			(dup)
	200	29	18	3	0.009	*			
	500	33	14	3	2e-04	***			
Varying branch length scaler	0.05	32	16	2	0.002	**			
	0.1	29	15	6	0.002	**			
	1	28	18	4	0.009	*			(dup)
	10	37	12	1	5e-07	*****	MC		
	100	41	9	0	8e-08	*****	MC		
	200	28	20	2	0.2				
Varying missingness parameter	0.5	28	11	11	0.009	*			
	0.55	36	9	5	3e-05	****	MC		
	0.6	28	18	4	0.009	*			(dup)
	0.65	45	4	1	1e-12	*****	MC		
	0.7	43	7	0	8e-09	*****	MC		
	0.75	45	5	0	3e-10	*****	MC		

Table B.10: **Testing for differences between TREE-QMC-n2 vs ASTRID on the Asteroid data.** BET is the number of replicates for which TREE-QMC-n2 has lower species tree error and thus is better than ASTRID, and WOR is the opposite. Significance is evaluated using Wilcoxon signed-rank tests on the error rates. The symbols *, **, ***, ****, ***** indicate significance at $p < 0.5$, 0.005, and so on. MC indicates significance after Bonferroni correction, i.e., $p < 0.05$ / 312 = 2e-04 for the 312 tests made on the Asteroid data.

TREE-QMC-n2 vs ASTRID								
		BET	WOR	TIE	p-val	sig	MC	note
False negative error rate								
Varying population size	10	40	5	5	3e-09	*****	MC	
	50000000	35	10	5	1e-06	*****	MC	
	100000000	38	8	4	1e-06	*****	MC	
	500000000	34	11	5	5e-05	****	MC	
	1000000000	30	15	5	0.05			
Varying number of taxa	25	25	14	11	0.04	*		
	75	36	9	5	5e-07	*****	MC	
	50	35	10	5	1e-06	*****	MC	(dup)
	100	40	8	2	6e-09	*****	MC	
	125	45	4	1	3e-11	*****	MC	
	150	48	2	0	5e-13	*****	MC	
Varying number of genes	250	38	9	3	1e-05	****	MC	
	500	40	7	3	6e-08	*****	MC	
	1000	35	10	5	1e-06	*****	MC	(dup)
	2000	36	8	6	3e-07	*****	MC	
Varying sequence length	50	30	17	3	0.007	*		
	100	35	10	5	1e-06	*****	MC	(dup)
	200	41	6	3	9e-08	*****	MC	
	500	42	4	4	6e-10	*****	MC	
Varying branch length scaler	0.05	30	14	6	0.005	*		
	0.1	35	10	5	1e-05	****	MC	
	1	35	10	5	1e-06	*****	MC	(dup)
	10	34	13	3	3e-05	****	MC	
	100	34	14	2	7e-04	**		
	200	26	23	1	0.6			
Varying missingness parameter	0.5	29	10	11	4e-04	***		
	0.55	32	15	3	0.003	**		
	0.6	35	10	5	1e-06	*****	MC	(dup)
	0.65	35	9	6	4e-08	*****	MC	
	0.7	43	3	4	2e-11	*****	MC	
	0.75	46	1	3	7e-14	*****	MC	

Table B.11: **Testing for differences between TREE-QMC-n2 vs ASTRID on the Asteroid data (continued).**

TREE-QMC-n2 vs ASTRID								
		BET	WOR	TIE	p-val	sig	MC	note
False positive error rate								
Varying population size	10	44	4	2	7e-12	*****	MC	
	50000000	39	9	2	1e-08	*****	MC	
	100000000	40	8	2	3e-08	*****	MC	
	500000000	38	10	2	5e-06	****	MC	
	1000000000	34	15	1	0.006	*		
Varying number of taxa	25	31	12	7	0.002	**		
	75	42	7	1	9e-10	*****	MC	
	50	39	9	2	1e-08	*****	MC	(dup)
	100	43	6	1	1e-11	*****	MC	
	125	47	3	0	2e-13	*****	MC	
	150	49	1	0	9e-15	*****	MC	
Varying number of genes	250	45	5	0	1e-10	*****	MC	
	500	43	7	0	1e-10	*****	MC	
	1000	39	9	2	1e-08	*****	MC	(dup)
	2000	40	6	4	1e-09	*****	MC	
Varying sequence length	50	31	17	2	3e-04	***		
	100	39	9	2	1e-08	*****	MC	(dup)
	200	45	5	0	1e-10	*****	MC	
	500	45	2	3	4e-12	*****	MC	
Varying branch length scaler	0.05	35	14	1	3e-04	***		
	0.1	38	10	2	2e-07	*****	MC	
	1	39	9	2	1e-08	*****	MC	(dup)
	10	39	10	1	1e-07	*****	MC	
	100	35	14	1	2e-04	***		
	200	27	23	0	0.3			
Varying missingness parameter	0.5	31	10	9	2e-04	***		
	0.55	33	15	2	9e-04	**		
	0.6	39	9	2	1e-08	*****	MC	(dup)
	0.65	45	5	0	4e-13	*****	MC	
	0.7	49	1	0	4e-15	*****	MC	
	0.75	49	1	0	4e-15	*****	MC	

Table B.12: **Testing for differences between TREE-QMC-n2 vs TREE-QMC-n2-shared on the Asteroid data.** BET is the number of replicates for which TREE-QMC-n2 has lower species tree error and thus is better than TREE-QMC-n2-shared, and WOR is the opposite. Significance is evaluated using Wilcoxon signed-rank tests on the error rates. The symbols *, **, ***, ****, ***** indicate significance at $p < 0.5$, 0.005, and so on. MC indicates significance after Bonferroni correction, i.e., $p < 0.05$ / 312 = 2e-04 for the 312 tests made on the Asteroid data.

TREE-QMC-n2 vs TREE-QMC-n2-shared								
		BET	WOR	TIE	p-val	sig	MC	note
False negative error rate								
Varying population size	10	41	5	4	1e-09	*****	MC	
	50000000	37	7	6	5e-08	*****	MC	
	100000000	45	1	4	2e-13	*****	MC	
	500000000	44	3	3	2e-10	*****	MC	
	1000000000	46	4	0	4e-11	*****	MC	
Varying number of taxa	25	28	9	13	1e-04	***	MC	
	75	46	2	2	1e-12	*****	MC	
	50	37	7	6	5e-08	*****	MC	(dup)
	100	50	0	0	2e-15	*****	MC	
	125	50	0	0	2e-15	*****	MC	
	150	50	0	0	2e-15	*****	MC	
Varying number of genes	250	41	5	4	4e-10	*****	MC	
	500	41	6	3	3e-10	*****	MC	
	1000	37	7	6	5e-08	*****	MC	(dup)
	2000	41	4	5	8e-11	*****	MC	
Varying sequence length	50	41	3	6	2e-11	*****	MC	
	100	37	7	6	5e-08	*****	MC	(dup)
	200	37	7	6	9e-08	*****	MC	
	500	40	4	6	4e-08	*****	MC	
Varying branch length scaler	0.05	49	1	0	5e-15	*****	MC	
	0.1	48	2	0	1e-13	*****	MC	
	1	37	7	6	5e-08	*****	MC	(dup)
	10	43	6	1	3e-10	*****	MC	
	100	48	2	0	2e-14	*****	MC	
	200	46	4	0	1e-11	*****	MC	
Varying missingness parameter	0.5	34	7	9	2e-06	*****	MC	
	0.55	39	5	6	1e-09	*****	MC	
	0.6	37	7	6	5e-08	*****	MC	(dup)
	0.65	45	5	0	1e-10	*****	MC	
	0.7	44	1	5	2e-13	*****	MC	
	0.75	41	6	3	4e-09	*****	MC	

Table B.13: **Testing for differences between TREE-QMC-n2 vs TREE-QMC-n2-shared on the Asteroid data (continued).**

TREE-QMC-n2 vs TREE-QMC-n2-shared								
		BET	WOR	TIE	p-val	sig	MC	note
False positive error rate								
Varying population size	10	42	5	3	7e-10	*****	MC	
	50000000	39	7	4	1e-08	*****	MC	
	100000000	46	1	3	4e-14	*****	MC	
	500000000	45	4	1	2e-10	*****	MC	
	1000000000	46	4	0	3e-11	*****	MC	
Varying number of taxa	25	28	11	11	2e-04	***		
	75	46	3	1	7e-13	*****	MC	
	50	39	7	4	1e-08	*****	MC	(dup)
	100	50	0	0	2e-15	*****	MC	
	125	50	0	0	2e-15	*****	MC	
	150	50	0	0	2e-15	*****	MC	
Varying number of genes	250	41	8	1	8e-10	*****	MC	
	500	45	5	0	9e-11	*****	MC	
	1000	39	7	4	1e-08	*****	MC	(dup)
	2000	42	4	4	3e-10	*****	MC	
Varying sequence length	50	42	6	2	1e-11	*****	MC	
	100	39	7	4	1e-08	*****	MC	(dup)
	200	38	7	5	1e-07	*****	MC	
	500	40	6	4	1e-08	*****	MC	
Varying branch length scaler	0.05	49	1	0	4e-15	*****	MC	
	0.1	48	2	0	6e-14	*****	MC	
	1	39	7	4	1e-08	*****	MC	(dup)
	10	44	6	0	6e-11	*****	MC	
	100	49	1	0	1e-14	*****	MC	
	200	47	3	0	9e-12	*****	MC	
Varying missingness parameter	0.5	34	7	9	3e-06	*****	MC	
	0.55	38	6	6	2e-09	*****	MC	
	0.6	39	7	4	1e-08	*****	MC	(dup)
	0.65	46	4	0	1e-11	*****	MC	
	0.7	46	3	1	5e-13	*****	MC	
	0.75	44	6	0	2e-10	*****	MC	

Table B.14: **Testing for differences between TREE-QMC-n2 vs TREE-QMC-n1 on the Asteroid data.** BET is the number of replicates for which TREE-QMC-n2 has lower species tree error and thus is better than TREE-QMC-n1, and WOR is the opposite. Significance is evaluated using Wilcoxon signed-rank tests on the error rates. The symbols *, **, ***, ****, ***** indicate significance at $p < 0.5$, 0.005, and so on. MC indicates significance after Bonferroni correction, i.e., $p < 0.05 / 312 =$ 2e-04 for the 312 tests made on the Asteroid data.

TREE-QMC-n2 vs TREE-QMC-n1								
		BET	WOR	TIE	p-val	sig	MC	note
False negative error rate								
Varying population size	10	17	12	21	0.6			
	50000000	11	13	26	0.9			
	100000000	18	16	16	0.6			
	500000000	19	17	14	0.5			
	1000000000	14	15	21	0.8			
Varying number of taxa	25	7	8	35	1			
	75	22	15	13	0.2			
	50	11	13	26	0.9			(dup)
	100	21	17	12	0.2			
	125	26	16	8	0.04	*		
	150	27	16	7	0.03	*		
Varying number of genes	250	18	14	18	0.5			
	500	20	13	17	0.2			
	1000	11	13	26	0.9			(dup)
	2000	14	8	28	0.2			
Varying sequence length	50	16	17	17	0.5			
	100	11	13	26	0.9			(dup)
	200	14	14	22	1			
	500	6	13	31	0.09			
Varying branch length scaler	0.05	15	24	11	0.3			
	0.1	14	22	14	0.2			
	1	11	13	26	0.9			(dup)
	10	21	10	19	0.01	*		
	100	30	12	8	0.002	**		
	200	22	19	9	0.6			
Varying missingness parameter	0.5	16	12	22	0.4			
	0.55	16	12	22	0.7			
	0.6	11	13	26	0.9			(dup)
	0.65	21	12	17	0.09			
	0.7	21	15	14	0.3			
	0.75	14	22	14	0.2			

Table B.15: **Testing for differences between TREE-QMC-n2 vs TREE-QMC-n1 on the Asteroid data (continued).**

TREE-QMC-n2 vs TREE-QMC-n1								
		BET	WOR	TIE	p-val	sig	MC	note
False positive error rate								
Varying population size	10	17	13	20	0.5			
	50000000	13	13	24	1			
	100000000	18	17	15	0.7			
	500000000	19	17	14	0.5			
	1000000000	17	18	15	0.9			
Varying number of taxa	25	7	8	35	1			
	75	22	16	12	0.3			
	50	13	13	24	1			(dup)
	100	21	17	12	0.2			
	125	27	16	7	0.2			
	150	28	16	6	0.03	*		
Varying number of genes	250	16	19	15	1			
	500	22	16	12	0.2			
	1000	13	13	24	1			(dup)
	2000	14	8	28	0.2			
Varying sequence length	50	16	17	17	0.7			
	100	13	13	24	1			(dup)
	200	14	14	22	0.8			
	500	6	13	31	0.06			
Varying branch length scaler	0.05	16	25	9	0.4			
	0.1	14	23	13	0.2			
	1	13	13	24	1			(dup)
	10	24	10	16	0.02	*		
	100	31	12	7	0.01	*		
	200	22	19	9	0.5			
Varying missingness parameter	0.5	16	12	22	0.5			
	0.55	17	12	21	0.8			
	0.6	13	13	24	1			(dup)
	0.65	22	12	16	0.05			
	0.7	24	15	11	0.3			
	0.75	18	24	8	0.3			

Table B.16: **Testing for differences between TREE-QMC-n2 vs TREE-QMC-n0 on the Asteroid data.** BET is the number of replicates for which TREE-QMC-n2 has lower species tree error and thus is better than TREE-QMC-n0, and WOR is the opposite. Significance is evaluated using Wilcoxon signed-rank tests on the error rates. The symbols *, **, ***, ****, ***** indicate significance at $p < 0.5$, 0.005, and so on. MC indicates significance after Bonferroni correction, i.e., $p < 0.05 / 312 =$ 2e-04 for the 312 tests made on the Asteroid data.

TREE-QMC-n2 vs TREE-QMC-n0								
		BET	WOR	TIE	p-val	sig	MC	note
False negative error rate								
Varying population size	10	33	10	7	4e-05	****	MC	
	50000000	30	12	8	6e-04	**		
	100000000	33	11	6	3e-05	****	MC	
	500000000	33	8	9	1e-04	***	MC	
	1000000000	35	11	4	3e-06	*****	MC	
Varying number of taxa	25	29	8	13	6e-05	***	MC	
	75	30	11	9	3e-04	***		
	50	30	12	8	6e-04	**		(dup)
	100	39	7	4	7e-09	*****	MC	
	125	39	6	5	1e-07	*****	MC	
	150	46	3	1	7e-13	*****	MC	
Varying number of genes	250	34	8	8	2e-06	*****	MC	
	500	33	10	7	1e-05	****	MC	
	1000	30	12	8	6e-04	**		(dup)
	2000	36	8	6	2e-05	****	MC	
Varying sequence length	50	31	9	10	1e-04	***	MC	
	100	30	12	8	6e-04	**		(dup)
	200	26	11	13	0.004	**		
	500	27	8	15	0.002	**		
Varying branch length scaler	0.05	29	14	7	0.01	*		
	0.1	34	10	6	7e-05	***	MC	
	1	30	12	8	6e-04	**		(dup)
	10	34	12	4	1e-05	****	MC	
	100	37	7	6	3e-08	*****	MC	
	200	31	13	6	6e-04	**		
Varying missingness parameter	0.5	33	8	9	0.001	**		
	0.55	31	6	13	3e-05	****	MC	
	0.6	30	12	8	6e-04	**		(dup)
	0.65	38	8	4	3e-07	*****	MC	
	0.7	31	11	8	0.003	**		
	0.75	30	15	5	0.01	*		

Table B.17: **Testing for differences between TREE-QMC-n2 vs TREE-QMC-n0 on the Asteroid data (continued).**

TREE-QMC-n2 vs TREE-QMC-n0								
		BET	WOR	TIE	p-val	sig	MC	note
False positive error rate								
Varying population size	10	35	9	6	3e-05	****	MC	
	50000000	32	13	5	5e-04	**		
	100000000	35	11	4	1e-05	****	MC	
	500000000	34	11	5	1e-04	***	MC	
	1000000000	36	14	0	2e-06	*****	MC	
Varying number of taxa	25	30	10	10	3e-04	***		
	75	30	13	7	4e-04	***		
	50	32	13	5	5e-04	**		(dup)
	100	40	8	2	3e-09	*****	MC	
	125	42	5	3	5e-08	*****	MC	
	150	47	3	0	8e-14	*****	MC	
Varying number of genes	250	35	11	4	3e-06	*****	MC	
	500	36	9	5	4e-06	*****	MC	
	1000	32	13	5	5e-04	**		(dup)
	2000	36	8	6	1e-05	****	MC	
Varying sequence length	50	32	9	9	5e-05	****	MC	
	100	32	13	5	5e-04	**		(dup)
	200	28	13	9	0.004	**		
	500	28	9	13	4e-04	***		
Varying branch length scaler	0.05	33	15	2	0.004	**		
	0.1	35	12	3	3e-05	****	MC	
	1	32	13	5	5e-04	**		(dup)
	10	37	12	1	5e-06	*****	MC	
	100	40	8	2	3e-08	*****	MC	
	200	35	13	2	3e-04	***		
Varying missingness parameter	0.5	33	9	8	0.002	**		
	0.55	35	6	9	1e-05	****	MC	
	0.6	32	13	5	5e-04	**		(dup)
	0.65	40	7	3	2e-07	*****	MC	
	0.7	32	13	5	9e-04	**		
	0.75	36	13	1	5e-04	**		

B.5.2 Additional Results on S100 data

Table B.18: **Species tree (RF) error rate for S100 simulated data.** Mean error rate is given across 50 replicates for each method. Note: TREE-QMC-n2 (no weighting) refines polytomies in the input gene trees randomly; thus, it is always given trees with bootstrap support because IQTree refines polytomies when computing abayes support

# of genes	sequence length	ASTRID ws	ASTRAL IV-wh	TREE-QMC wh-n2	TREE-QMC wh-n1	TREE-QMC wh-n0	TREE-QMC ws-n2	TREE-QMC n2
bootstrap support for weighted methods								
50	200	0.16310	0.15530	**0.15390**	0.15630	0.16740	0.15610	0.17590
50	400	0.13120	0.12550	**0.12220**	0.12350	0.13200	0.12350	0.13100
50	800	0.11350	0.10390	**0.10310**	0.10430	0.11040	0.10530	0.11450
50	1600	0.10350	0.09590	**0.09290**	0.09530	0.09920	0.09690	0.09710
200	200	0.09710	0.09630	**0.09490**	0.09880	0.10020	0.09820	0.10960
200	400	0.07960	0.07670	**0.07290**	0.07690	0.08330	0.07470	0.08550
200	800	0.06900	0.06470	**0.06290**	0.06430	0.06800	0.06410	0.06980
200	1600	0.06140	0.05820	**0.05550**	0.05880	0.06160	0.05610	0.06410
500	200	0.07740	**0.07530**	0.07550	0.07920	0.08020	0.07940	0.09260
500	400	0.06160	0.05960	**0.05550**	0.05960	0.06450	0.06000	0.07220
500	800	0.05080	0.04900	**0.04800**	0.04960	0.05350	0.04920	0.05470
500	1600	0.04410	0.04290	**0.04060**	0.04260	0.04430	0.04240	0.04880
1000	200	0.06690	**0.06530**	0.06730	0.07160	0.07020	0.07040	0.08000
1000	400	0.05160	0.05140	**0.04800**	0.05020	0.05350	0.05060	0.06330
1000	800	0.04200	0.04180	**0.03900**	0.04260	0.04490	0.04120	0.05040
1000	1600	0.03590	0.03610	**0.03510**	0.03630	0.03820	0.03530	0.04100
abayes support for weighted methods								
50	200	0.15040	0.14570	**0.14140**	0.14570	0.15860	0.14550	0.17590
50	400	0.12390	0.12180	**0.11120**	0.11690	0.12330	0.11530	0.13100
50	800	0.11140	0.10840	**0.09670**	0.10370	0.11260	0.10060	0.11450
50	1600	0.10450	0.09550	**0.08630**	0.09310	0.09940	0.08820	0.09710
200	200	0.09470	0.09220	**0.08310**	0.08710	0.09800	0.09430	0.10960
200	400	0.07430	0.07220	**0.06710**	0.07120	0.07780	0.07180	0.08550
200	800	0.06780	0.06410	**0.06180**	0.06410	0.06760	0.06240	0.06980
200	1600	0.05880	0.05650	**0.05160**	0.05490	0.05800	0.05370	0.06410
500	200	0.07530	0.07160	**0.06920**	0.07290	0.07780	0.07840	0.09260
500	400	0.06080	**0.05760**	0.05860	0.05840	0.06080	0.06160	0.07220
500	800	0.04880	0.04690	**0.04370**	0.04590	0.04960	0.04820	0.05470
500	1600	0.04080	0.03920	**0.03530**	0.03860	0.04220	0.03820	0.04880
1000	200	0.06220	0.06310	**0.05350**	0.06100	0.06840	0.05920	0.08000
1000	400	0.05120	0.05160	**0.04650**	0.05160	0.05610	0.05240	0.06330
1000	800	0.04220	0.04180	**0.03630**	0.04060	0.04650	0.04140	0.05040
1000	1600	0.03740	0.03530	**0.02880**	0.03490	0.04000	0.03410	0.04100

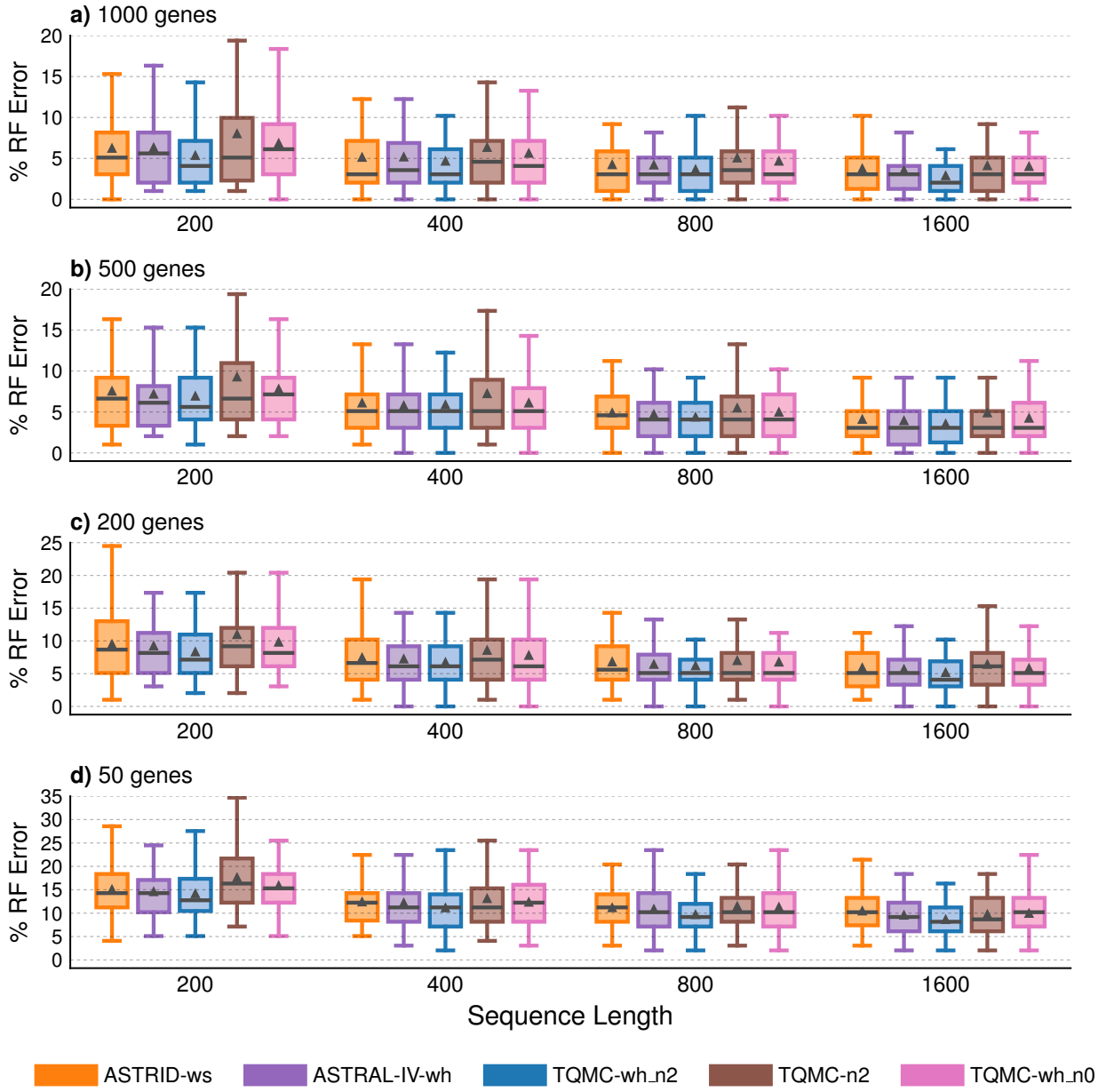


Figure B.9: **Species tree error for S100 data with abayes support.** Percent species tree (RF) error is shown on *y*-axis for 50 replicates in each model condition. Bars represent medians, triangles represent means, outliers are not shown.

Table B.19: **Testing for differences between TREE-QMC-wh_n2 vs ASTRAL-IV-wh on the S100 simulated data.** BET is the number of replicates for which TREE-QMC-wh_n2 has lower species tree (RF) error and thus is better than ASTRAL-IV-wh, WOR is the number of replicates for which TREE-QMC-wh_n2 has higher RF error and thus is worse than ASTRAL-IV-wh, and TIE is the number of replicates where the two methods tie. Significance is evaluated using paired, two-sided Wilcoxon signed-rank tests on the RF error rates. The symbols *, **, ***, ****, ***** indicate significance at $p < 0.5$, 0.005, and so on. MC indicates significance after Bonferroni correction, i.e., $p < 0.05 / 96$ = 5e-04 for the 96 tests made on the S100 data.

TREE-QMC-wh_n2 vs ASTRAL-IV-wh								
# of genes	sequence length	BET	WOR	TIE	p-val	sig	MC	note
Abayes Support for weighted methods								
50	200	23	17	10	0.3			
50	400	33	7	10	4e-04	***	MC	
50	800	27	8	15	3e-04	***	MC	
50	1600	23	6	21	4e-04	***	MC	
200	200	30	9	11	0.007	*		
200	400	21	14	15	0.2			
200	800	18	11	21	0.2			
200	1600	22	10	18	0.02	*		
500	200	22	15	13	0.8			
500	400	14	22	14	0.5			
500	800	18	11	21	0.5			
500	1600	21	13	16	0.2			
1000	200	20	9	21	0.006	*		
1000	400	20	6	24	0.003	**		
1000	800	20	7	23	0.008	*		
1000	1600	19	5	26	0.002	**		

Table B.20: **Testing for differences between TREE-QMC-wh_n2 vs ASTRAL-IV-wh on the S100 simulated data (continued).**

TREE-QMC-wh_n2 vs ASTRAL-IV-wh								
# of genes	sequence length	BET	WOR	TIE	p-val	sig	MC	note
Boostrap Support for weighted methods								
50	200	22	17	11	0.4			
50	400	19	14	17	0.5			
50	800	23	18	9	0.8			
50	1600	21	16	13	0.2			
200	200	19	17	14	0.6			
200	400	21	11	18	0.06			
200	800	17	18	15	0.6			
200	1600	17	11	22	0.2			
500	200	23	18	9	0.8			
500	400	21	14	15	0.1			
500	800	17	10	23	0.3			
500	1600	13	8	29	0.2			
1000	200	16	19	15	0.6			
1000	400	16	10	24	0.1			
1000	800	17	6	27	0.09			
1000	1600	13	10	27	0.4			

Table B.21: **Testing for differences between TREE-QMC-wh_n2 vs ASTRID-ws on the S100 simulated data.** BET is the number of replicates for which TREE-QMC-wh_n2 has lower species tree (RF) error and thus is better than ASTRID-ws, WOR is the number of replicates for which TREE-QMC-wh_n2 has higher RF error and thus is worse than ASTRID-ws, and TIE is the number of replicates where the two methods tie. Significance is evaluated using paired, two-sided Wilcoxon signed-rank tests on the RF error rates. The symbols *, **, ***, ****, ***** indicate significance at $p < 0.5$, 0.005, and so on. MC indicates significance after Bonferroni correction, i.e., $p < 0.05 / 96$ = 5e-04 for the 96 tests made on the S100 data.

TREE-QMC-wh_n2 vs ASTRID-ws								
# of genes	sequence length	BET	WOR	TIE	p-val	sig	MC	note
Abayes Support for weighted methods								
50	200	30	16	4	0.03	*		
50	400	33	12	5	0.001	**		
50	800	35	10	5	2e-04	***	MC	
50	1600	34	10	6	2e-05	****	MC	
200	200	27	7	16	3e-04	***	MC	
200	400	25	13	12	0.03	*		
200	800	25	13	12	0.03	*		
200	1600	29	7	14	0.006	*		
500	200	24	16	10	0.1			
500	400	23	13	14	0.1			
500	800	22	11	17	0.02	*		
500	1600	24	9	17	0.04	*		
1000	200	25	9	16	0.008	*		
1000	400	21	13	16	0.07			
1000	800	21	8	21	0.004	**		
1000	1600	23	5	22	4e-04	***	MC	

Table B.22: **Testing for differences between TREE-QMC-wh_n2 vs ASTRID-ws on the S100 simulated data (continued).**

TREE-QMC-wh_n2 vs ASTRID-ws								
# of genes	sequence length	BET	WOR	TIE	p-val	sig	MC	note
Boostrap Support for weighted methods								
50	200	30	12	8	0.004	**		
50	400	31	12	7	0.002	**		
50	800	31	13	6	0.002	**		
50	1600	32	9	9	2e-04	***	MC	
200	200	20	18	12	0.6			
200	400	22	14	14	0.03	*		
200	800	23	11	16	0.02	*		
200	1600	19	11	20	0.04	*		
500	200	17	14	19	0.6			
500	400	25	9	16	0.002	**		
500	800	17	12	21	0.2			
500	1600	20	9	21	0.03	*		
1000	200	23	18	9	1			
1000	400	21	11	18	0.09			
1000	800	15	9	26	0.1			
1000	1600	15	12	23	0.4			

Table B.23: **Testing for differences between TREE-QMC-wh_n2 vs TREE-QMC-n2 on the S100 simulated data.** BET is the number of replicates for which TREE-QMC-wh_n2 has lower species tree (RF) error and thus is better than TREE-QMC-n2, WOR is the number of replicates for which TREE-QMC-wh_n2 has higher RF error and thus is worse than TREE-QMC-n2, and TIE is the number of replicates where the two methods tie. Significance is evaluated using paired, two-sided Wilcoxon signed-rank tests on the RF error rates. The symbols *, **, ***, ****, ***** indicate significance at $p < 0.5$, 0.005, and so on. MC indicates significance after Bonferroni correction, i.e., $p < 0.05$ / 96 = 5e-04 for the 96 tests made on the S100 data. Note: TREE-QMC-n2 refines polytomies in the input gene trees randomly (note that it is given trees with bootstrap support because IQTree refines polytomies when computing abayes support)

TREE-QMC-wh_n2 vs TREE-QMC-n2								
# of genes	sequence length	BET	WOR	TIE	p-val	sig	MC	note
Abayes Support for weighted methods								
50	200	38	8	4	3e-08	*****	MC	
50	400	31	11	8	9e-05	***	MC	
50	800	31	8	11	9e-05	***	MC	
50	1600	24	16	10	0.05	*		
200	200	33	11	6	1e-05	****	MC	
200	400	34	8	8	3e-06	*****	MC	
200	800	22	12	16	0.04	*		
200	1600	29	9	12	9e-04	**		
500	200	28	10	12	0.001	**		
500	400	28	8	14	9e-05	***	MC	
500	800	22	9	19	0.008	*		
500	1600	23	6	21	4e-04	***	MC	
1000	200	27	7	16	4e-05	****	MC	
1000	400	28	4	18	2e-06	*****	MC	
1000	800	28	5	17	9e-06	****	MC	
1000	1600	22	5	23	0.002	**		

Table B.24: **Testing for differences between TREE-QMC-wh_n2 vs TREE-QMC-n2 on the S100 simulated data (continued).**

TREE-QMC-wh_n2 vs TREE-QMC-n2								
# of genes	sequence length	BET	WOR	TIE	p-val	sig	MC	note
Boostrap Support for weighted methods								
50	200	29	13	8	3e-04	***	MC	
50	400	25	20	5	0.4			
50	800	26	15	9	0.04	*		
50	1600	22	18	10	0.3			
200	200	24	16	10	0.05	*		
200	400	29	12	9	0.005	*		
200	800	17	14	19	0.3			
200	1600	24	9	17	0.003	**		
500	200	24	19	7	0.05	*		
500	400	26	9	15	2e-04	***	MC	
500	800	18	16	16	0.3			
500	1600	17	12	21	0.2			
1000	200	26	17	7	0.04	*		
1000	400	27	10	13	0.001	**		
1000	800	21	9	20	0.005	**		
1000	1600	12	11	27	0.6			

B.5.3 Additional Results on ASTRAL-II data

Table B.25: **Runtime for ASTRAL-II simulated data.** Mean runtime (in seconds) is given across replicates for each method. All weighted methods were given gene trees with abayes support. CA-ML trees from original study were not available for 1000-taxon, 1000-gene data sets.

# of taxa	ILS level	speciation location	# of genes	ASTRID ws	TREE-QMC (n2)	TREE-QMC wh (n2)	ASTRAL-IV wh (16 threads)	ASTRAL-IV wh (1 thread)
10	medium	shallow	1000	0.0	0.2	0.3	0.1	0.7
50	medium	shallow	1000	0.1	3.3	7.5	2.4	34.5
100	medium	shallow	1000	0.1	13.8	40.5	10.7	131.6
200	low	deep	1000	0.3	51.8	170.8	41.5	700.0
200	low	shallow	1000	0.3	50.4	159.9	36.9	645.5
200	medium	deep	1000	0.3	55.7	181.3	44.0	734.3
200	medium	shallow	1000	0.3	51.7	167.1	41.1	661.1
200	high	deep	1000	0.4	61.1	201.6	54.0	955.0
200	high	shallow	1000	0.4	58.8	192.7	53.8	869.3
500	medium	shallow	1000	1.9	322.8	1084.0	273.1	5145.7
1000	medium	shallow	1000	7.3	1303.3	4469.8	1272.5	28758.8

Table B.26: **Species tree (RF) error rate for ASTRAL-II simulated data.** Mean error rate is given across replicates for each method. All weighted methods were given gene trees with abayes support. CA-ML trees from original study were not available for 1000-taxon, 1000-gene data sets.

# of taxa	ILS level	speciation location	# of genes	CA-ML	ASTRAL-IV wh	TREE-QMC wh (n2)	TREE-QMC (n2)	ASTRID ws
10	medium	shallow	50	0.03830	0.03060	0.03060	0.03570	**0.02810**
10	medium	shallow	200	0.01790	**0.00760**	0.01020	0.01790	**0.00760**
10	medium	shallow	1000	0.02080	**0.01560**	**0.01560**	**0.01560**	**0.01560**
50	medium	shallow	50	0.07800	0.06070	0.05850	0.07090	**0.05760**
50	medium	shallow	200	0.04520	0.03060	**0.02840**	0.04120	0.03550
50	medium	shallow	1000	0.02660	0.01860	**0.01770**	0.02530	0.01910
100	medium	shallow	50	0.09120	0.07160	**0.06720**	0.07230	0.07040
100	medium	shallow	200	0.04740	0.03830	**0.03610**	0.04680	0.03930
100	medium	shallow	1000	0.02490	**0.01910**	0.01980	0.03040	0.02400
200	low	deep	50	**0.03980**	0.05830	0.05160	0.06710	0.06490
200	low	deep	200	**0.02230**	0.03520	0.03200	0.05060	0.05130
200	low	deep	1000	**0.01780**	0.03010	0.02480	0.04520	0.04850
200	low	shallow	50	0.05360	0.04810	**0.04240**	0.05280	0.04430
200	low	shallow	200	0.03110	0.02350	0.02240	0.02910	**0.02230**
200	low	shallow	1000	0.01440	0.01260	**0.01120**	0.01880	0.01300
200	medium	deep	50	0.10300	0.08920	**0.08570**	0.09750	0.09100
200	medium	deep	200	0.05690	0.05230	**0.04660**	0.05700	0.05500
200	medium	deep	1000	**0.02840**	0.03520	0.03090	0.03980	0.03920
200	medium	shallow	50	0.09220	0.07060	**0.06690**	0.08090	0.07280
200	medium	shallow	200	0.05520	0.04030	**0.03820**	0.04750	0.04170
200	medium	shallow	1000	0.02780	0.02410	**0.02270**	0.03170	0.02600
200	high	deep	50	0.28160	0.18980	**0.18580**	0.20670	0.20630
200	high	deep	200	0.16100	**0.09130**	0.09190	0.10940	0.10560
200	high	deep	1000	0.08010	**0.04710**	0.04860	0.06170	0.04930
200	high	shallow	50	0.27950	0.18180	**0.17430**	0.18980	0.20220
200	high	shallow	200	0.16250	0.08940	**0.08300**	0.09600	0.10130
200	high	shallow	1000	0.07950	0.04000	**0.03770**	0.04930	0.04710
500	medium	shallow	50	0.09240	0.07400	**0.06650**	0.07630	0.07400
500	medium	shallow	200	0.04720	0.04000	**0.03410**	0.04260	0.04100
500	medium	shallow	1000	0.02330	0.02430	**0.02030**	0.02810	0.02610
1000	medium	shallow	50	0.09760	0.08670	**0.07680**	0.10110	0.08810
1000	medium	shallow	200	0.05150	0.04850	**0.04180**	0.05880	0.05180
1000	medium	shallow	1000	nan	0.03050	**0.02460**	0.03640	0.03320

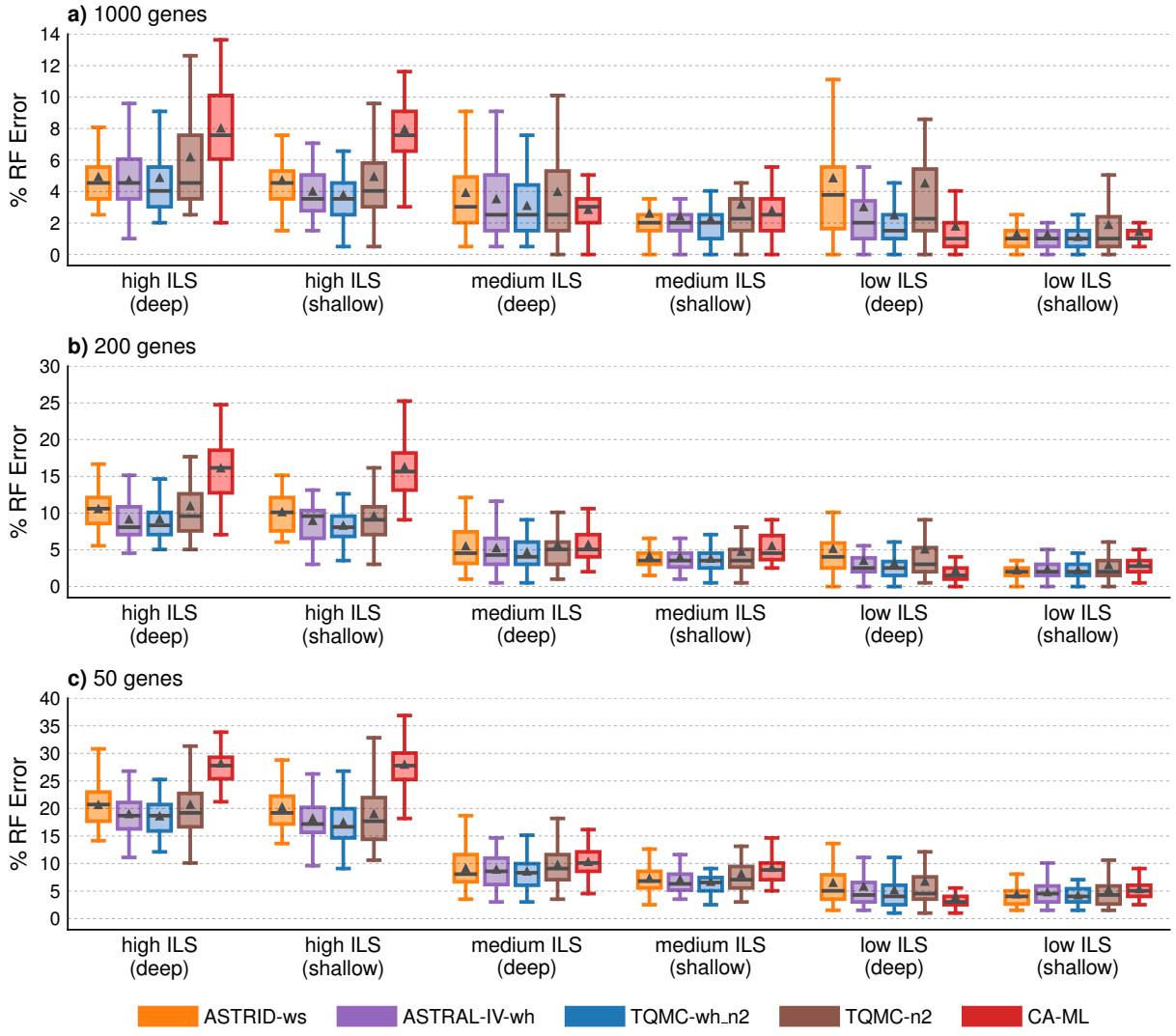


Figure B.10: **Species tree error for ASTRAL-II data (abayes support) with varying levels of ILS.** Percent species tree (RF) error across replicates (bars represent medians; triangles represent means; outliers are not shown).

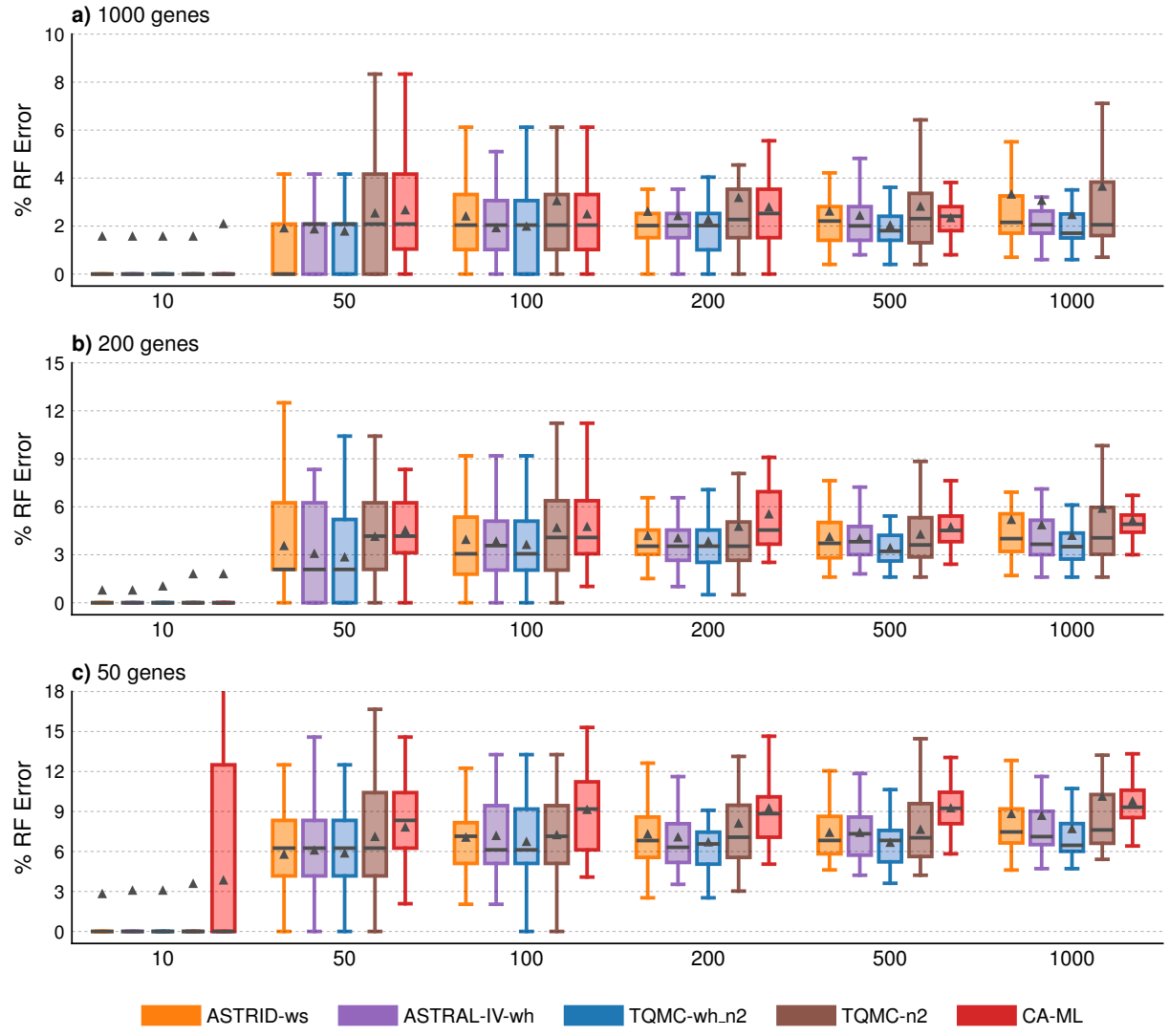


Figure B.11: **Species tree error for ASTRAL-II data (abayes support) with varying numbers of taxa.** Percent species tree (RF) error across replicates (bars represent medians; triangles represent means; outliers are not shown). Note that CA-ML trees from original study were not available for 1000-taxon, 1000-gene data sets. Also, the results for 200-taxon data sets are duplicated from Figure B.10

Table B.27: **Testing for differences between TREE-QMC-wh_n2 vs CAML on the ASTRAL-II simulated data (abayes support) with varying levels of ILS.** BET is the number of replicates for which TREE-QMC-wh_n2 has lower species tree (RF) error and thus is better than CAML, WOR is the number of replicates for which TREE-QMC-wh_n2 has higher RF error and thus is worse than CAML, and TIE is the number of replicates where the two methods tie. Significance is evaluated using paired, two-sided Wilcoxon signed-rank tests on the RF error rates. The symbols *, **, ***, ****, ***** indicate significance at $p <$ 0.5, 0.005, and so on. MC indicates significance after Bonferroni correction, i.e., $p <$ 0.05 / 99 = 5e-04 for the 99 tests made on the S100 data.

TREE-QMC-wh_n2 vs CAML

ILS Level	Speciation	# of genes	BET	WOR	TIE	p-val	sig	MC	note
low	deep	50	12	31	7	0.001014695		**	(CAML better)
low	deep	200	13	28	9	0.001628425		**	(CAML better)
low	deep	1000	13	29	8	0.01314316		*	(CAML better)
low	shallow	50	36	9	5	5.632589e-06		****	MC
low	shallow	200	38	6	6	8.63588e-06		****	MC
low	shallow	1000	29	10	11	0.005499648		*	
medium	deep	50	38	10	2	4.729318e-06		*****	MC
medium	deep	200	37	9	4	0.0004051359		***	MC
medium	deep	1000	24	19	7	0.9784243			
medium	shallow	50	44	3	3	3.538503e-12		*****	MC
medium	shallow	200	40	4	6	2.048751e-09		*****	MC
medium	shallow	1000	33	8	9	0.0003452434		***	MC
high	deep	50	50	0	0	1.776357e-15		*****	MC
high	deep	200	50	0	0	1.776357e-15		*****	MC
high	deep	1000	45	4	0	1.20962e-07		*****	MC
high	shallow	50	47	0	0	1.421085e-14		*****	MC
high	shallow	200	47	0	0	1.421085e-14		*****	MC
high	shallow	1000	46	0	1	2.842171e-14		*****	MC

Table B.28: **Testing for differences between TREE-QMC-wh_n2 vs ASTRAL-IV-wh on the ASTRAL-II simulated data (abayes support) with varying levels of ILS.** BET is the number of replicates for which TREE-QMC-wh_n2 has lower species tree (RF) error and thus is better than ASTRAL-IV-wh, WOR is the number of replicates for which TREE-QMC-wh_n2 has higher RF error and thus is worse than ASTRAL-IV-wh, and TIE is the number of replicates where the two methods tie. Significance is evaluated using paired, two-sided Wilcoxon signed-rank tests on the RF error rates. The symbols *, **, ***, ****, ***** indicate significance at $p < 0.5$, 0.005, and so on. MC indicates significance after Bonferroni correction, i.e., $p <$ 0.05 / 99 = 5e-04 for the 99 tests made on the S100 data.

TREE-QMC-wh_n2 vs ASTRAL-IV-wh

ILS Level	Speciation	# of genes	BET	WOR	TIE	p-val	sig	MC	note
low	deep	50	30	3	17	2.060551e-07		*****	MC
low	deep	200	21	11	18	0.02371259		*	
low	deep	1000	26	2	22	1.696497e-05		****	MC
low	shallow	50	32	10	8	6.118376e-05		***	MC
low	shallow	200	21	13	16	0.1606247			
low	shallow	1000	17	5	28	0.03724957		*	
medium	deep	50	24	14	12	0.05695275			
medium	deep	200	29	8	13	0.0001611809		***	MC
medium	deep	1000	22	11	17	0.0186442		*	
medium	shallow	50	22	17	11	0.07322669			
medium	shallow	200	25	10	15	0.01056484		*	
medium	shallow	1000	17	5	28	0.01173639		*	
high	deep	50	25	21	4	0.2112505			
high	deep	200	24	21	5	0.9229223			
high	deep	1000	20	17	12	0.9970997			
high	shallow	50	29	13	5	0.01055602		*	
high	shallow	200	29	12	6	0.01537982		*	
high	shallow	1000	26	14	7	0.07923641			

Table B.29: **Testing for differences between TREE-QMC-wh_n2 vs ASTRID-ws on the ASTRAL-II simulated data (abayes support) with varying levels of ILS.** BET is the number of replicates for which TREE-QMC-wh_n2 has lower species tree (RF) error and thus is better than ASTRID-ws, WOR is the number of replicates for which TREE-QMC-wh_n2 has higher RF error and thus is worse than ASTRID-ws, and TIE is the number of replicates where the two methods tie. Significance is evaluated using paired, two-sided Wilcoxon signed-rank tests on the RF error rates. The symbols *, **, ***, ****, ***** indicate significance at $p < 0.5$, 0.005, and so on. MC indicates significance after Bonferroni correction, i.e., $p <$ 0.05 / 99 = 5e-04 for the 99 tests made on the S100 data.

TREE-QMC-wh_n2 vs ASTRID-ws

ILS Level	Speciation	# of genes	BET	WOR	TIE	p-val	sig	MC	note
low	deep	50	31	7	12	1.008374e-05		****	MC
low	deep	200	38	5	7	2.443812e-09		*****	MC
low	deep	1000	43	2	5	3.581135e-12		*****	MC
low	shallow	50	22	13	15	0.2696331			
low	shallow	200	12	20	18	0.2902047			
low	shallow	1000	14	7	29	0.2642612			
medium	deep	50	25	20	5	0.05764566			
medium	deep	200	33	10	7	6.837434e-05		***	MC
medium	deep	1000	30	6	14	4.754454e-05		****	MC
medium	shallow	50	30	13	7	0.001086662		**	
medium	shallow	200	26	13	11	0.02047486		*	
medium	shallow	1000	26	12	12	0.01566447		*	
high	deep	50	35	9	6	9.161746e-06		****	MC
high	deep	200	37	10	3	1.476171e-05		****	MC
high	deep	1000	29	13	7	0.1618871			
high	shallow	50	39	5	3	8.781171e-10		*****	MC
high	shallow	200	40	5	2	5.238405e-08		*****	MC
high	shallow	1000	33	8	6	1.492041e-06		*****	MC

Table B.30: **Testing for differences between TREE-QMC-wh_n2 vs CAML on the ASTRAL-II simulated data (abayes support) with varying numbers of taxa.** BET is the number of replicates for which TREE-QMC-wh_n2 has lower species tree (RF) error and thus is better than CAML, WOR is the number of replicates for which TREE-QMC-wh_n2 has higher RF error and thus is worse than CAML, and TIE is the number of replicates where the two methods tie. Significance is evaluated using paired, two-sided Wilcoxon signed-rank tests on the RF error rates. The symbols *, **, ***, ****, ***** indicate significance at $p <$ 0.5, 0.005, and so on. MC indicates significance after Bonferroni correction, i.e., $p < 0.05 / 99$ = 5e-04 for the 99 tests made on the S100 data.

TREE-QMC-wh_n2 vs CAML

# of taxa	# of genes	BET	WOR	TIE	p-val	sig	MC	note
10	50	8	6	35	0.6334229			
10	200	4	2	43	0.53125			
10	1000	2	1	45	0.75			
50	50	28	9	10	0.0001925013	***	MC	
50	200	29	12	6	0.0009076932	**		
50	1000	21	3	23	0.0007148981	**		
100	50	38	5	5	1.594049e-08	*****	MC	
100	200	32	9	7	0.001508224	**		
100	1000	23	13	12	0.04959638	*		
200	50	44	3	3	3.538503e-12	*****	MC	(dup)
200	200	40	4	6	2.048751e-09	*****	MC	(dup)
200	1000	33	8	9	0.0003452434	***	MC	(dup)
500	50	46	3	1	8.881784e-14	*****	MC	
500	200	42	7	1	3.823587e-09	*****	MC	
500	1000	36	11	3	0.004434673	**		
1000	50	45	5	0	8.173611e-08	*****	MC	
1000	200	42	8	0	1.173318e-06	*****	MC	

Table B.31: **Testing for differences between TREE-QMC-wh_n2 vs ASTRAL-IV-wh on the ASTRAL-II simulated data (abayes support) with varying numbers of taxa.** BET is the number of replicates for which TREE-QMC-wh_n2 has lower species tree (RF) error and thus is better than ASTRAL-IV-wh, WOR is the number of replicates for which TREE-QMC-wh_n2 has higher RF error and thus is worse than ASTRAL-IV-wh, and TIE is the number of replicates where the two methods tie. Significance is evaluated using paired, two-sided Wilcoxon signed-rank tests on the RF error rates. The symbols *, **, ***, ****, ***** indicate significance at $p < 0.5$, 0.005, and so on. MC indicates significance after Bonferroni correction, i.e., $p < 0.05 / 99$ = 5e-04 for the 99 tests made on the S100 data.

TREE-QMC-wh_n2 vs ASTRAL-IV-wh								
# of taxa	# of genes	BET	WOR	TIE	p-val	sig	MC	note
10	50	1	1	47	1			
10	200	0	1	48	1			
10	1000	0	0	48	NA	NA	NA	
50	50	11	8	28	0.4520187			
50	200	8	3	36	0.2265625			
50	1000	5	3	39	0.7265625			
100	50	17	8	23	0.0395084	*		
100	200	15	7	26	0.1132421			
100	1000	8	10	30	0.67379			
200	50	22	17	11	0.07322669			(dup)
200	200	25	10	15	0.01056484	*		(dup)
200	1000	17	5	28	0.01173639	*		(dup)
500	50	38	10	2	1.885201e-07	*****	MC	
500	200	40	6	4	5.277997e-09	*****	MC	
500	1000	36	8	6	1.304655e-05	****	MC	
1000	50	44	2	4	4.547474e-13	*****	MC	
1000	200	38	4	8	4.101821e-10	*****	MC	
1000	1000	42	2	4	2.273737e-11	*****	MC	

Table B.32: **Testing for differences between TREE-QMC-wh_n2 vs ASTRID-ws on the ASTRAL-II simulated data (abayes support) with varying numbers of taxa.** BET is the number of replicates for which TREE-QMC-wh_n2 has lower species tree (RF) error and thus is better than ASTRID-ws, WOR is the number of replicates for which TREE-QMC-wh_n2 has higher RF error and thus is worse than ASTRID-ws, and TIE is the number of replicates where the two methods tie. Significance is evaluated using paired, two-sided Wilcoxon signed-rank tests on the RF error rates. The symbols *, **, ***, ****, ***** indicate significance at $p < 0.5$, 0.005, and so on. MC indicates significance after Bonferroni correction, i.e., $p <$ 0.05 / 99 = 5e-04 for the 99 tests made on the S100 data.

TREE-QMC-wh_n2 vs ASTRID-ws

# of taxa	# of genes	BET	WOR	TIE	p-val	sig	MC	note
10	50	1	2	46	1			
10	200	0	1	48	1			
10	1000	0	0	48	NA	NA	NA	
50	50	12	13	22	0.8604026			
50	200	18	3	26	0.00435257	**		
50	1000	6	3	38	0.5078125			
100	50	21	14	13	0.1545381			
100	200	19	13	16	0.1799249			
100	1000	14	7	27	0.0765667			
200	50	30	13	7	0.001086662	**		(dup)
200	200	26	13	11	0.02047486	*		(dup)
200	1000	26	12	12	0.01566447	*		(dup)
500	50	35	13	2	0.0001402717	***	MC	
500	200	41	7	2	4.245093e-08	*****	MC	
500	1000	29	9	12	3.824975e-05	****	MC	
1000	50	43	6	1	1.760259e-09	*****	MC	
1000	200	46	3	1	7.94742e-12	*****	MC	
1000	1000	39	6	3	6.072582e-10	*****	MC	

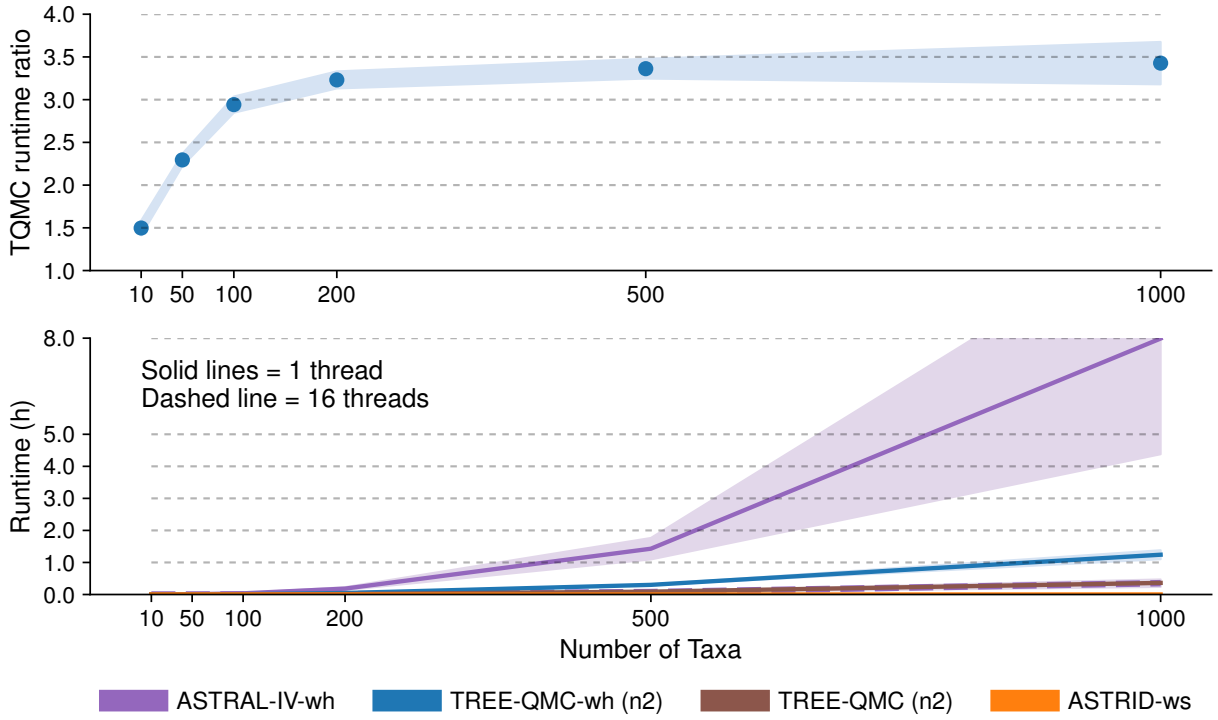


Figure B.12: **Runtime for ASTRAL-II data with varying numbers of taxa and 1000 gene trees with abayes support.** Subfigure (a) shows the ratio between the runtime for hybrid weighted TREE-QMC (n2) divided the original TREE-QMC (n2) method (-f option). Subfigure (b) shows the average runtime in hours for each of summary method (shaded region is standard error). Note that unweighted TREE-QMC (n2) is directly beneath ASTER-wh (16 threads).

B.6 Additional Results on Biological Data Sets

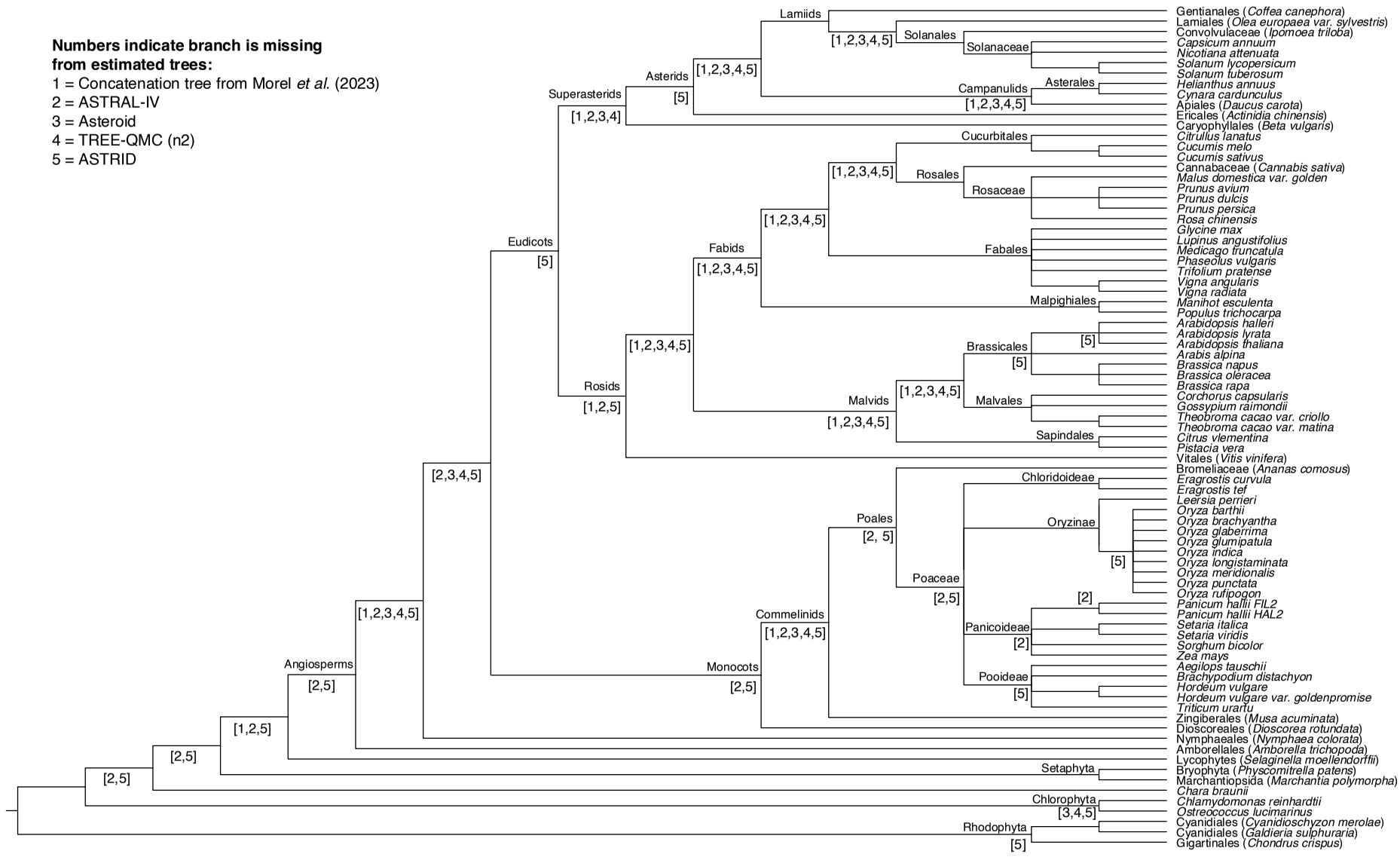


Figure B.13: **Plant reference tree.** Branches are annotated by clade name and numerical values, indicating which of the estimated trees is missing the branch. 1 is the concatenation tree from [86]. 2 is ASTRAL-IV. 3 is Asteroid. 4 is TREE-QMC (n2). 5 is ASTRID.

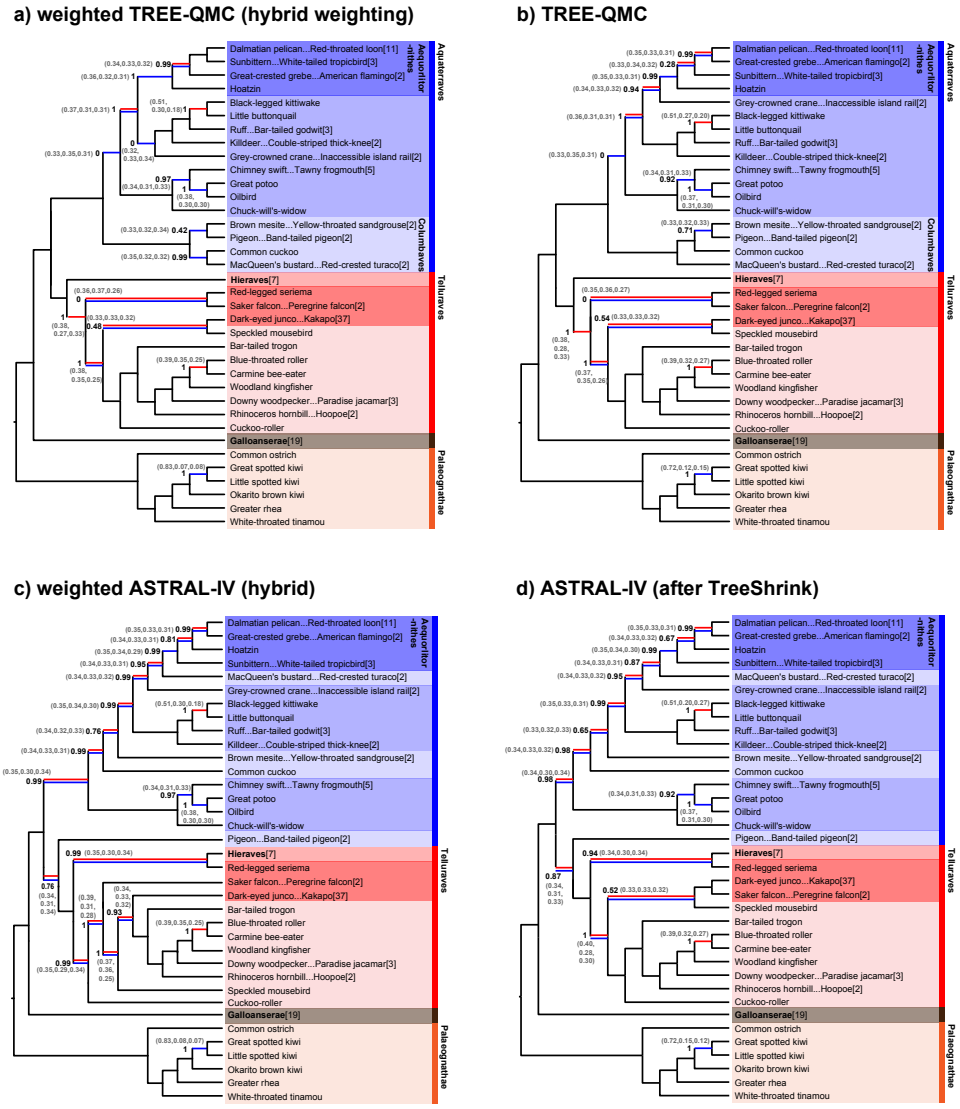


Figure B.14: **Species trees estimated from Wu *et al.*, 2024 data [145].** Subfigures (a)–(d) shows species tree we estimated on CDs and introns, continuing from the figure in the main text. For unweighted ASTRAL-IV only (subfigure d), outlier taxa were removed via TreeShrink prior to species tree estimation. Red lines are differences with the NJst tree. Blue lines are differences with the RAxML tree. Branch support (i.e., ASTRAL's local posterior probability) is shown on branches where there are disagreements between the NJst and RAxML trees. QQS values for branch are in parentheses. QQS values are computed with hybrid weighted quartets for subfigures (a) and (c) and unweighted quartets (on input with taxon filtering with TreeShrink) except for subfigures (c) and (d).

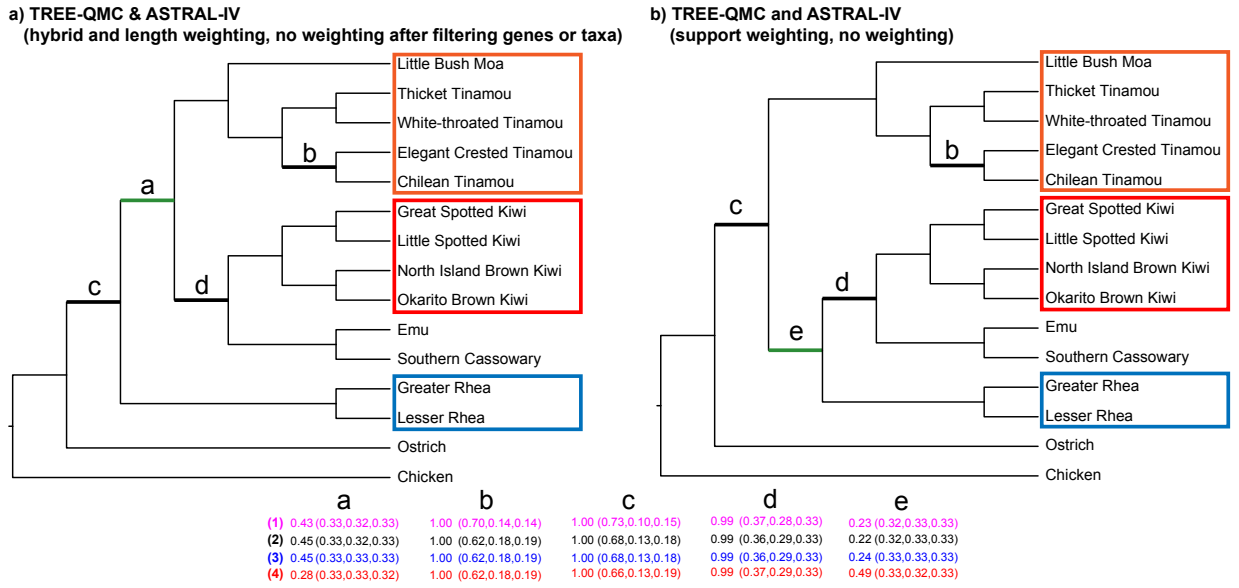


Figure B.15: **Species trees estimated from Cloutier *et al.*, 2019 data [145].** We used two types of data filtering: (1) removing the impacted taxa (Chicken and White-Throated Tinamou) from the 105 gene trees with homology errors and (2) these 105 gene trees entirely. Only two species trees were recovered by all methods. ASTRID and Asteroid also recovered the tree shown in subfigure (b) when using support, length, or no weighting (before or after either type of filtering). Branch support (ASTRAL's local posterior probability followed by the three normalized QQS values) is shown at bottom of the figure. Support was estimated in four ways: (1) hybrid quartet weighting (magenta), (2) no quartet weighting after filtering impacted taxa (black), (3) no quartet weighting after filtering gene trees (blue), (4) no quartet weighting (red).

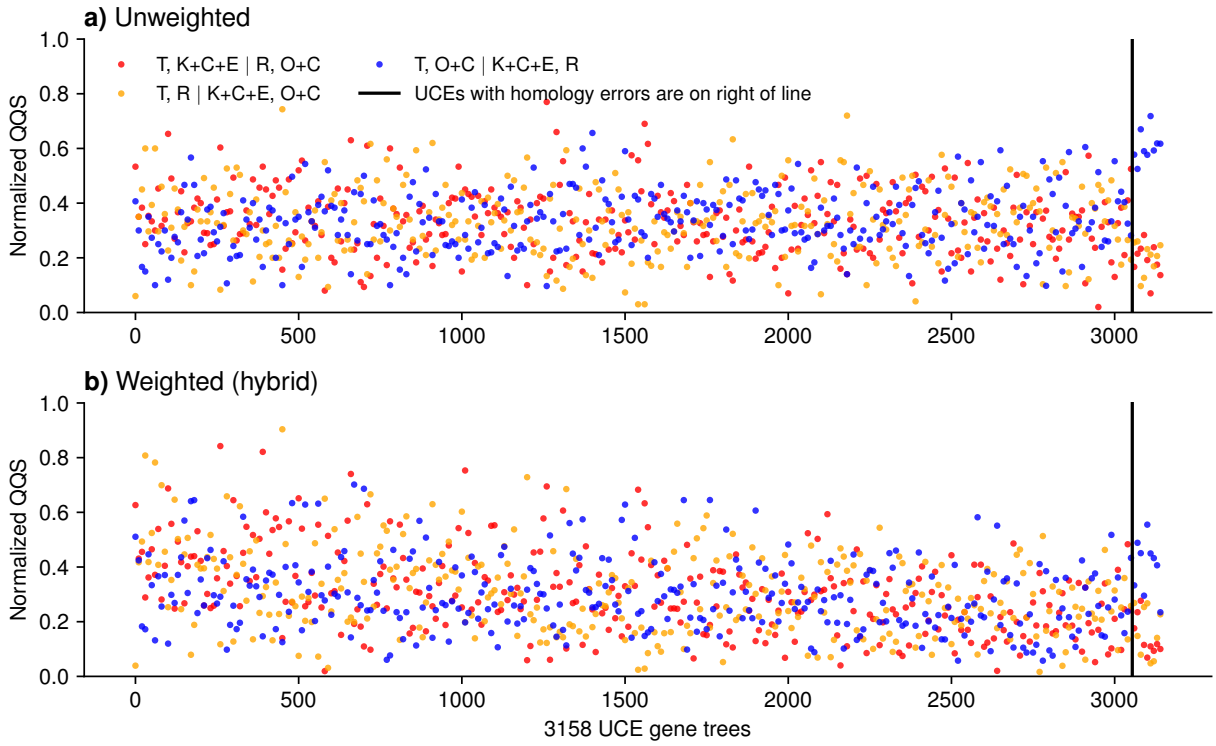


Figure B.16: **Partitioned Coalescent Support (PCS) analysis of Cloutier *et al.*, 2019 data [19].** The x-axis shows the 3,053 UCE gene trees sorted by mean abayes branch support, plus the 105 UCE gene trees with homology errors at the end. Subfigures (a) and (b) show normalized QQS values on the y-axis for unweighted and weighted (hybrid) quartets, respectively. Dots are averages across 10 gene trees (non-overlapping windows). The red, orange, and blue colors indicate the three possible resolutions of the focal branch. T corresponds to four Tinamou species plus Little Bush Moa, $K+C+E$ corresponds to the four Kiwi species plus Southern Cassowary and Emu, and R corresponds to the two Rhea species. $O+C$ indicates Ostrich plus Chicken. The homology errors cluster White-Throated Tinamou and Chicken, resulting in increased support for the related topology (blue).

Appendix C: Supplementary Materials for Chapter 4

C.1 Supplemental Methods

C.1.1 Network Simulation

We simulated networks with SiPhyNetwork R package [55], following the tutorial provided by the package.[1]

```
library(ape)
library(SiPhyNetwork)
set.seed(42)
inheritance.fxn <- make.beta.draw(10,10)
hybrid_proportions <- c(0.5, 0.25, 0.25)
gsa_nets <- sim.bdh.taxa.gsa(m = <upper bound number of taxa>,
                             n = <desired number of taxa>,
                             numbsim = <number of replicates>,
                             lambda = 1,
                             mu = 0.01,
                             nu = <hybridization rate>,
                             hybprops = hybrid_proportions,
                             hyb.inher.fxn = inheritance.fxn)
```

[1] https://cran.r-project.org/web/packages/SiPhyNetwork/vignettes/introduction.html

where m is an upper bound on the number of taxa (otherwise the network is discarded), n is the desired number of taxa, *numbsim* is the number of networks to simulate, λ is the speciation/birth rate, μ is the death/extinction rate, and ν is the hybridization rate. We set the parameters based on the vignette, except that we lowered the death rate to 0.01, as this sped up the simulation, although it still took hours per model condition.

We simulated two model conditions based on the number of taxa (50 and 100). For the 50 taxa model condition, we simulated 10,000 replicates, setting $m = 100$, $n = 50$, and $\nu = 0.02$; this yielded 35 level-0 networks, 447 level-1 networks, and 754 level 2-networks (the remaining networks were higher level). For the 100 taxa model condition, we simulated 20,000 replicates, setting $m = 150$, $n = 100$, and $\nu = 0.002$; this yielded 95 level-0 networks, 771 level 1-networks, and 1,179 level 2-networks (the remaining networks were higher level). For 50 and 100 taxa, we selected 50 replicates for level 1 and 50 replicates for level 2; we also selected 25 replicates for level 0, as a control to evaluate the level of incomplete lineage sorting (ILS).

C.1.2 Simulation of Gene Trees

We simulated gene trees from each network with PhyloNetworks Julia package [127], following the tutorial provided by the package.[2]

```
using PhyloNetworks
using PhyloCoalSimulations
using PhyloPlots
foreach(readdir(<directory>))do f
    println("Object: ", f)
    net = readTopology(readline(<directory> * f))
    trees = simulatecoalescent(<species network>, <number of gene trees>,
```

[2] https://juliaphylo.github.io/PhyloCoalSimulations.jl/stable/man/getting_started

```
        <number of individuals per species>)
    writeMultiTopology(trees, <directory> * f * ".truegenetrees")
end
```

We set the number of individuals per species to 1 and the number of gene trees to 1000.

C.1.3 Reconstruction of Trees of Blobs

TINNiK. We ran TINNiK [4], as implemented in MSCquartets v3.2, following the tutorial provided by the package.[3].

```
args=commandArgs(trailingOnly=TRUE)
library('MSCquartets')
gts = read.tree(file=args[1])
output = TINNIK(gts, alpha=as.numeric(args[3]),
              beta=as.numeric(args[4]), plot=FALSE)
write.tree(output$ToB, file=args[2])
```

where the first argument is the input gene trees, the second argument is the output tree of blobs, the third argument is the α hyperparameter for the tree-test, and the fourth argument is the β hyperparameter for the star-test. We ran TINNiK on each data set with varying the α and β values, as described in the main text.

TOB-QMC. We implemented TOB-QMC, within TREE-QMC (`https://github.com/molloy-lab/TREE-QMC`), as two steps. In step 1, we reconstruct a tree using TREE-QMC, with each branch annotated by (1) the p-value for the star-test based on average quartet Concordance Factors (qCFs) and (2) the minimum p-value found for the tree-test.

[3] `https://cran.r-project.org/web/packages/MSCquartets/vignettes/TINNIK.html`

```
./tree-qmc \
    -i <input gene trees> \
    -o <output tree with p-values for each branch> \
    --iter_limit_blob <iteration limit> --store_pvalue
```

The minimum p-value is found via heuristic search, which requires the user to specific the iteration limit. To run `TQMC-m`, we set the iteration limit to 5000 for 50 taxa and 20,000 for 100 taxa, as this corresponds to 2 times the number of taxa squared. To run `TQMC-f`, we set the iteration limit to be 8 times less than `TQMC-m` (i.e., we set the iteration to 625 for 50 taxa and 2500 for 100 taxa). To run exhaustive search, denoted `TQMC-s`, we set the iteration limit to `0`. Note that the star-tests and tree-tests are performed by invoking the same R code as TINNiK. In step 2, we contract branches in the output of step 1 based on the user-provided hyperparameters α and β.

```
${TREEQMC} \
    -i <input tree with p-values for each branch> \
    -o <output tree of blobs> \
    --blob --alpha <alpha threshold> \
    --beta <beta threshold> --load_pvalue
```

We ran this second step of TOB-QMC on each data set with varying the α and β values, as described in the main text. The runtime reported for TOB-QMC only includes the time for step (1) because step (2) is very fast. Note that TOB-QMC is under active development and these software commands are subject to change.

C.2 Additional Results

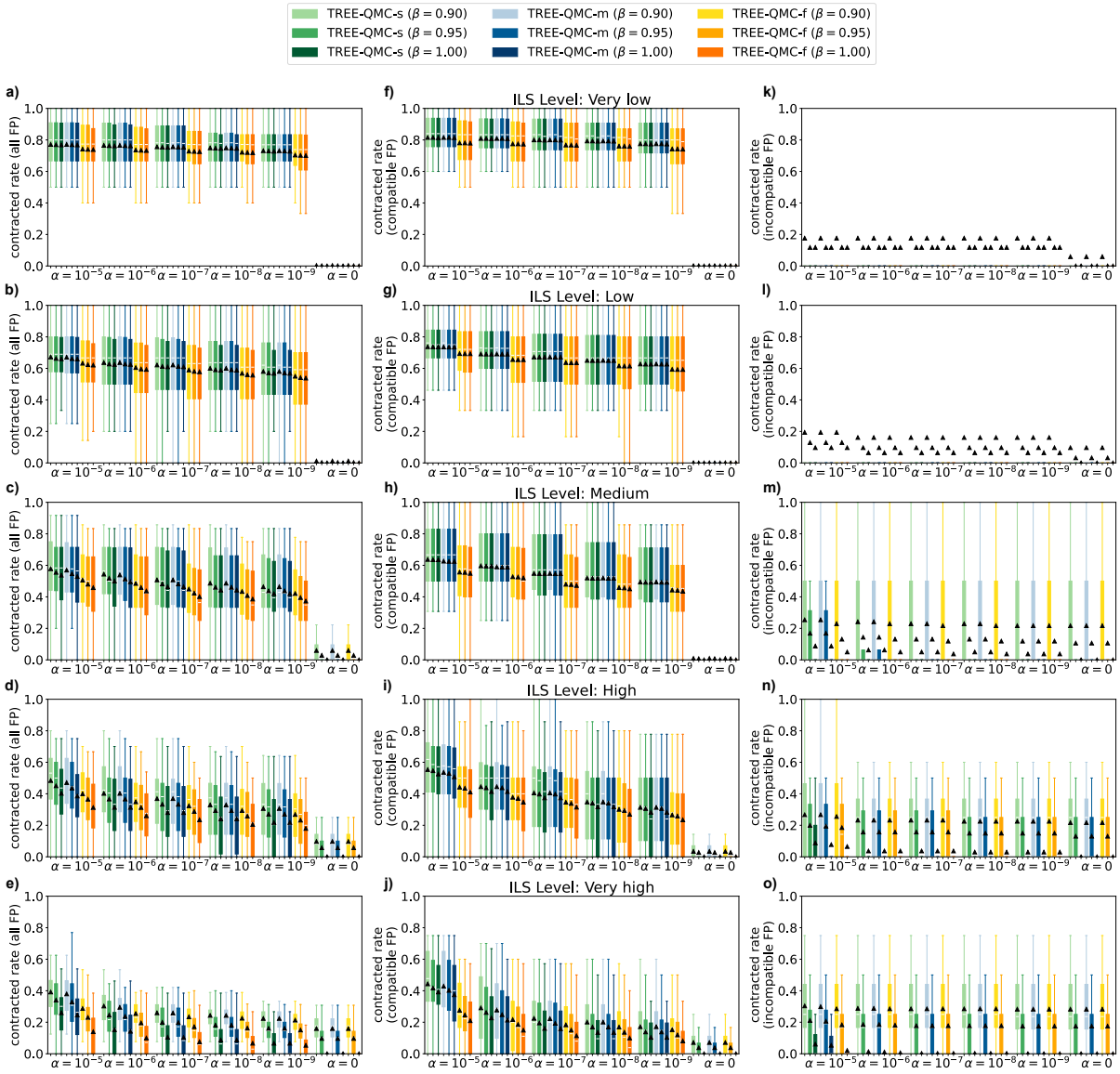


Figure C.1: **Impact of TOB-QMC's blob detection algorithm on level-1 networks with 50 taxa.** Rows correspond to ILS level. Boxplots show data for the 50 replicates. The refinement tree estimated by TREE-QMC is the same for each replicate; however, the branches contracted to form the tree of blobs changes based on the α hyperparameter (x-axis), the β hyperparameter (boxplot shade), and the method used to search for a minimum p-value for the tree-test (boxplot color). The left column shows the fraction of FP branches contracted (i.e., the fraction of branches in the estimated refinement tree that are missing from the tree of blobs that are contracted). The middle and right columns show the same quantity by separated based on whether the FP branch is compatible with the true tree of blobs or not.

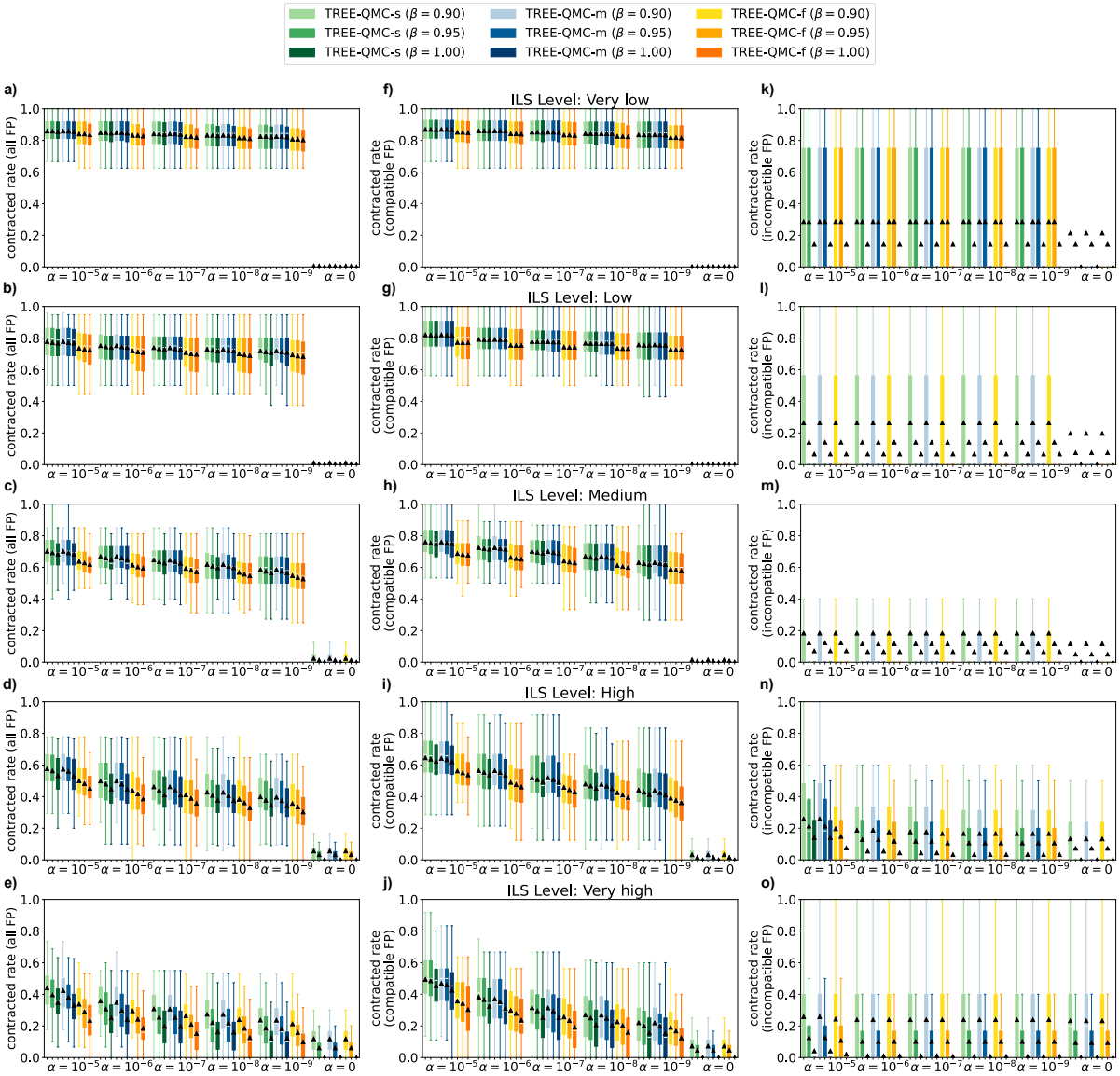


Figure C.2: **Impact of TOB-QMC's blob detection algorithm on level-2 networks with 50 taxa.**

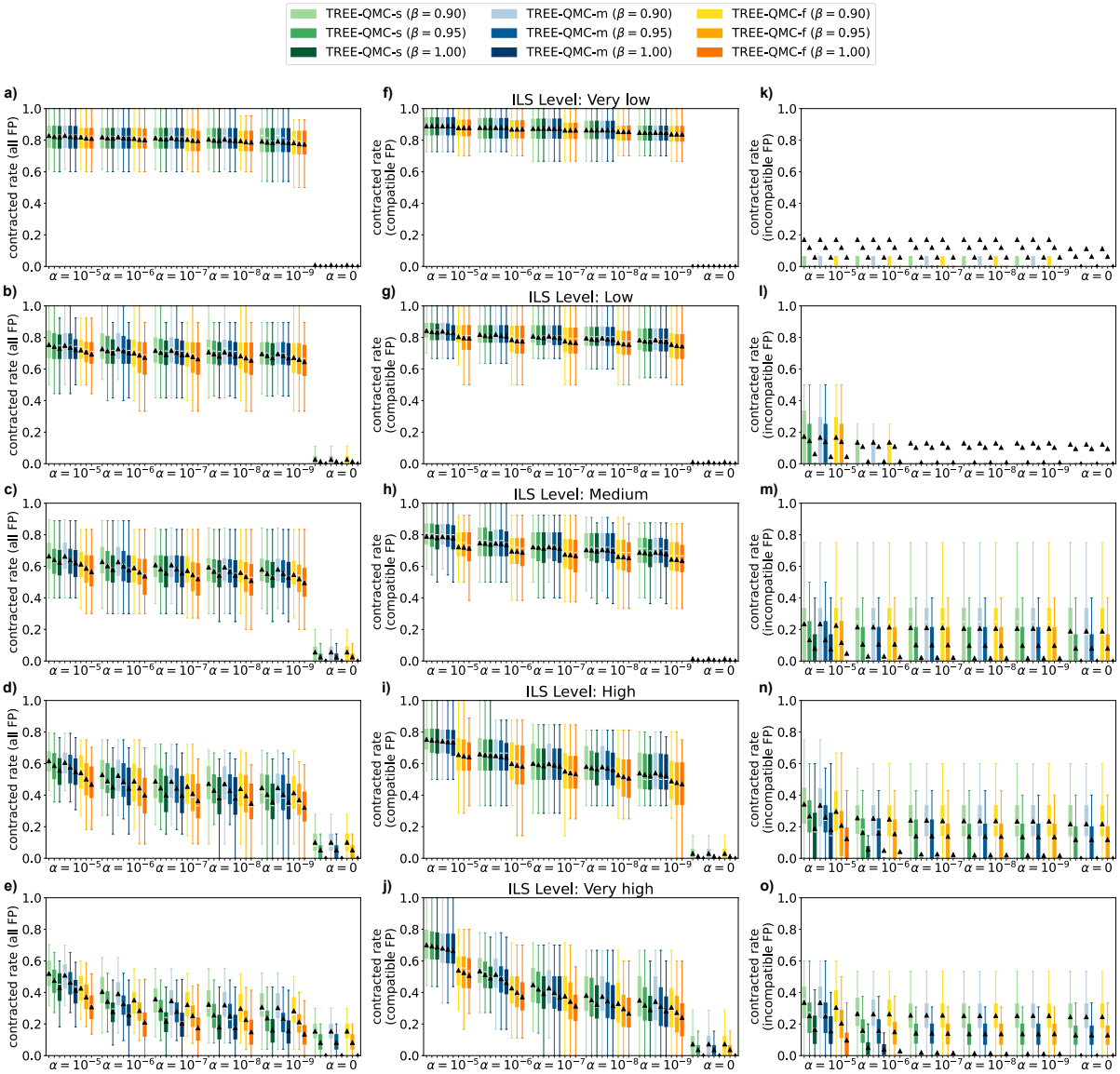


Figure C.3: **Impact of TOB-QMC's blob detection algorithm on level-1 networks with 100 taxa.**

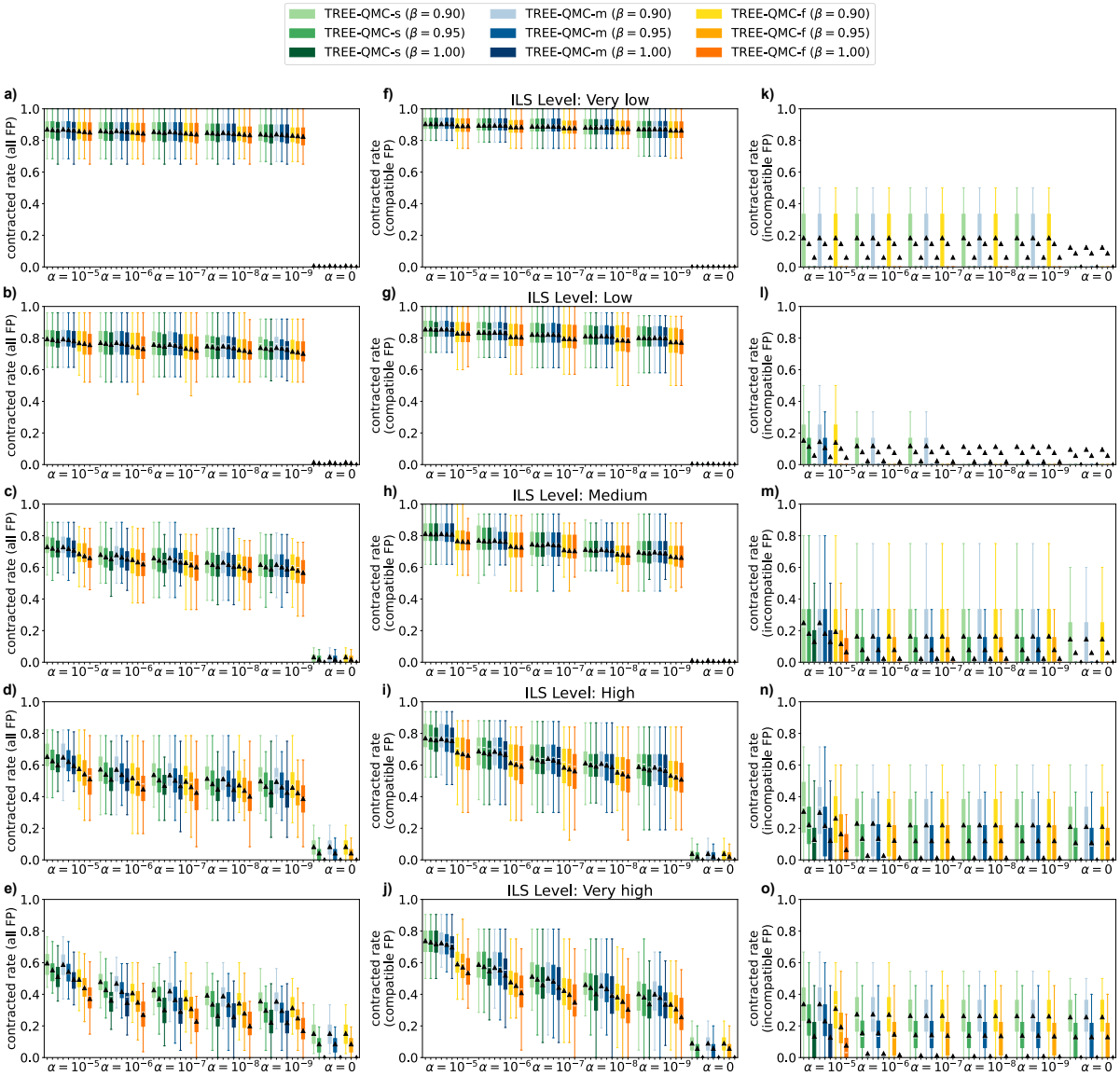


Figure C.4: **Impact of TOB-QMC's blob detection algorithm on level-2 networks with 100 taxa.**

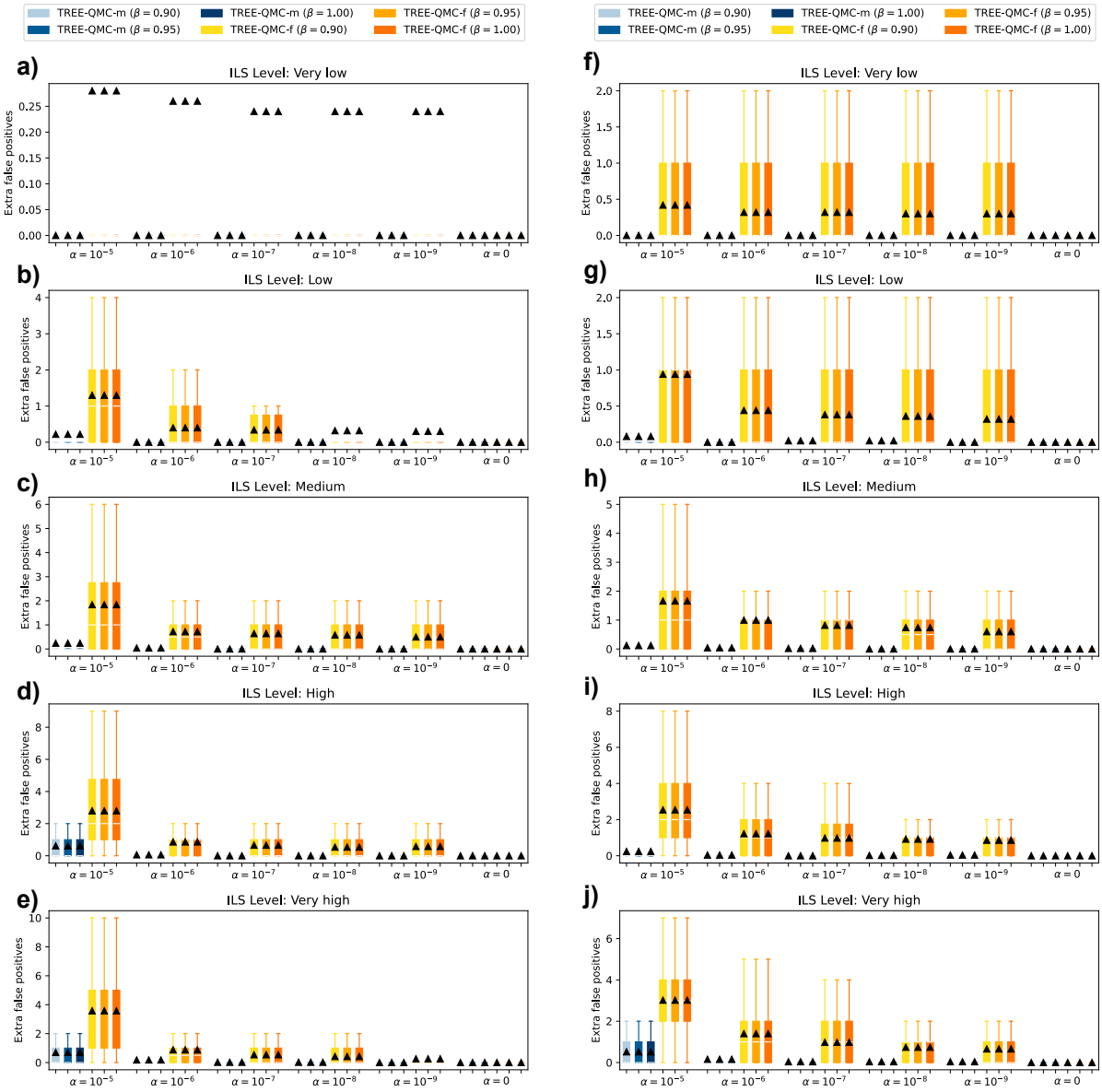


Figure C.5: **False positives introduced by the heuristic search algorithm for level-1 and level-2 networks with 50 taxa.** Rows correspond to ILS level. Left and right columns correspond to level-1 and level-2 networks, respectively. Boxplots show data for the 50 replicates. The refinement tree estimated by TREE-QMC is the same for each replicate; however, the branches contracted to form the tree of blobs changes based on the α hyperparameter (x-axis), the β hyperparameter (boxplot shade), and the heuristic used to search for a minimum p-value for the tree-test (boxplot color). The y-axis shows number of extra FP branches, compared to using exhaustive search for the minimum p-value.

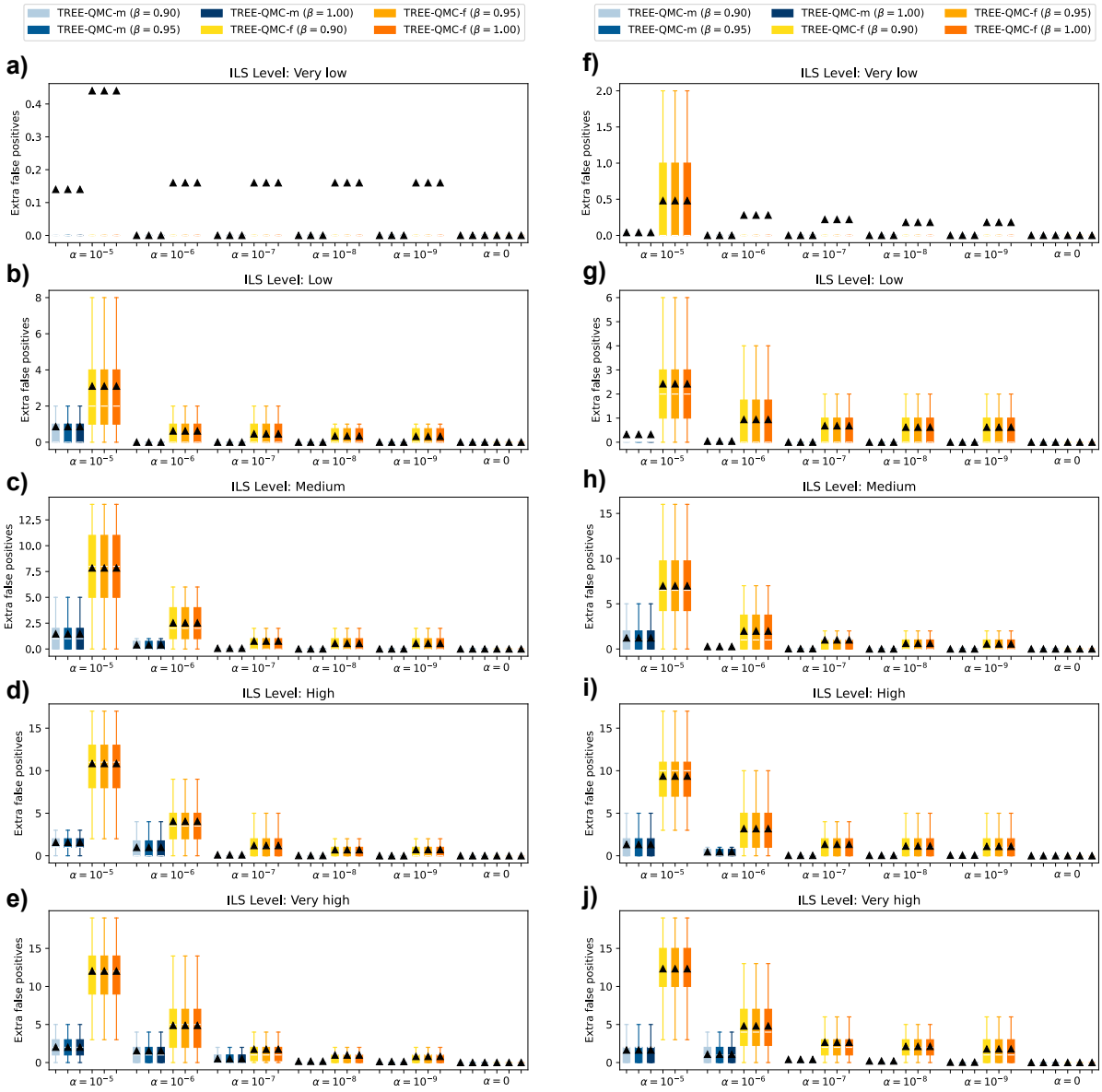


Figure C.6: **False positives introduced by the heuristic search algorithm for level-1 and level-2 networks with 100 taxa**.

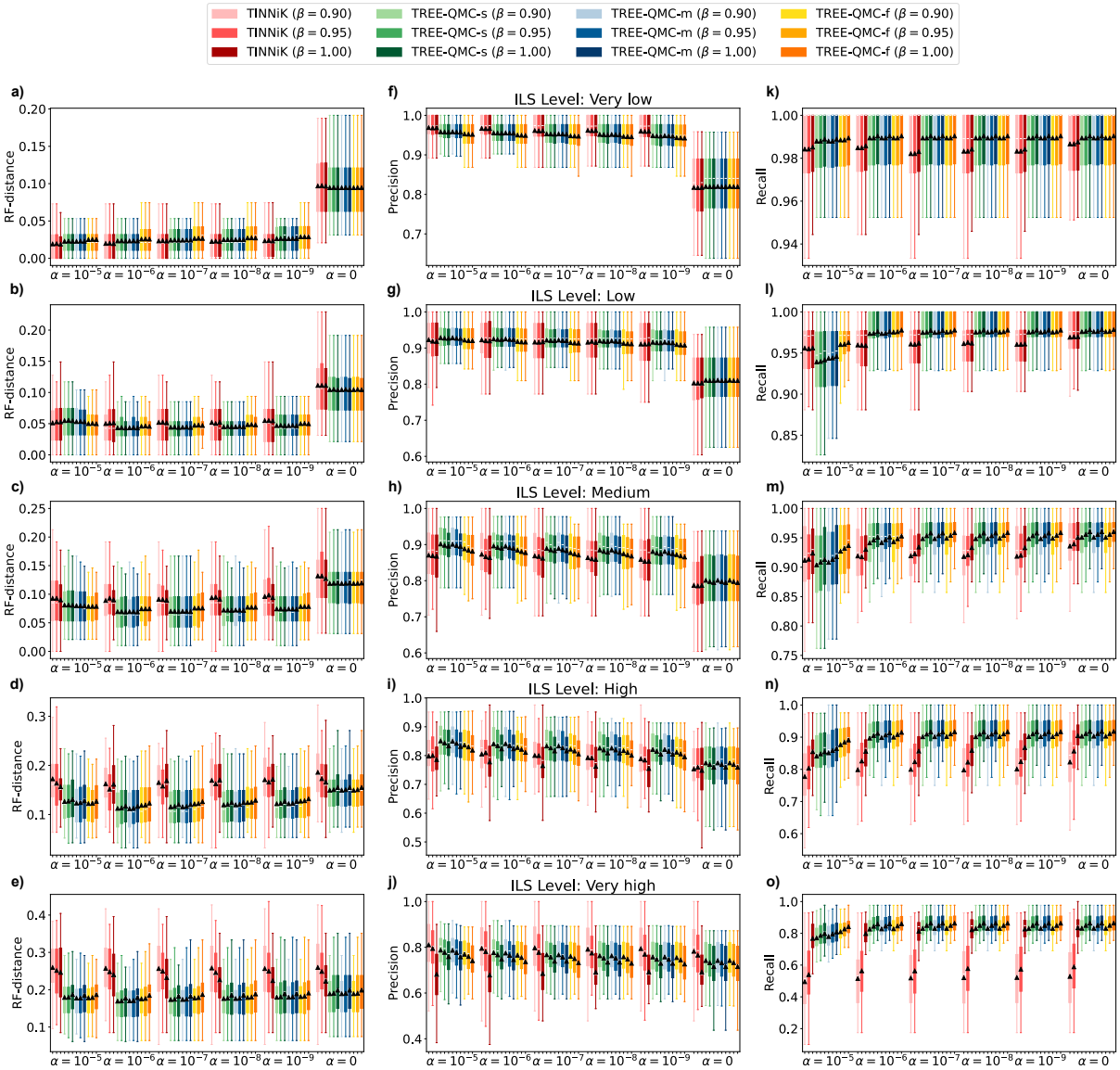


Figure C.7: **Topological error and accuracy of TOB-QMC and TINNiK on level-1 networks with 50 taxa.** Row corresponds to an ILS level, and columns correspond to error/accuracy metric on the *y*-axis. Boxplots show data for the 50 replicates. Red color corresponds to TINNiK; otherwise, color corresponds the method used to search for a minimum p-value for the tree-test by TREE-QMC. The β hyperparameter is indicated by boxplot shade, and the α hyperparameter is indicated on the *x*-axis.

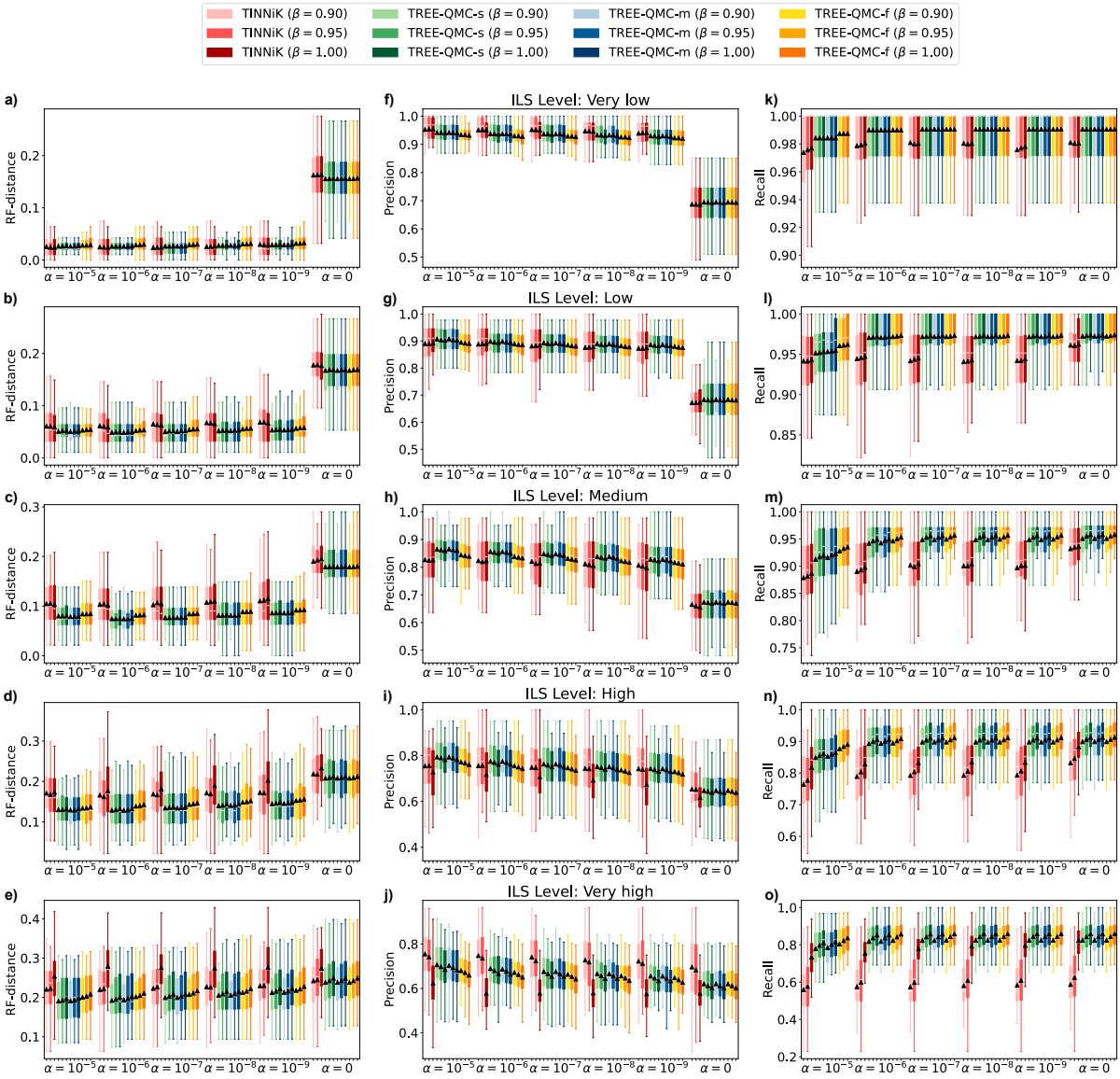


Figure C.8: **Accuracy of TREE-QMC and TINNiK on level-2 networks with 50 taxa.**

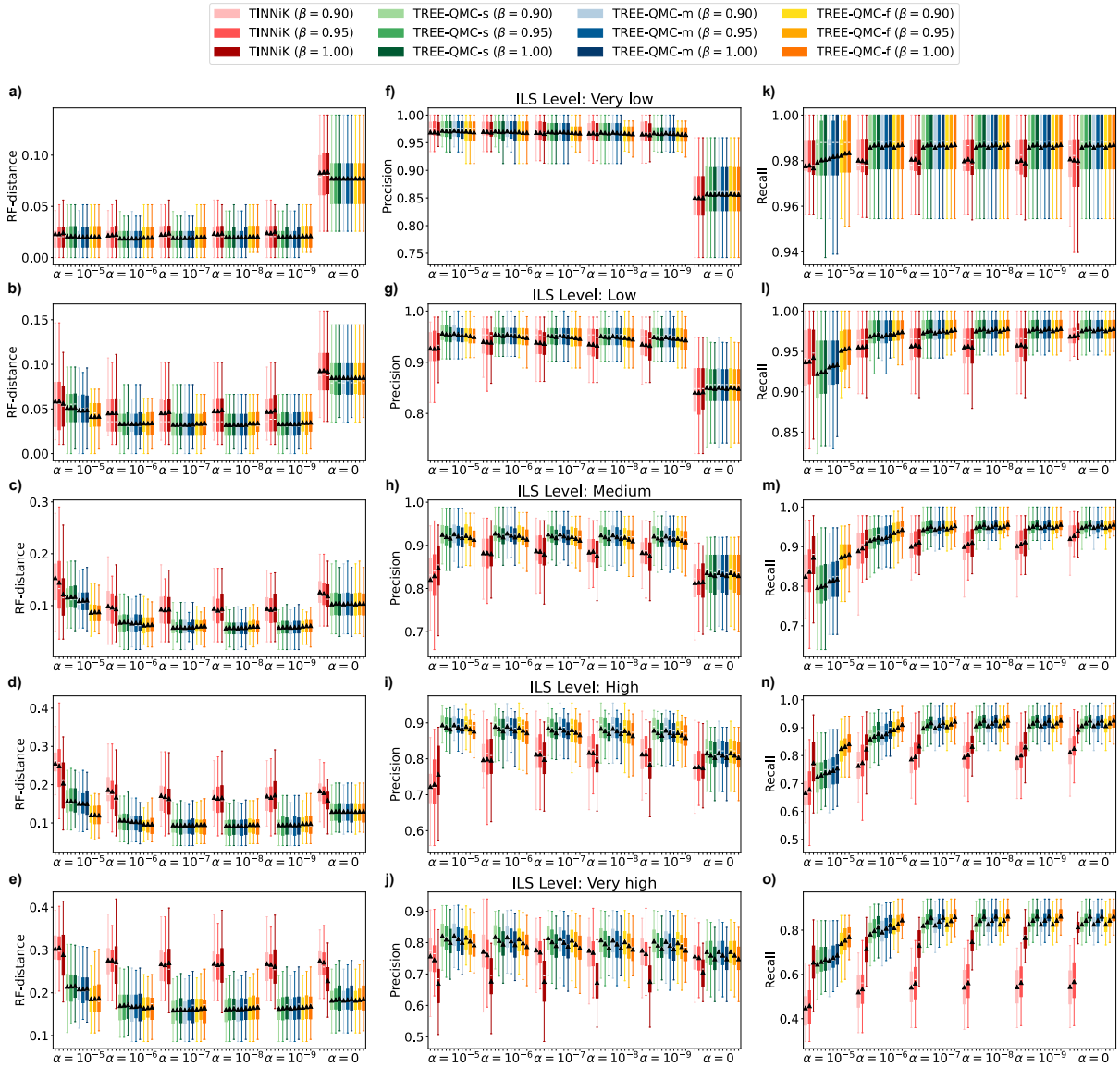


Figure C.9: **Accuracy of TREE-QMC and TINNiK on level-1 networks with 100 taxa.**

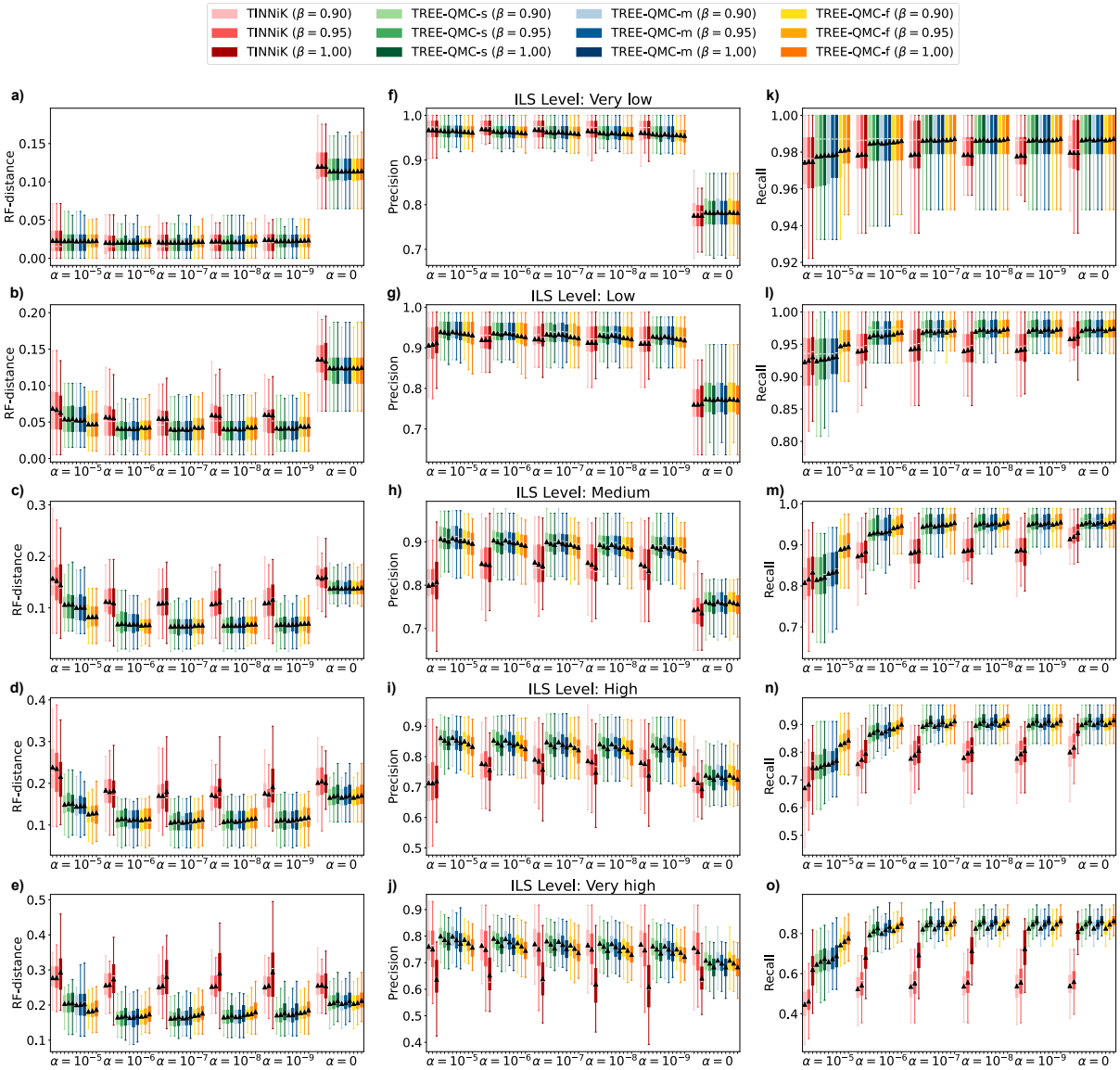


Figure C.10: **Accuracy of TREE-QMC and TINNiK on level-2 networks with 100 taxa.**

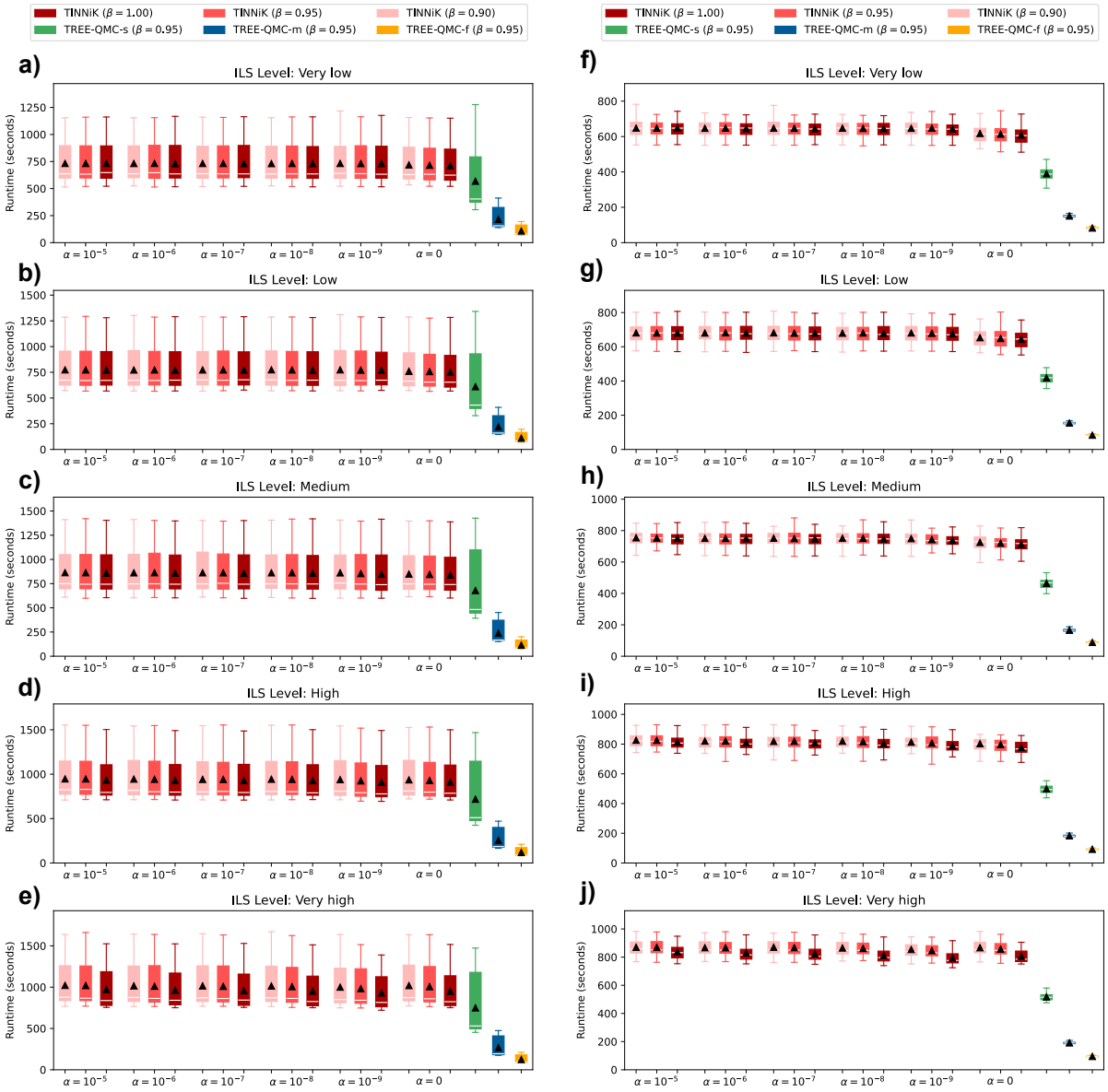


Figure C.11: **Runtime on level-1 and level-2 networks with 50 taxa**. Row corresponds to an ILS level. Left and right columns corresponds to level-1 and level-2 networks, respectively. Boxplots show data for the 50 replicates. The *y*-axis is method runtime in seconds. Red color corresponds to TINNiK, with the β hyperparameter indicated by shade and the α hyperparameter indicated by the *x*-axis. Otherwise, color corresponds the method used to search for a minimum p-value for the tree-test by TREE-QMC (note that the α and β hyperparameters do not impact the runtime of TREE-QMC).

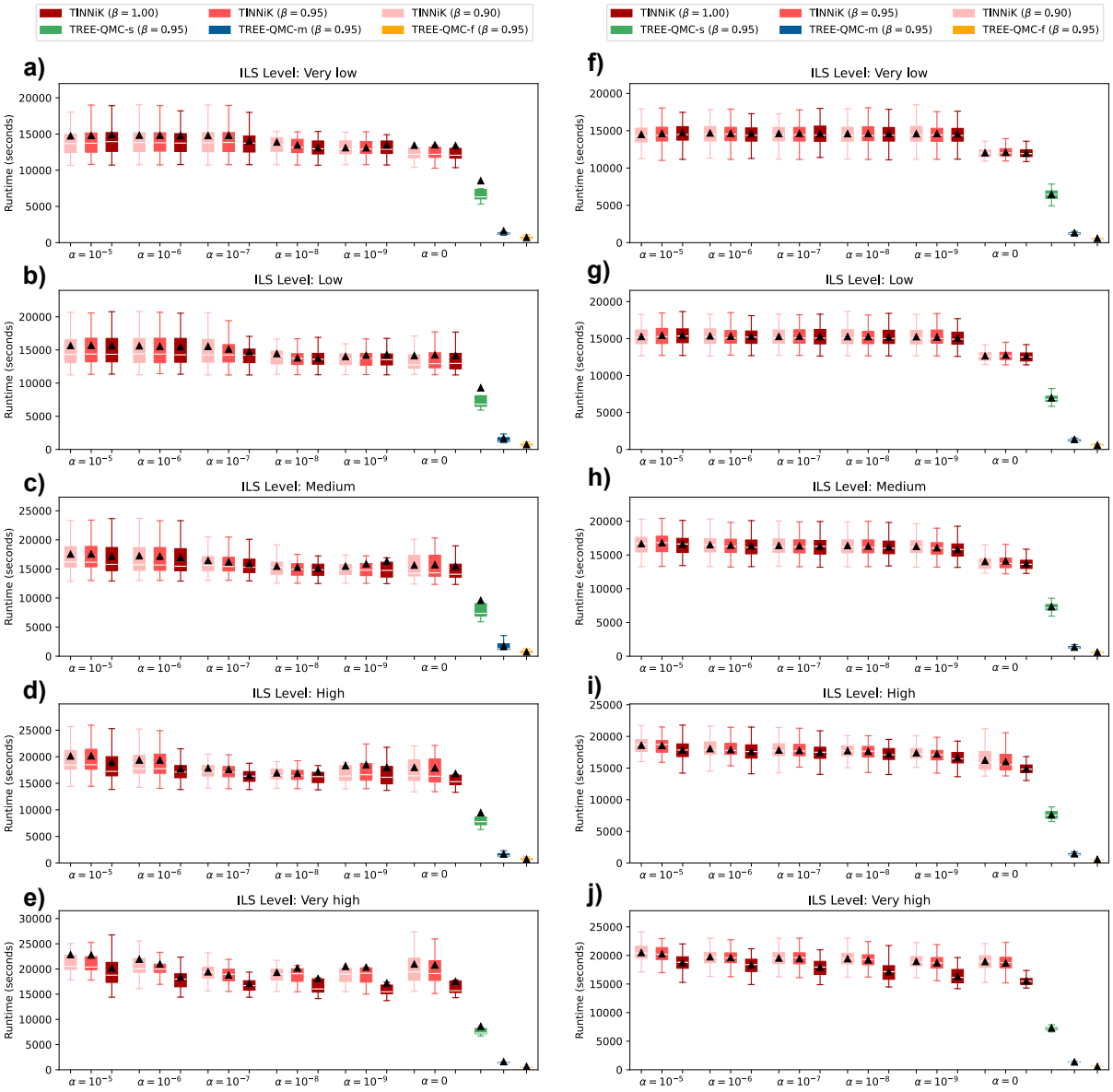


Figure C.12: **Runtime on level-1 and level-2 networks with 100 taxa**.

Bibliography

[1] Amir Abboud and Virginia Vassilevska Williams. Popular conjectures imply strong lower bounds for dynamic problems. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 434–443. IEEE, 2014.

[2] Elizabeth S Allman, Hector Baños, and John A Rhodes. NANUQ: a method for inferring species networks from gene trees under the coalescent model. *Algorithms for Molecular Biology*, 14(1):24, December 2019. doi: 10.1186/s13015-019-0159-2.

[3] Elizabeth S Allman, Jonathan D Mitchell, and John A Rhodes. Gene tree discord, simplex plots, and statistical tests under the coalescent. *Systematic Biology*, 71(4):929–942, 2021. doi: 10.1093/sysbio/syab008.

[4] Elizabeth S Allman, Hector Baños, Jonathan D Mitchell, and John A Rhodes. Tinnik: inference of the tree of blobs of a species network under the coalescent model. *Algorithms for Molecular Biology*, 19(1):23, 2024.

[5] Maria Anisimova, Manuel Gil, Jean-François Dufayard, Christophe Dessimoz, and Olivier Gascuel. Survey of Branch Support Methods Demonstrates Accuracy, Power, and Robustness of Fast Likelihood-based Approximation Schemes. *Systematic Biology*, 60(5):685–699, 05 2011. doi: 10.1093/sysbio/syr041.

[6] Shayesteh Arasti and Siavash Mirarab. *Median quartet tree search algorithms using optimal subtree prune and regraft*, 19(1), 2024. doi: 10.1186/s13015-024-00257-3.

[7] Eliran Avni, Reuven Cohen, and Sagi Snir. Weighted quartets phylogenetics. *Systematic Biology*, 64(2):233–242, 11 2014. ISSN 1063-5157. doi: 10.1093/sysbio/syu087.

[8] Arturs Backurs and Piotr Indyk. Edit distance cannot be computed in strongly subquadratic time (unless seth is false). In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 51–58, 2015.

[9] Hector Baños. Identifying species network features from gene tree quartets under the coalescent model. *Bulletin of Mathematical Biology*, 81(2):494–534, 2019. doi: 10.1007/s11538-018-0485-4.

[10] David A. Baum. Concordance trees, concordance factors, and the exploration of reticulate genealogy. *TAXON*, 56(2):417–426, 2007. doi: 10.1002/tax.562013.

[11] Olaf RP Bininda-Emonds. Inferring the Tree of Life: chopping a phylogenomic problem down to size? *BMC Biology*, 9(1):59, 2011.

[12] Paul D Blischak, Julia Chifman, Andrea D Wolfe, and Laura S Kubatko. HyDe: A python package for genome-scale hybridization detection. *Systematic Biology*, 67(5):821–829, 03 2018. doi: 10.1093/sysbio/syy023.

[13] E.L. Braun and R.T. Kimball. Data Types and the Phylogeny of Neoaves. *Birds*, 2(1):1–22, 2021. doi: 10.3390/birds2010001.

[14] Gerth Stølting Brodal, Rolf Fagerberg, Thomas Mailund, Christian N. S. Pedersen, and Andreas Sand. Efficient algorithms for computing the triplet and quartet distance between trees of arbitrary degree. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '13, page 1814–1832, USA, 2013. Society for Industrial and Applied Mathematics. ISBN 9781611972511.

[15] David Bryant and Vincent Moulton. Neighbor-net: an agglomerative method for the construction of phylogenetic networks. *Molecular biology and evolution*, 21(2):255–265, 2004.

[16] Samuel Burer, Renato DC Monteiro, and Yin Zhang. Rank-two relaxation heuristics for max-cut and other binary quadratic programs. *SIAM Journal on Optimization*, 12(2):503–521, 2002.

[17] Zhen Cao, Meng Li, Huw Ogilvie, and Luay Nakhleh. The impact of model misspecification on phylogenetic network inference. *Bulletin of the Society of Systematic Biologists*, 3(1), 2024. doi: 10.18061/bssb.v3i1.9553.

[18] Julia Chifman and Laura Kubatko. Quartet inference from SNP data under the coalescent model. *Bioinformatics*, 30(23):3317–3324, 2014. doi: 10.1093/bioinformatics/btu530.

[19] A. Cloutier, T. B. Sackton, P. Grayson, M. Clamp, A. J. Baker, and S. V. Edwards. Whole-genome analyses resolve the phylogeny of flightless birds (Palaeognathae) in the presence of an empirical anomaly zone. *Systematic Biology*, 68:937–955, 2019.

[20] Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. *Introduction to Algorithms*. MIT Press, 3rd edition, 2009.

[21] Tauana Junqueira Cunha, James Davis Reimer, and Gonzalo Giribet. Investigating Sources of Conflict in Deep Phylogenomics of Vetigastropod Snails. *Systematic Biology*, 71(4): 1009–1022, 10 2021. doi: 10.1093/sysbio/syab071.

[22] J. H. Degnan and L. A. Salter. Gene tree distributions under the coalescent process. *Evolution*, 59:24–37, 2005.

[23] Payam Dibaeinia, Shayan Tabe-Bordbar, and Tandy Warnow. FASTRAL: improving scalability of phylogenomic analysis. *Bioinformatics*, 37(16):2317–2324, 02 2021. doi: 10.1093/bioinformatics/btab093.

[24] Vu Dinh and Hector Baños. Misspecification strikes: Astral can mislead in the presence of hybridization, even for nonanomalous scenarios. *Molecular Biology and Evolution*, 42(3): msaf049, 2025. doi: 10.1093/molbev/msaf049.

[25] Andreas WM Dress and Daniel H Huson. Constructing splits graphs. *IEEE/ACM transactions on Computational Biology and Bioinformatics*, 1(3):109–115, 2004.

[26] Iain Dunning, Swati Gupta, and John Silberholz. What works best when? A systematic evaluation of heuristics for Max-Cut and QUBO. *INFORMS Journal of Computing*, 30(3): 421–624, 2018. doi: 10.1287/ijoc.2017.0798.

[27] Nathaniel B. Edelman, Paul B. Frandsen, Miriam Miyagi, Bernardo Clavijo, John Davey, Rebecca B. Dikow, Gonzalo García-Accinelli, Steven M. Van Belleghem, Nick Patterson, Daniel E. Neafsey, Richard Challis, Sujai Kumar, Gilson R. P. Moreira, Camilo Salazar, Mathieu Chouteau, Brian A. Counterman, Riccardo Papa, Mark Blaxter, Robert D. Reed, Kanchon K. Dasmahapatra, Marcus Kronforst, Mathieu Joron, Chris D. Jiggins, W. Owen McMillan, Federica Di Palma, Andrew J. Blumberg, John Wakeley, David Jaffe, and James Mallet. Genomic architecture and introgression shape a butterfly radiation. *Science*, 366 (6465):594–599, 2019. doi: 10.1126/science.aaw2090.

[28] Scott V Edwards. Is a new and general theory of molecular systematics emerging? *Evolution*, 63(1):1–19, 2009.

[29] James S Farris. Methods for computing wagner trees. *Systematic Biology*, 19(1):83–92, 1970.

[30] Joseph Felsenstein. Cases in which parsimony or compatibility methods will be positively misleading. *Systematic Biology*, 27(4):401–410, 1978. doi: 10.1093/sysbio/27.4.401.

[31] Joseph Felsenstein. Evolutionary trees from DNA sequences: A maximum likelihood approach. *Journal of Molecular Evolution*, 17(6):368–376, 1981. doi: 10.1007/BF01734359.

[32] Shaohong Feng, Ming Bai, Iker Rivas-González, Cai Li, Shiping Liu, Yijie Tong, Haidong Yang, Guangji Chen, Duo Xie, Karen E. Sears, Lida M. Franco, Juan Diego Gaitan-Espitia, Roberto F. Nespolo, Warren E. Johnson, Huanming Yang, Parice A. Brandies, Carolyn J. Hogg, Katherine Belov, Marilyn B. Renfree, Kristofer M. Helgen, Jacobus J. Boomsma, Mikkel Heide Schierup, and Guojie Zhang. Incomplete lineage sorting and phenotypic evolution in marsupials. *Cell*, 185(10):1646–1660.e18, 2022. doi: 10.1016/j.cell.2022.03.034.

[33] Mareike Fischer, Lina Herbst, Sophie Kersting, Luise Kühn, and Kristina Wicke. *Tree balance indices - a comprehensive survey*. Springer, Berlin, 2023. ISBN 978-3-031-39799-8.

[34] Ronald Aylmer Fisher. *The genetical theory of natural selection: a complete variorum edition*. Oxford University Press, 1999.

[35] Walter M Fitch. Toward defining the course of evolution: minimum change for a specific tree topology. *Systematic Biology*, 20(4):406–416, 1971.

[36] John Fogg, Elizabeth S Allman, and Cécile Ané. PhyloCoalSimulations: A simulator for network multispecies coalescent models, including a new extension for the inheritance of gene flow. *Systematic Biology*, 72(5):1171–1179, 2023. doi: 10.1093/sysbio/syad030.

[37] L.R. Foulds and R.L. Graham. The steiner problem in phylogeny is np-complete. *Advances in Applied Mathematics*, 3(1):43–49, 1982. doi: 10.1016/S0196-8858(82)80004-3.

[38] J. Gatesy and M. S. Springer. Concatenation versus coalescence versus "concatalescence".

Proceedings of the National Academy of Sciences USA, 110:E1179, 2013. doi: 10.1073/pnas.1221121110.

[39] John Gatesy and Mark S. Springer. Phylogenetic analysis at deep timescales: Unreliable gene trees, bypassed hidden support, and the coalescence/ concatalescence conundrum. *Molecular Phylogenetics and Evolution*, 80:231–266, 2014. doi: 10.1016/j.ympev.2014.08.013.

[40] John Gatesy, Robert W. Meredith, Jan E. Janecka, Mark P. Simmons, William J. Murphy, and Mark S. Springer. Resolution of a concatenation/coalescence kerfuffle: partitioned coalescence support and a robust family-level tree for mammalia. *Cladistics*, 33:295–332, 2017. doi: 10.1111/cla.12170.

[41] John Gatesy, Daniel B. Sloan, Jessica M. Warren, Richard H. Baker, Mark P. Simmons, and Mark S. Springer. Partitioned coalescence support reveals biases in species-tree methods and detects gene trees that determine phylogenomic conflicts. *Molecular Phylogenetics and Evolution*, 139:106539, 2019. doi: 10.1016/j.ympev.2019.106539.

[42] D. Gusfield. *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology*. Cambridge University Press, Cambridge, United Kingdom, 1997.

[43] Dan Gusfield, Vikas Bansal, Vineet Bafna, and Yun S. Song. A decomposition theory for phylogenetic networks and incompatible characters. *Journal of Computational Biology*, 14 (10):1247–1272, 2007. doi: 10.1089/cmb.2006.0137.

[44] John Burdon Haldane. *The causes of evolution*, volume 5. Princeton University Press, 1990.

[45] Yunheng Han and Erin K Molloy. Improving quartet graph construction for scalable and accurate species tree estimation from gene trees. *Genome Research*, 33(7):1042–1105, 2023. doi: 10.1101/gr.277629.122.

[46] Yunheng Han and Erin K. Molloy. Quartets enable statistically consistent estimation of cell lineage trees under an unbiased error and missingness model. *Algorithms for Molecular Biology*, 18:19, 2024. doi: 10.1186/s13015-023-00248-w.

[47] Yunheng Han and Erin K Molloy. Improved robustness to gene tree incompleteness, estimation errors, and systematic homology errors with weighted tree-qmc. *Systematic Biology*, page syaf009, 2025. doi: 10.1093/sysbio/syaf009.

[48] Monika Henzinger, Sebastian Krinninger, Danupon Nanongkai, and Thatchaphol Saranurak. Unifying and strengthening hardness for dynamic problems via the online matrix-vector multiplication conjecture. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 21–30, 2015.

[49] Niels Holtgrefe, Katharina T Huber, Leo van Iersel, Mark Jones, Samuel Martin, and Vincent Moulton. Squirrel: Reconstructing semi-directed phylogenetic level-1 networks from four-leaved networks or sequence alignments. *Molecular Biology and Evolution*, 42(4): msaf067, 2025. ISSN 1537-1719. doi: 10.1093/molbev/msaf067.

[50] John Hopcroft and Robert Tarjan. Algorithm 447: efficient algorithms for graph manipulation. *Communications of the ACM*, 16(6):372—-378, 1973. doi: 10.1145/362248.362272.

[51] P. A. Hosner, B. C. Faircloth, T. C. Glenn, E. L. Braun, and R. T. Kimball. Avoiding missing data biases in phylogenomic inference: An empirical study in the landfowl (Aves: Galliformes). *Mol. Biol. Evol.*, 33:1110–1125, 2016.

[52] Torsten Hothorn, Henric Winell, Kurt Hornik, Mark A. van de Wiel, and Achim Zeileis. *coin: Conditional Inference Procedures in a Permutation Test Framework*, 2024. URL `https://CRAN.R-project.org/package=coin`. R package version 1.4-3.

[53] Mazharul Islam, Kowshika Sarker, Trisha Das, Rezwana Reaz, and Md. Shamsuzzoha

Bayzid. STELAR: a statistically consistent coalescent-based species tree estimation method by maximizing triplet consistency. *BMC Genomics*, 21(1):136, 2020. doi: 10.1186/s12864-020-6519-y.

[54] Erich D. Jarvis, Siavash Mirarab, et al. Whole-genome analyses resolve early branches in the tree of life of modern birds. *Science*, 346(6215):1320–1331, 2014. doi: 10.1126/science.1253451.

[55] Joshua A Justison, Claudia Solis-Lemus, and Tracy A Heath. SiPhyNetwork: An R package for simulating phylogenetic networks. *Methods in Ecology and Evolution*, 14(7):1687–1698, 2023. doi: 10.1111/2041-210X.14116.

[56] Richard M. Karp. *Reducibility among Combinatorial Problems*, pages 85–103. Springer, Boston, MA, 1972. doi: 10.1007/978-1-4684-2001-2_9.

[57] Sophie J. Kersting, Kristina Wicke, and Mareike Fischer. Tree balance in phylogenetic models. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 380(1919):20230303, 2025. doi: 10.1098/rstb.2023.0303.

[58] Alexander Knyshov, Yana Hrytsenko, Robert Literman, and Rachel S. Schwartz. Interrogating genomic data in the phylogenetic placement of treeshrews reveals potential sources of conflict. *bioRxiv*, 2022. doi: 10.1101/2021.11.18.469131.

[59] Nathan Kolbow, Sungsik Kong, and Claudia Solís-Lemus. A method for massively scalable phylogenetic network inference. *bioRxiv*, 2025. doi: 10.1101/2025.05.05.652278.

[60] Alexey M Kozlov, Andre J Aberer, and Alexandros Stamatakis. ExaML version 3: a tool for phylogenomic analyses on supercomputers. *Bioinformatics*, 31(15):2577–2579, August 2015.

[61] Laura Salter Kubatko and James H. Degnan. Inconsistency of phylogenetic estimates from concatenated data under coalescence. *Systematic Biology*, 56(1):17–24, 2007. doi: 10.1080/10635150601146041.

[62] Manuel Lafond and Celine Scornavacca. On the weighted quartet consensus problem. *Theoretical Computer Science*, 769:1–17, 2019. doi: 10.1016/j.tcs.2018.10.005.

[63] Bret R. Larget, Satish K. Kotha, Colin N. Dewey, and Cécile Ané. BUCKy: Gene tree/species tree reconciliation with bayesian concordance analysis. *Bioinformatics*, 26(22): 2910–2911, 2010. doi: 10.1093/bioinformatics/btq539.

[64] Brandon Legried, Erin K. Molloy, Tandy Warnow, and Sébastien Roch. Polynomial-time statistical estimation of species trees under gene duplication and loss. *Journal of Computational Biology*, 28(5):452–468, 2021. doi: 10.1089/cmb.2020.0424.

[65] Baqiao Liu and Tandy Warnow. Weighted astrid: fast and accurate species trees from weighted internode distances. *Algorithms for Molecular Biology*, 18:6, 2023. doi: 10.1186/s13015-023-00230-6.

[66] Liang Liu and Lili Yu. Estimating species trees from unrooted gene trees. *Systematic Biology*, 60(5):661–667, 2011. doi: 10.1093/sysbio/syr027.

[67] Liang Liu, Lili Yu, and Scott V. Edwards. A maximum pseudo-likelihood approach for estimating species trees under the coalescent model. *BMC Evolutionary Biology*, 10:302, 2010. doi: 10.1186/1471-2148-10-302.

[68] Wayne P Maddison. Gene trees in species trees. *Systematic biology*, 46(3):523–536, 1997.

[69] Mahim Mahbub, Zahin Wahab, Rezwana Reaz, M Saifur Rahman, and Md Shamsuzzoha Bayzid. wQFM: highly accurate genome-scale species tree estimation from weighted quartets. *Bioinformatics*, 37(21):3734–3743, 06 2021. doi: 10.1093/bioinformatics/btab428.

[70] Uyen Mai and Siavash Mirarab. Treeshrink: fast and accurate detection of outlier long branches in collections of phylogenetic trees. *BMC Genomics*, 19(5):272, 2018.

[71] John E McCormack, Brant C Faircloth, Nicholas G Crawford, Patricia Adair Gowaty, Robb T Brumfield, and Travis C Glenn. Ultraconserved elements are novel phylogenomic markers that resolve placental mammal phylogeny when combined with species-tree analysis. *Genome Research*, 22(4):746–754, 2012. doi: 10.1101/gr.125864.111.

[72] Monnie McGee. Case for omitting tied observations in the two-sample t-test and the wilcoxon-mann-whitney test. *PLOS ONE*, 13(7):1–19, 07 2018. doi: 10.1371/journal.pone.0200837.

[73] Kelly A. Meiklejohn, Brant C. Faircloth, Travis C. Glenn, Rebecca T. Kimball, and Edward L. Braun. Analysis of a rapid evolutionary radiation using ultraconserved elements: Evidence for a bias in some multispecies coalescent methods. *Systematic Biology*, 65(4):612–627, 2016. doi: 10.1093/sysbio/syw014.

[74] Fábio K. Mendes and Matthew W. Hahn. Why concatenation fails near the anomaly zone. *Systematic Biology*, 67(1):158–169, 2017. doi: 10.1093/sysbio/syx063.

[75] Chen Meng and Laura Salter Kubatko. Detecting hybrid speciation in the presence of incomplete lineage sorting using gene tree incongruence: A model. *Theoretical Population Biology*, 75(1):35–45, 2009. doi: 10.1016/j.tpb.2008.10.004.

[76] Bui Quang Minh, Heiko A Schmidt, Olga Chernomor, Dominik Schrempf, Michael D Woodhams, Arndt von Haeseler, and Robert Lanfear. IQ-TREE 2: New Models and Efficient Methods for Phylogenetic Inference in the Genomic Era. *Molecular Biology and Evolution*, 37(5):1530–1534, 2020. doi: 10.1093/molbev/msaa015.

[77] S. Mirarab, R. Reaz, Md. S. Bayzid, T. Zimmermann, M. S. Swenson, and T. Warnow.

ASTRAL: genome-scale coalescent-based species tree estimation. *Bioinformatics*, 30(17): i541–i548, 2014. doi: 10.1093/bioinformatics/btu462.

[78] Siavash Mirarab and Tandy Warnow. ASTRAL-II: coalescent-based species tree estimation with many hundreds of taxa and thousands of genes. *Bioinformatics*, 31(12):i44–i52, 2015. doi: 10.1093/bioinformatics/btv234.

[79] Siavash Mirarab, Md. Shamsuzzoha Bayzid, Bastien Boussau, and Tandy Warnow. Statistical binning enables an accurate coalescent-based estimation of the avian tree. *Science*, 346 (6215):1250463, 2014. doi: 10.1126/science.1250463.

[80] Siavash Mirarab, Iker Rivas-González, Shaohong Feng, Josefin Stiller, Qi Fang, Uyen Mai, Glenn Hickey, Guangji Chen, Nadolina Brajuka, Olivier Fedrigo, Giulio Formenti, Jochen B. W. Wolf, Kerstin Howe, Agostinho Antunes, Mikkel H. Schierup, Benedict Paten, Erich D. Jarvis, Guojie Zhang, and Edward L. Braun. A region of suppressed recombination misleads neoavian phylogenomics. *Proceedings of the National Academy of Sciences*, 121 (15):e2319506121, 2024. doi: 10.1073/pnas.2319506121.

[81] Erin K Molloy and Tandy Warnow. To include or not to include: The impact of gene filtering on species tree estimation methods. *Systematic Biology*, 67(2):285–303, 2018. doi: 10.1093/sysbio/syx077.

[82] Erin K Molloy and Tandy Warnow. Statistically consistent divide-and-conquer pipelines for phylogeny estimation using NJMerge. *Algorithms for Molecular Biology*, 14(1):14, 2019. doi: 10.1186/s13015-019-0151-x.

[83] Erin K Molloy and Tandy Warnow. TreeMerge: a new method for improving the scalability of species tree estimation methods. *Bioinformatics*, 35(14):i417–i426, 2019. doi: 10.1093/bioinformatics/btz344.

[84] Erin K. Molloy, John Gatesy, and Mark S. Springer. Theoretical and practical considerations when using retroelement insertions to estimate species trees in the anomaly zone. *Systematic Biology*, 71(3):721–740, 2022. doi: 10.1093/sysbio/syab086.

[85] Benoit Morel, Alexey M Kozlov, and Alexandros Stamatakis. ParGenes: a tool for massively parallel model selection and phylogenetic tree inference on thousands of genes. *Bioinformatics*, 35(10):1771–1773, 2018. doi: 10.1093/bioinformatics/bty839.

[86] Benoit Morel, Tom A Williams, and Alexandros Stamatakis. Asteroid: a new algorithm to infer species trees from gene trees under high proportions of missing data. *Bioinformatics*, 29(1):btac832, 2023. doi: 10.1093/bioinformatics/btac832.

[87] Lam-Tung Nguyen, Heiko A Schmidt, Arndt Von Haeseler, and Bui Quang Minh. Iq-tree: a fast and effective stochastic algorithm for estimating maximum-likelihood phylogenies. *Molecular biology and evolution*, 32(1):268–274, 2015.

[88] Michael Nute, Jed Chou, Erin K. Molloy, and Tandy Warnow. The performance of coalescent-based species tree estimation methods under models of missing data. *BMC Genomics*, 19(Suppl 5):286, 2018. doi: 10.1186/s12864-018-4619-8.

[89] One Thousand Plant Transcriptomes Initiative. One thousand plant transcriptomes and the phylogenomics of green plants. *Nature*, 574(7780):679–685, 2019. doi: 10.1038/s41586-019-1693-2.

[90] Jente Ottenburghs. How common is hybridization in birds? *Journal of Ornithology*, 164 (4):913–920, 2023. doi: 10.1007/s10336-023-02080-w.

[91] P Pamilo and M Nei. Relationships between gene trees and species trees. *Molecular Biology and Evolution*, 5(5):568–583, 1988. doi: 10.1093/oxfordjournals.molbev.a040517.

[92] Nick Patterson, Priya Moorjani, Yontao Luo, Swapan Mallick, Nadin Rohland, Yiping Zhan, Teri Genschoreck, Teresa Webster, and David Reich. Ancient admixture in human history. *Genetics*, 192(3):1065–1093, 2012. doi: 10.1534/genetics.112.145037.

[93] Pexels. Chimpanzee image, . URL `https://www.pexels.com/photo/photography-of-monkey-1238352/`. Downloaded in March 2020.

[94] Pexels. Gorilla image, . URL `https://www.pexels.com/photo/close-up-photo-of-black-gorilla-1851537/`. Downloaded in March 2020.

[95] Pexels. Human image, . URL `https://www.pexels.com/photo/man-wearing-black-zip-jacket-holding-smartphone-surrounded-by-grey-concrete-buildings-775091/`. Downloaded in March 2020.

[96] Pexels. Lemur image, . URL `https://www.pexels.com/photo/white-and-gray-lemur-on-brown-surface-33149/`. Downloaded in March 2020.

[97] Pexels. Orangutan image, . URL `https://www.pexels.com/photo/brown-and-orange-monkey-52530/`. Downloaded in March 2020.

[98] II Platt II, Roy N, Brant C Faircloth, Kevin A M Sullivan, Troy J Kieran, Travis C Glenn, Michael W Vandewege, Jr. Lee, Thomas E, Robert J Baker, Richard D Stevens, and David A Ray. Conflicting Evolutionary Histories of the Mitochondrial and Nuclear Genomes in New World Myotis Bats. *Systematic Biology*, 67(2):236–249, 2017. ISSN 1063-5157. doi: 10.1093/sysbio/syx070.

[99] J. W. Pratt. Remarks on zeros and ties in the wilcoxon signed rank procedures. *Journal of the American Statistical Association*, 54(287):655—667, 1959. doi: 10.1080/01621459.1959.10501526.

[100] Morgan N. Price, Paramvir S. Dehal, and Adam P. Arkin. FastTree 2 – approximately maximum-likelihood trees for large alignments. *PLOS ONE*, 5(3):1–10, 03 2010. doi: 10.1371/journal.pone.0009490.

[101] Maryam Rabiee, Erfan Sayyari, and Siavash Mirarab. Multi-allele species reconstruction using ASTRAL. *Molecular Phylogenetics and Evolution*, 130:286–296, 2019. doi: 10.1016/j.ympev.2018.10.033.

[102] Bruce Rannala and Ziheng Yang. Bayes estimation of species divergence times and ancestral population sizes using DNA sequences from multiple loci. *Genetics*, 164(4):1645–1656, 2003.

[103] Rezwana Reaz, Md. Shamsuzzoha Bayzid, and M. Sohel Rahman. Accurate phylogenetic tree reconstruction from quartets: A heuristic approach. *PLOS ONE*, 9(8):1–13, 08 2014. doi: 10.1371/journal.pone.0104008.

[104] Benjamin D. Redelings and Marc A. Suchard. Joint bayesian estimation of alignment and phylogeny. *Systematic Biology*, 54(3):401–418, 2005. doi: 10.1080/10635150590947041.

[105] Arang Rhie, Shane A McCarthy, Olivier Fedrigo, Joana Damas, Giulio Formenti, Sergey Koren, Marcela Uliano-Silva, William Chow, Arkarachai Fungtammasan, Juwan Kim, Chul Lee, Byung June Ko, Mark Chaisson, Gregory L Gedman, Lindsey J Cantin, Francoise Thibaud-Nissen, Leanne Haggerty, Iliana Bista, Michelle Smith, Bettina Haase, Jacquelyn Mountcastle, Sylke Winkler, Sadye Paez, Jason Howard, Sonja C Vernes, Tanya M Lama, Frank Grutzner, Wesley C Warren, Christopher N Balakrishnan, Dave Burt, Julia M George, Matthew T Biegler, David Iorns, Andrew Digby, Daryl Eason, Bruce Robertson, Taylor Edwards, Mark Wilkinson, George Turner, Axel Meyer, Andreas F Kautt, Paolo Franchini, H William Detrich, 3rd, Hannes Svardal, Maximilian Wagner, Gavin J P Naylor, Martin

Pippel, Milan Malinsky, Mark Mooney, Maria Simbirsky, Brett T Hannigan, Trevor Pesout, Marlys Houck, Ann Misuraca, Sarah B Kingan, Richard Hall, Zev Kronenberg, Ivan Sović, Christopher Dunn, Zemin Ning, Alex Hastie, Joyce Lee, Siddarth Selvaraj, Richard E Green, Nicholas H Putnam, Ivo Gut, Jay Ghurye, Erik Garrison, Ying Sims, Joanna Collins, Sarah Pelan, James Torrance, Alan Tracey, Jonathan Wood, Robel E Dagnew, Dengfeng Guan, Sarah E London, David F Clayton, Claudio V Mello, Samantha R Friedrich, Peter V Lovell, Ekaterina Osipova, Farooq O Al-Ajli, Simona Secomandi, Heebal Kim, Constantina Theofanopoulou, Michael Hiller, Yang Zhou, Robert S Harris, Kateryna D Makova, Paul Medvedev, Jinna Hoffman, Patrick Masterson, Karen Clark, Fergal Martin, Kevin Howe, Paul Flicek, Brian P Walenz, Woori Kwak, Hiram Clawson, Mark Diekhans, Luis Nassar, Benedict Paten, Robert H S Kraus, Andrew J Crawford, M Thomas P Gilbert, Guojie Zhang, Byrappa Venkatesh, Robert W Murphy, Klaus-Peter Koepfli, Beth Shapiro, Warren E Johnson, Federica Di Palma, Tomas Marques-Bonet, Emma C Teeling, Tandy Warnow, Jennifer Marshall Graves, Oliver A Ryder, David Haussler, Stephen J O'Brien, Jonas Korlach, Harris A Lewin, Kerstin Howe, Eugene W Myers, Richard Durbin, Adam M Phillippy, and Erich D Jarvis. Towards complete and error-free genome assemblies of all vertebrate species. *Nature*, 592(7856):737–746, 2021.

[106] John A Rhodes, Hector Baños, Jonathan D Mitchell, and Elizabeth S Allman. MSCquartets 1.0: quartet methods for species trees and networks under the multispecies coalescent model in R. *Bioinformatics*, 37(12):1766–1768, 2020. doi: 10.1093/bioinformatics/btaa868.

[107] D.F. Robinson and L.R. Foulds. Comparison of phylogenetic trees. *Mathematical Biosciences*, 53(1):131–147, 1981. doi: https://doi.org/10.1016/0025-5564(81)90043-2.

[108] Sebastien Roch. A short proof that phylogenetic tree reconstruction by maximum likelihood is hard. *IEEE/ACM Transactions on Computation Biology and Bioinformatics*, 3(1):92, 2006. doi: 10.1109/TCBB.2006.4.

[109] Sebastien Roch and Sagi Snir. Recovering the tree-like trend of evolution despite extensive lateral genetic transfer: A probabilistic analysis. In Benny Chor, editor, *Research in Computational Molecular Biology*, pages 224–238, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg. doi: 10.1007/978-3-642-29627-7_23.

[110] Sebastien Roch and Mike Steel. Likelihood-based tree reconstruction on a concatenation of aligned sequence data sets can be statistically inconsistent. *Theoretical Population Biology*, 100:56–62, 2015. doi: 10.1016/j.tpb.2014.12.005.

[111] Sebastien Roch, Michael Nute, and Tandy Warnow. Long-branch attraction in species tree estimation: Inconsistency of partitioned likelihood and topology-based summary methods. *Systematic Biology*, 68(2):281–297, 2018. doi: 10.1093/sysbio/syy061.

[112] N. A. Rosenberg. The probability of topological concordance of gene trees and species trees. *Theoretical Population Biology*, 61(2):225–247, 2002. doi: 10.1006/tpbi.2001.1568.

[113] Naruya Saitou and Masatoshi Nei. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular biology and evolution*, 4(4):406–425, 1987.

[114] Erfan Sayyari and Siavash Mirarab. Fast coalescent-based computation of local branch support from quartet frequencies. *Molecular Biology and Evolution*, 33(7):1654–1668, 2016. doi: 10.1093/molbev/msw079.

[115] Erfan Sayyari, James B Whitfield, and Siavash Mirarab. Fragmentary Gene Sequences Negatively Impact Gene Tree and Species Tree Reconstruction. *Molecular Biology and Evolution*, 34(12):3279–3291, 2017. doi: 10.1093/molbev/msx261.

[116] Tae-Kun Seo. Calculating bootstrap probabilities of phylogeny using multilocus sequence data. *Molecular Biology and Evolution*, 25(5):960–971, 2008. doi: 10.1093/molbev/msn043.

[117] Xing-Xing Shen, Chris T. Hittinger, and Antonis Rokas. Contentious relationships in phylogenomic studies can be driven by a handful of genes. *Nature Ecology Evolution*, 1:1–10, 2017. doi: 10.1038/s41559-017-0126.

[118] Mark P. Simmons and John Gatesy. Coalescence vs. concatenation: Sophisticated analyses vs. first principles applied to rooting the angiosperms. *Molecular Phylogenetics and Evolution*, 91:98–122, 2015. doi: https://doi.org/10.1016/j.ympev.2015.05.011.

[119] Mark P. Simmons and John Gatesy. Collapsing dubiously resolved gene-tree branches in phylogenomic coalescent analyses. *Molecular Phylogenetics and Evolution*, 158:107092, 2021. doi: https://doi.org/10.1016/j.ympev.2021.107092.

[120] Mark P. Simmons, Daniel B. Sloan, and John Gatesy. The effects of subsampling gene trees on coalescent methods applied to ancient divergences. *Molecular Phylogenetics and Evolution*, 97:76–89, 2016. doi: 10.1016/j.ympev.2015.12.013.

[121] Mark P. Simmons, Mark S. Springer, and John Gatesy. Gene-tree misrooting drives conflicts in phylogenomic coalescent analyses of palaeognath birds. *Molecular Phylogenetics and Evolution*, 167:107344, 2022. doi: 10.1016/j.ympev.2021.107344.

[122] Chris Simon. An evolving view of phylogenetic support. *Systematic Biology*, 71(4):921–928, 2020. doi: 10.1093/sysbio/syaa068.

[123] Megan L. Smith, Dan Vanderpool, and Matthew W. Hahn. Using all gene families vastly expands data available for phylogenomic inference. *Molecular Biology and Evolution*, 39 (6):msac112, 06 2022. doi: 10.1093/molbev/msac112.

[124] Sagi Snir and Satish Rao. Quartets MaxCut: A divide and conquer quartets algorithm. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 7(4):704–718, October 2010. doi: 10.1109/TCBB.2008.133.

[125] Sagi Snir and Satish Rao. Quartet MaxCut: A fast algorithm for amalgamating quartet trees. *Molecular Phylogenetics and Evolution*, 62(1):1–8, 2012. ISSN 1055-7903. doi: 10.1016/j.ympev.2011.06.021.

[126] Claudia Solís-Lemus and Cécile Ané. Inferring phylogenetic networks with maximum pseudolikelihood under incomplete lineage sorting. *PLOS Genetics*, 12(3):1–21, 2016. doi: 10.1371/journal.pgen.1005896.

[127] Claudia Solís-Lemus, Paul Bastide, and Cécile Ané. Phylonetworks: A package for phylogenetic networks. *Molecular Biology and Evolution*, 34(12):3292–3298, 2017. doi: 10.1093/molbev/msx235.

[128] S. Song, L. Liu, S. V. Edwards, and S. Wu. Resolving conflict in eutherian mammal phylogeny using phylogenomics and the multispecies coalescent model. *Proceedings of the National Academy of Sciences of the United States of America*, 109:14942–14947, 2012.

[129] Mark S. Springer and John Gatesy. The gene tree delusion. *Molecular Phylogenetics and Evolution*, 94:1–33, 2016. ISSN 1055-7903. doi: 10.1016/j.ympev.2015.07.018.

[130] Mark S. Springer and John Gatesy. A new phylogeny for aves is compromised by pervasive misalignment and homology problems. *Proceedings of the National Academy of Sciences USA*, 121(29):e2406494121, 2024. doi: 10.1073/pnas.2406494121.

[131] Mark S. Springer, Erin K. Molloy, Daniel B. Sloan, Mark P. Simmons, and John Gatesy. ILS-Aware analysis of low-homoplasy retroelement insertions: Inference of species trees and introgression Using quartets. *Journal of Heredity*, 111(2):147–168, 2020. doi: 10.1093/jhered/esz076.

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

[133] Alexandros Stamatakis. RAxML version 8: a tool for phylogenetic analysis and post-analysis of large phylogenies. *Bioinformatics*, 30(9):1312–1313, 2014. ISSN 1367-4803. doi: 10.1093/bioinformatics/btu033.

[134] Jacob L. Steenwyk, Yuanning Li, Xiaofan Zhou, Xing-Xing Shen, and Antonis Rokas. Incongruence in the phylogenomics era. *Nature Reviews Genetics*, 24(12):834–850, 2023. doi: 10.1038/s41576-023-00620-x.

[135] Jacob L. Steenwyk, Gemma I. Martínez-Redondo, Thomas J. Buida III, Emile Gluck-Thaler, Xing-Xing Shen, Toni Gabaldón, Antonis Rokas, and Rosa Fernández. PhyKIT: A Multitool for Phylogenomics. *Current Protocols*, 4(11):e70016, 2024. doi: 10.1002/cpz1.70016.

[136] Peter F. Stevens. Angiosperm Phylogeny Website, version 14, 2017. Accessed `https://www.mobot.org/mobot/research/APweb/` and `https://d3amtssd1tejdt.cloudfront.net/2019/2320/6/APP-E-2019-V3-PPA.pdf` on December 11, 2024.

[137] Josefin Stiller, Shaohong Feng, Al-Aabid Chowdhury, Iker Rivas-González, David A Duchêne, Qi Fang, Yuan Deng, Alexey Kozlov, Alexandros Stamatakis, Santiago Claramunt, Jacqueline M T Nguyen, Simon Y W Ho, Brant C Faircloth, Julia Haag, Peter Houde, Joel Cracraft, Metin Balaban, Uyen Mai, Guangji Chen, Rongsheng Gao, Chengran Zhou, Yulong Xie, Zijian Huang, Zhen Cao, Zhi Yan, Huw A Ogilvie, Luay Nakhleh, Bent Lindow, Benoit Morel, Jon Fjeldså, Peter A Hosner, Rute R da Fonseca, Bent Petersen, Joseph A Tobias, Tamás Székely, Jonathan David Kennedy, Andrew Hart Reeve, Andras Liker, Martin Stervander, Agostinho Antunes, Dieter Thomas Tietze, Mads F Bertelsen, Fumin Lei, Carsten Rahbek, Gary R Graves, Mikkel H Schierup, Tandy Warnow, Edward L Braun, M Thomas P Gilbert, Erich D Jarvis, Siavash Mirarab, and Guojie Zhang. Complexity of avian evolution revealed by family-level genomes. *Nature*, 629(8013):851–860, 2024.

[138] Simon Tavaré. Some probabilistic and statistical problems in the analysis of DNA sequences. *Lectures on Mathematics in the Life Sciences*, 17(2):57–86, 1986.

[139] Úlfur Árnason, Fritjof Lammers, Vikas Kumar, Maria A. Nilsson, and Axel Janke. Whole-genome sequencing of the blue whale and other rorquals finds signatures for introgressive gene flow. *Science Advances*, 4(4):eaap9873, 2018. doi: 10.1126/sciadv.aap9873.

[140] Pranjal Vachaspati and Tandy Warnow. ASTRID: Accurate species trees from internode distances. *BMC Genomics*, 16(Suppl 10):S3, 2015. doi: 10.1186/1471-2164-16-S10-S3.

[141] T. Warnow. *Computational Phylogenetics: An Introduction to Designing Methods for Phylogeny Estimation*. Cambridge University Press, Cambridge, United Kingdom, 2017.

[142] Andrew M Waterhouse, James B Procter, David M A Martin, Michèle Clamp, and Geoffrey J Barton. Jalview version 2–a multiple sequence alignment editor and analysis workbench. *Bioinformatics*, 25(9):1189–1191, 2009.

[143] N. J. Wickett, S. Mirarab, et al. Phylotranscriptomic analysis of the origin and early diversification of land plants. *Proceedings of the National Academy of Sciences of the United States of America*, 111(45):E4859–E4868, 2014. doi: 10.1073/pnas.1323926111.

[144] F. Wilcoxon. *Some Rapid Approximate Statistical Procedures*, page 6. American Cyanamid Company, New York, 1949.

[145] Shaoyuan Wu, Frank E. Rheindt, Jin Zhang, Jiajia Wang, Lei Zhang, Cheng Quan, Zhiheng Li, Min Wang, Feixiang Wu, Yanhua Qu, Scott V. Edwards, Zhonghe Zhou, and Liang Liu. Genomes, fossils, and the concurrent rise of modern birds and flowering plants in the late cretaceous. *Proceedings of the National Academy of Sciences*, 121(8):e2319696121, 2024. doi: 10.1073/pnas.2319696121.

[146] Shaoyuan Wu, Frank E. Rheindt, Jin Zhang, Jiajia Wang, Lei Zhang, Cheng Quan, Zhiheng Li, Min Wang, Feixiang Wu, Yanhua Qu, Scott V. Edwards, Zhonghe Zhou, and Liang Liu. Reply to springer and gatesy: The impact of long branches and misalignments on phylogenetic analysis is minimal. *Proceedings of the National Academy of Sciences*, 121 (29):e2409344121, 2024. doi: 10.1073/pnas.2409344121.

[147] Zhenxiang Xi, Liang Liu, and Charles C. Davis. Genes with minimal phylogenetic information are problematic for coalescent analyses when gene tree estimation is biased. *Molecular Phylogenetics and Evolution*, 92:63–71, 2015. doi: 10.1016/j.ympev.2015.06.009.

[148] Zhi Yan, Megan L Smith, Peng Du, Matthew W Hahn, and Luay Nakhleh. Species tree inference methods intended to deal with incomplete lineage sorting are robust to the presence of paralogs. *Systematic Biology*, 71(2):367–381, 07 2021. doi: 10.1093/sysbio/syab056.

[149] Z Yang. Maximum-likelihood estimation of phylogeny from DNA sequences when substitution rates differ over sites. *Molecular Biology and Evolution*, 10(6):1396–1401, 1993. doi: 10.1093/oxfordjournals.molbev.a040082.

[150] DongAhn Yoo, Arang Rhie, Prajna Hebbar, Francesca Antonacci, Glennis A. Logsdon, Steven J. Solar, Dmitry Antipov, Brandon D. Pickett, Yana Safonova, Francesco Montinaro, Yanting Luo, Joanna Malukiewicz, Jessica M. Storer, Jiadong Lin, Abigail N. Sequeira, Riley J. Mangan, Glenn Hickey, Graciela Monfort Anez, Parithi Balachandran, Anton Bankevich, Christine R. Beck, Arjun Biddanda, Matthew Borchers, Gerard G. Bouffard, Emry Brannan, Shelise Y. Brooks, Lucia Carbone, Laura Carrel, Agnes P. Chan, Juyun Crawford, Mark Diekhans, Eric Engelbrecht, Cedric Feschotte, Giulio Formenti, Gage H. Garcia, Luciana de Gennaro, David Gilbert, Richard E. Green, Andrea Guarracino, Ishaan Gupta, Diana Haddad, Junmin Han, Robert S. Harris, Gabrielle A. Hartley, William T. Harvey, Michael Hiller, Kendra Hoekzema, Marlys L. Houck, Hyeonsoo Jeong, Kaivan Kamali,

Manolis Kellis, Bryce Kille, Chul Lee, Youngho Lee, William Lees, Alexandra P. Lewis, Qiuhui Li, Mark Loftus, Yong Hwee Eddie Loh, Hailey Loucks, Jian Ma, Yafei Mao, Juan F. I. Martinez, Patrick Masterson, Rajiv C. McCoy, Barbara McGrath, Sean McKinney, Britta S. Meyer, Karen H. Miga, Saswat K. Mohanty, Katherine M. Munson, Karol Pal, Matt Pennell, Pavel A. Pevzner, David Porubsky, Tamara Potapova, Francisca R. Ringeling, Joana L. Rocha, Oliver A. Ryder, Samuel Sacco, Swati Saha, Takayo Sasaki, Michael C. Schatz, Nicholas J. Schork, Cole Shanks, Linnéa Smeds, Dongmin R. Son, Cynthia Steiner, Alexander P. Sweeten, Michael G. Tassia, Françoise Thibaud-Nissen, Edmundo Torres-González, Mihir Trivedi, Wenjie Wei, Julie Wertz, Muyu Yang, Panpan Zhang, Shilong Zhang, Yang Zhang, Zhenmiao Zhang, Sarah A. Zhao, Yixin Zhu, Erich D. Jarvis, Jennifer L. Gerton, Iker Rivas-González, Benedict Paten, Zachary A. Szpiech, Christian D. Huber, Tobias L. Lenz, Miriam K. Konkel, Soojin V. Yi, Stefan Canzar, Corey T. Watson, Peter H. Sudmant, Erin Molloy, Erik Garrison, Craig B. Lowe, Mario Ventura, Rachel J. O'Neill, Sergey Koren, Kateryna D. Makova, Adam M. Phillippy, and Evan E. Eichler. Complete sequencing of ape genomes. *bioRxiv*, 2024. doi: 10.1101/2024.07.31.605654.

[151] Yun Yu and Luay Nakhleh. A maximum pseudo-likelihood approach for phylogenetic networks. *BMC Genomics*, 16 Suppl 10(S10):S10, 2015. doi: 10.1186/1471-2164-16-S10-S10.

[152] Yun Yu, James H. Degnan, and Luay Nakhleh. The probability of a gene tree topology within a phylogenetic network with applications to hybridization detection. *PLOS Genetics*, 8(4):1–10, 04 2012. doi: 10.1371/journal.pgen.1002660.

[153] Yun Yu, Jianrong Dong, Kevin J. Liu, and Luay Nakhleh. Maximum likelihood inference of reticulate evolutionary histories. *Proceedings of the National Academy of Sciences*, 111 (46):16448–16453, 2014. doi: 10.1073/pnas.1407950111.

[154] Li Yujian and Xu Liye. Unweighted multiple group method with arithmetic mean. In *2010 IEEE Fifth International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA)*, pages 830–834. IEEE, 2010.

[155] G. Udny Yule. A mathematical theory of evolution, based on the conclusions of Dr. J. C. Willis, F.R.S. *Philosophical Transactions of the Royal Society of London. Series B, Containing Papers of a Biological Character*, 213:21–87, 1925.

[156] Chao Zhang and Siavash Mirarab. Weighting by gene tree uncertainty improves accuracy of quartet-based species trees. *Molecular Biology and Evolution*, 39(12):msac215, 2022. doi: 10.1093/molbev/msac215.

[157] Chao Zhang, Maryam Rabiee, Erfan Sayyari, and Siavash Mirarab. ASTRAL-III: Polynomial time species tree reconstruction from partially resolved gene trees. *BMC Bioinformatics*, 19(6):153, 2018. doi: 10.1186/s12859-018-2129-y.

[158] Chao Zhang, Celine Scornavacca, Erin K Molloy, and Siavash Mirarab. ASTRAL-Pro: Quartet-based species-tree inference despite paralogy. *Molecular Biology and Evolution*, 37(11):3292–3307, 2020.

[159] Chao Zhang, Rasmus Nielsen, and Siavash Mirarab. Aster: A package for large-scale phylogenomic reconstructions. *Molecular Biology and Evolution*, page msaf172, 2025. doi: 10.1093/molbev/msaf172.

[160] Chao Zhang, Rasmus Nielsen, and Siavash Mirarab. Caster: Direct species tree inference from whole-genome alignments. *Science*, 387(6737):eadk9688, 2025. doi: 10.1126/science.adk9688.