Barton, T., Gross, E., Long, C., & Rusinko, J. (2025). statistical learning with phylogenetic network invariants. *Bulletin of the Society of Systematic Biologists*.
https://doi.org/10.18061/bssb.5822

Investigations

Statistical Learning With Phylogenetic Network Invariants

Travis Barton[1], Elizabeth Gross[2], Colby Long[3], Joseph Rusinko[4]

[1] Meta, [2] University of Hawai'i at Mānoa, [3] The College of Wooster, [4] Hobart and William Smith Colleges

Keywords: coalescent, statistical phylogenetics, phylogenetic networks, algebraic statistics

https://doi.org/10.18061/bssb.5822

Bulletin of the Society of Systematic Biologists

Abstract

Phylogenetic networks provide a means of describing the evolutionary history of sets of species believed to have undergone hybridization or gene flow during their evolution. The mutation process for a set of such species can be modeled asa Markov process on a phylogenetic network. Previous work has shown that a site-pattern probability distribution from a Jukes-Cantor phylogenetic network model must satisfy certain algebraic invariants. As a corollary, aspects of the phylogenetic network are theoretically identifiable from site-pattern frequencies. In practice, because of the probabilistic nature of sequence evolution, the phylogenetic network invariants will rarely be satisfied, even for data generated under the model. Thus, using network invariants for inferring phylogenetic networks requires some means of interpreting the residuals, or deviations from zero, when observed site-pattern frequencies are substituted into the invariants. In this work, we propose a method of utilizing invariant residuals and support vector machines to infer 4-leaf level-one phylogenetic networks, from which larger networks can be reconstructed. The support vector machine is first trained on model data to learn the patterns of residuals corresponding to different network structures to classify the network that produced the data. We demonstrate the performance of our method on simulated data from the specified model, a network model that includes the multispecies coalescent process, and primate data.

Phylogenetic networks are directed acyclic graphs that aim to describe the evolutionary relationships among a set of taxa. Less restrictive than their tree counterparts, phylogenetic networks have the flexibility to model gene flow and reticulation events such as hybridization and horizontal gene transfer. Due to this flexibility, phylogenetic networks are becoming increasingly common in phylogenetic analysis, and new tools are needed for their inference. In this work, we approach the inference problem from an algebro-geometric framework, combining tools from computational algebraic geometry and statistical learning.

Currently, there is no consensus on the best method for inferring a phylogenetic network from genetic data. Thus, having multiple approaches to compare and contrast is helpful as new tools emerge. Many of the early approaches for inferring networks adapted procedures that had been successful for tree inference, such as maximum parsimony (Jin et al., 2007; Park et al., 2010) and neighbor-joining (Bryant & Moulton, 2004), or building networks from a set of smaller inferred trees (Baroni et al., 2005; Huber et al., 2011; Nakhleh et al., 2005; Yang et al., 2014). More recently, distanced-based methods have shown some promise (Allman et al., 2022; Bordewich, Huber, et al., 2018; Bordewich, Semple, et al., 2018), as well as methods that incorporate possible effects from incomplete lineage sorting using network extensions of the multispecies coalescence model (Kubatko & Chifman, 2019; Rabier et al., 2021; Solís-Lemus & Ané, 2016; Wen et al., 2016; Yu et al., 2011; J. Zhu et al., 2018). The methods based on the network multispecies coalescent model, such as SNaQ (Solís-Lemus & Ané, 2016), have had the most recent success, with many implemented in PhyloNet (Wen et al., 2018) for general use. However, issues such as scalability (Hejase & Liu, 2016) and the identifiability of certain features (e.g., 3-cycles) remain (Baños, 2019; Solís-Lemus & Ané, 2016).

In this work, we take a model-based approach to inferring networks from observed site-pattern frequencies in aligned genomic sequences. Since our goal is to show the effectiveness of algebraic methods for network inference, we start with level-one network-based Markov models, also sometimes referred to as *displayed-tree models*. We also assume a Jukes-Cantor substitution process. The advantage of working with these relatively simple network models is that their algebraic properties are well understood (Gross et al., 2021; Gross & Long, 2018). Another advantage of these models is that the unrooted network topology is identifiable from site frequency data, and the semi-directed topology is identifiable up to 3-cycles (i.e., we can not identify

the direction of edges a 3-cycle) (Gross et al., 2021; Gross & Long, 2018).

Algebraic methods that use polynomials to distinguish between different underlying graph structures have a history in phylogenetic reconstruction (Casanellas & Fernández-Sánchez, 2011; Cavender & Felsenstein, 1987; Chifman & Kubatko, 2015; Lake, 1987; Rhodes et al., 2021), both at the genomic and genetic level. In these works, data in the form of site-pattern frequencies or gene tree distributions are substituted into distinguishing polynomials, referred to as *phylogenetic invariants*, and the resulting polynomial values, or *residuals*, are used to select graph structures depending on the distance of the residuals from zero. However, determining appropriate cut-offs for the residuals has been a challenge. Here, we show that statistical learning techniques can be used to interpret the residuals in the context of the network inference problem.

Our method for inferring quarnets, Quarnet Network Reconstruction using Support Vector Machines (QNR-SVM), has two parts: training and classifying. In the training part, we generate sequences of a fixed length according to all 24 level-one quarnets. We then compute the site-pattern frequencies and transform the resulting values into Fourier coordinates (see Section 2.1). We then evaluate the 1126 polynomials described in Section 2.2 on the transformed values. Finally, we train a support vector machine classifier on these points, which we use for classification. In simulations, the classifier performs well with approximately 88% accuracy with sequences of length one million. Reducing the length of the sequences reduces the accuracy; however, we still achieve 57% accuracy when using sequences of length one thousand.

QNR-SVM focuses solely on 4-leaf level-one networks, that is, level-one *quarnets*. It has already been shown that it is theoretically possible to construct level-one networks from their quarnets (Huber et al., 2018; Iersel & Moulton, 2014) Furthermore, more efficient puzzling techniques for reconstructing larger networks from smaller networks are quickly being developed (Huber et al., 2017; Huebler et al., 2019). Thus, the ability to accurately infer level-one quarnets provides the foundation for reconstructing level-one networks of arbitrary size.

To close the introduction, we want to underscore some of the assumptions on which our method is based. First, the model does not account for incomplete lineage sorting. This contrasts with methods such as SNaQ and HyDe, which assume data are generated under a network mulitspecies coalescent model (Blischak et al., 2018; Chifman & Kubatko, 2015). The model also assumes site independence and so does not capture linkage disequilibrium.

Still, because it is so well understood from an algebraic standpoint (Gross & Long, 2018), this model is the logical starting point for exploring the use of computational algebraic geometry for network inference. Showing the effectiveness of our method on a simplified model can help motivate further studies on the algebraic and geometric properties of more complicated network models along the lines of (Baños, 2019; Casanellas & Fernández-Sánchez, 2021; Cummings et al., 2021; Hollering & Sullivant, 2021; Martin et al., 2023).

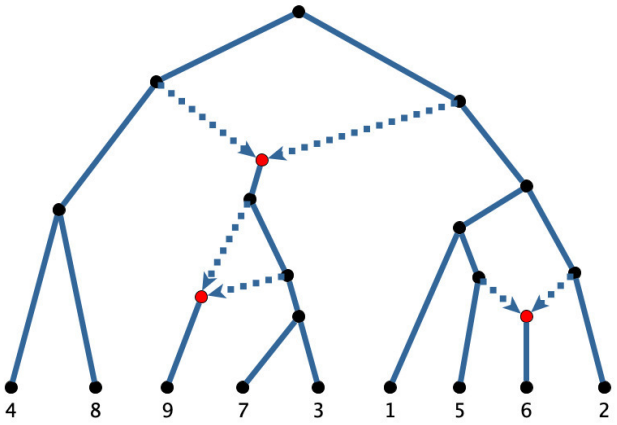


Figure 1. A level-one 9-leaf rooted binary phylogenetic network with three reticulation vertices highlighted in red and six reticulation edges distinguished by dotted lines.

1 Background

We begin this section with an introduction to the basic definitions and terminology for phylogenetic networks. We then introduce the particular phylogenetic network model that underlies our method. Finally, we give a brief overview of support vector machines, the statistical learning approach that forms the basis of our method.

1.1 Level-one semi-directed networks

Definition 1.1. A *rooted binary phylogenetic network* N on a set of leaves $[n] = \{1, \ldots, n\}$ is a rooted acyclic directed graph with no edges in parallel (i.e., no multiple edges) satisfying the following properties:

1. The root has out-degree two.
2. The only vertices with out-degree zero are the leaves, and each of these have in-degree one.
3. All other vertices either have in-degree one and out-degree two, or in-degree two and out-degree one.

The vertices of in-degree two in a phylogenetic network, such as the red vertices in Figure 1, are referred to as *reticulation vertices*, and the *level* of a phylogenetic network is the maximum number of reticulation vertices in a biconnected component of the network. A biconnected subgraph is a subgraph that remains connected (in this case, weakly connected) under the removal of any vertex, and a *biconnected component* of a graph is a maximal biconnected subgraph. Each cycle, in the undirected sense, of the network in Figure 1 is a biconnected component of the network. Since each biconnected component contains only a single reticulation vertex, the network is a level-one network. Edges directed into reticulation vertices are referred to as *reticulation edges*, while all other edges are referred to as *tree edges*. In Figure 1, the reticulation edges are marked by dotted lines.

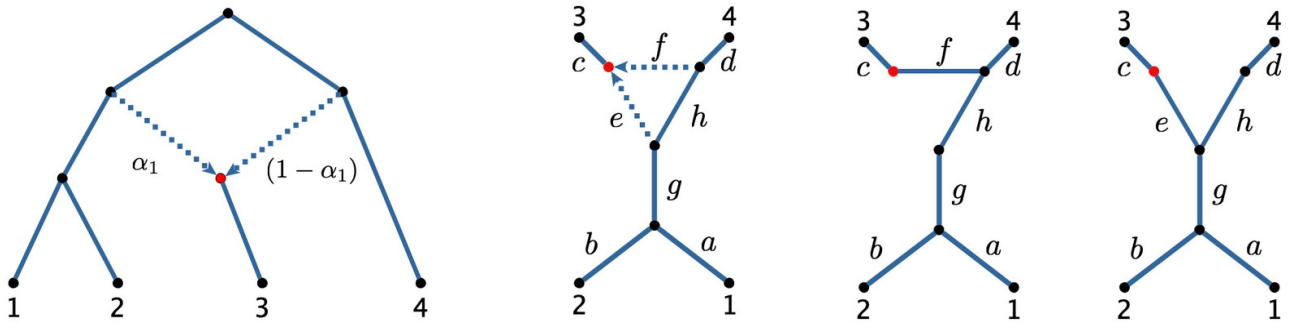


Figure 2. The four structures in the figure below include, from left to right: a level-one 4-leaf rooted binary phylogenetic network, the associated phylogenetic semi-directed network, and the two unrooted trees we obtain by deleting reticulation edges in the semi-directed network.

Tree-based networks are networks obtained by starting with a rooted binary tree and successively adding edges from tree edge to tree edge. While it is known that not all phylogenetic networks are tree-based networks (Francis & Steel, 2015), level-one networks are tree-based networks.

Given a binary phylogenetic network, if we undirect all non-reticulation edges of the network and suppress the root vertex, we obtain a *phylogenetic semi-directed network*; such graphs are called *semi-directed graphs* since some of the edges, in this case, the reticulation edges, are directed, while others are undirected. The set of phylogenetic semi-directed networks are exactly those leaf-labeled semi-directed graphs that can be obtained in this way. As an example, Figure 2 shows a 4-leaf rooted binary phylogenetic network alongside its associated phylogenetic semi-directed network. The level of a phylogenetic semi-directed network is defined just as for a phylogenetic network as are the reticulation vertices and edges. These semi-directed networks are important for our purposes since the location of the root of the phylogenetic network parameter is unidentifiable from the site-pattern probability distributions produced by the models we consider (Gross et al., 2021, sec. 2.3). Thus, all of the information about a Markov model on a binary phylogenetic network is contained in the associated phylogenetic semi-directed network.

Figure 3 represents all 4-leaf level-one binary phylogenetic semi-directed networks. Beginning in the bottom row, we refer to these as *trees*, *3-cycle networks*, *4-cycle networks*, and *double-triangle networks*. In each of the 4-cycle networks, the reticulation vertex is the large red vertex, and the reticulation edges are the two edges of the cycle directed into this vertex. For reasons of algebraic identifiability, which we discuss in Section 2.1, we do not show the reticulation edges on either the 3-cycle networks or double-triangle networks. Hence, each 3-cycle network shown represents three different phylogenetic semi-directed networks that can be obtained by specifying the reticulation vertex in the 3-cycle of the graph. Each double-triangle network represents eight phylogenetic semi-directed networks obtained by choosing the reticulation vertex in each triangle. Note that choosing the two adjacent vertices in the triangles to be reticulation vertices does not result in a phylogenetic semi-directed network since there is no root location compatible with the necessary edge orientations.

1.2 Network Models of Sequence Evolution

The method we suggest in this paper is based on the underlying network-based Markov model of sequence evolution described in (Gross & Long, 2018; Nakhleh, 2011). For a particular choice of parameters, a network-based Markov model returns a probability distribution on the n-tuples of DNA bases that may be observed at a particular site in the aligned DNA sequences of a set of n taxa. Each of these *site-pattern distributions* in the network-based model are weighted sums of site-pattern distributions belonging to tree-based phylogenetic models.

In a tree-based phylogenetic model, evolution is modeled as a k-state Markov process proceeding along a rooted n-leaf phylogenetic tree T with root ρ, where each vertex v of T is associated to a random variable X_v. In this paper, we will be concerned specifically with models of DNA sequence evolution, and so we let $k = 4$ and identify the states with the set of DNA bases $\{A, C, G, T\}$. Furthermore, since we will be concerned with *quarnets*, i.e., 4-leaf phylogenetic networks or semi-directed networks, we will also assume $n = 4$.

Let Δ^d be the d-dimensional probability simplex $\Delta^d := \{p \in \mathbb{R}^{d+1} \mid \sum_{i=1}^{d+1} p_i = 1,\ p_i \geq 0 \text{ for } 1 \leq i \leq d+1\}$. A distribution in the tree-based Markov model associated with a tree T is given by specifying a *root distribution*, a vector $\pi \in \Delta^3$ defined by $P(X_\rho = i) = \pi_i$, and a Markov transition matrix $M^{(u,v)}$ with $M^{(u,v)}_{ij} = P(X_v = j \mid X_u = i)$ to each edge (u, v) of T. The transition matrices encode the probability of mutations occurring along each edge of the tree. For phylogenetic analysis, we are particularly interested in the states at the four leaves of T. These *site-patterns* are the 4-tuples of the DNA bases that we may observe in the aligned DNA sequences for a set of species. To compute the probability of observing a particular site-pattern at the leaves, we marginalize over all possible states of the non-leaf vertices of T. In particular, let $\phi : V(T) \to \{A, C, G, T\}$ be an assignment of states to the vertices of T; we can think of ϕ as a vector of length $|V(T)|$ and use ϕ_v to denote the state of X_v. Let $\phi_{\mathcal{L}}$ be the restriction of ϕ to the leaves

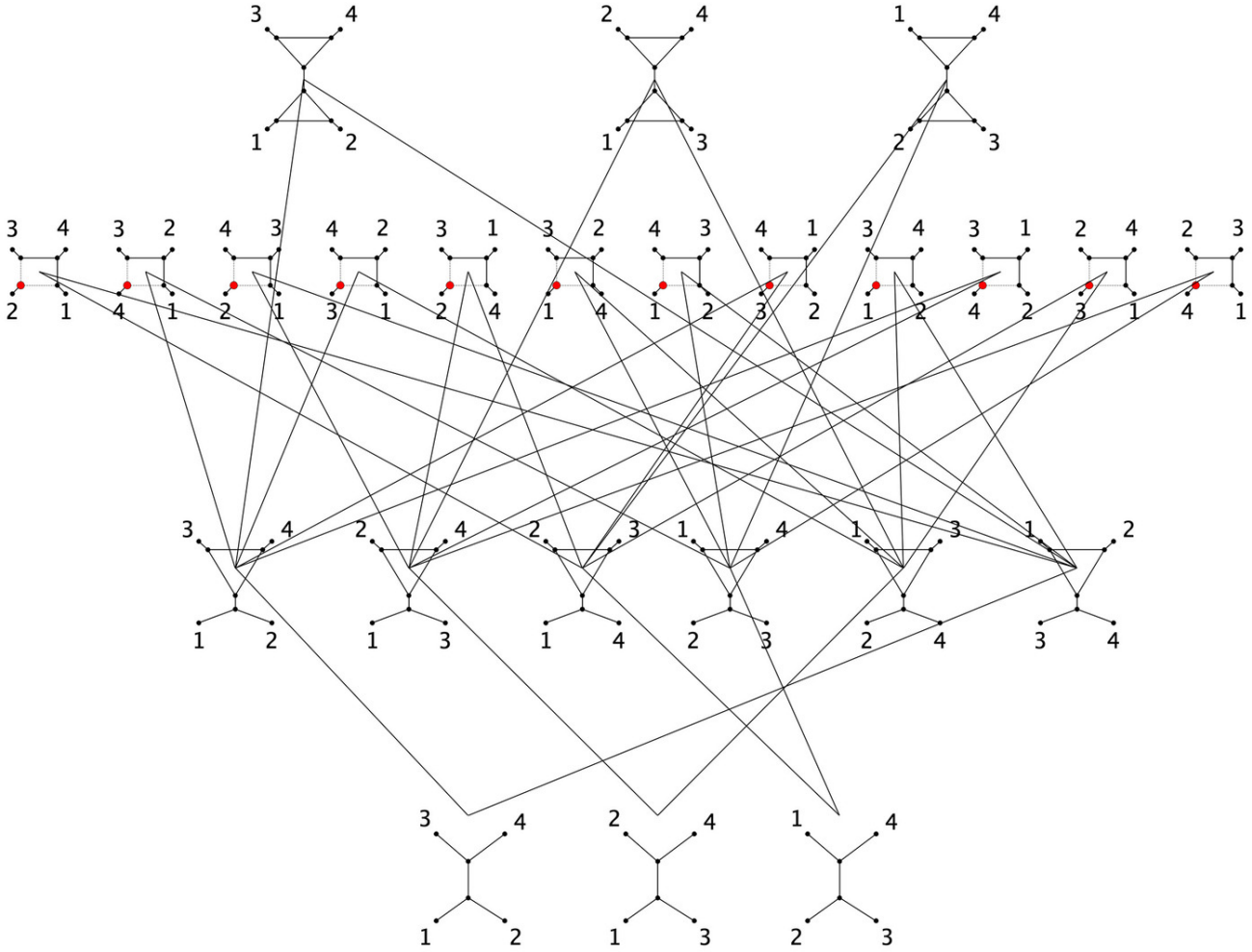


Figure 3. The poset of algebraically identifiable 4-leaf level-one phy-logenetic semi-directed networks. For the 4-cycle networks, the red dots represent reticulation vertices; for the 3-cycle networks the reticulation vertex is not identifiable. A line between two quarnets indicates that the ideal of the quarnet above is contained in the ideal of the quarnet below. For reference in the text and in the supplemental files, we number these networks from left to right and bottom to top so that the trees are numbered 1-3, the 3-cycle networks 4-9, the 4-cycle networks 10-21, and the double-triangle networks 22-24.

of T. Then the probability of observing the 4-tuple $\omega \in \{A, C, G, T\}^4$ is

$$\sum_{(\phi\,:\,\phi_{\mathcal{L}}=\omega)} \pi_{\phi_\rho} \prod_{(u,v)\in E(T)} M^{(u,v)}_{\phi_u,\phi_v}.$$

Thus, a four-leaf tree T defines a map $\psi_T : \Theta_T \to \Delta^{256-1}$ from the parameter space Θ, which includes the parameters of the root distribution and the entries of the Markov transition matrices to the set of probability distributions on the $4^4 = 256$ possible site-pattens that may be observed at the leaves of T. The *model associated to* T is defined to be $\mathcal{M}_T{:=}\mathrm{im}(\psi_T)$. A key observation from algebraic statistics that will allow us to use algebraic methods is that the map ψ_T is a polynomial map in the parameters of the model.

Similar to tree-based models, a phylogenetic network model on a 4-leaf phylogenetic level-one network N defines a polynomial map $\psi_N : \Theta_N \to \Delta^{256-1}$ from the parameter space of the network model to the set of site-pattern distributions. Again, similar to tree models, the parameter space includes a root distribution and a Markov transition matrix associated to each edge of the network. However, for a phylogenetic network model with m reticulation vertices $v_1, \ldots, v_m$, there are m additional parameters $\alpha_i \in [0, 1]$ for $1 \leq i \leq m$. In the case of level-one quarnets, the number of reticulation vertices m is at most two.

Each parameter α_i is arbitrarily associated to one of the reticulation edges e_i^0, directed into v_i. The pattern of inheritance at v_i is directed through the edge e_i^0 with probability α_i and through the other reticulation edge e_i^1 with probability $(1 - \alpha_i)$. Thus, to produce a site-pattern from the network model, for each $1 \leq i \leq m$ we independently select either e_i^0 with probability α_i, or e_i^1 with probability $(1 - \alpha_i)$, and remove the selected edge. After removing one of each pair of reticulation edges in a level-one phylogenetic network, the result is a 4-leaf phylogenetic tree with the original leaf set. A site-pattern probability distribution can be obtained from this tree-based model, as described above. Therefore, the map ψ_N can be described as a weighted sum of the maps associated to the 2^m trees obtained by deleting one of each pair of reticulation edges. The *model associated to* N is defined to be $\mathcal{M}_N{:=}\mathrm{im}(\psi_N)$. We note that network-

based Markov models are strict submodels of phylogenetic mixture models (Elizabeth S. Allman et al., 2012; Matsen et al., 2008). In the former, if two edges from the 2^m trees in the mixed distribution correspond to the same edge in $\mathcal{N}$, then their associated transition matrices are the same along each edge, whereas in the latter, the transition matrices may differ.

For tree-based and network-based Markov models in phylogenetics, it is quite common to simplify the model by adding constraints on the transition matrices. Such constraints reduce the dimension of the parameter space and the dimension of the images of ψ_T and ψ_N. In this paper, we will assume the 4–state Jukes-Cantor model of DNA sequence evolution in which the root distribution is assumed to be uniform, and each Markov transition matrix has the form

$$\begin{pmatrix} 1-3\beta & \beta & \beta & \beta \\ \beta & 1-3\beta & \beta & \beta \\ \beta & \beta & 1-3\beta & \beta \\ \beta & \beta & \beta & 1-3\beta \end{pmatrix}$$

for some $\beta \in [0, 1/4]$. The β parameter in the Markov transition matrix along a particular edge encodes the confounded effects of time and mutation rate along that edge. Thus, branch lengths are often given in terms of *expected number of substitutions per site*. These are the units used by the function `seqgen` from the `R` package `phyclust`, which we use to generate sequences. In these units, an edge of length ℓ corresponds to $\beta = \frac{1}{4}(1 - \exp(-\frac{4\ell}{3}))$.

Because the Jukes-Cantor model is time-reversible, the root in either a tree or network-based Jukes–Cantor model cannot be identified (Felsenstein, 1981; Gross et al., 2021; Gross & Long, 2018). Consequently, for rooted networks N_1 and N_2, the models $\mathcal{M}_{N_1}$ and $\mathcal{M}_{N_2}$ will be equal if N_1 and N_2 yield the same phylogenetic semi-directed network after unrooting. Therefore, given data produced by a Jukes-Cantor phylogenetic network model, it is only possible to recover the phylogenetic semi-directed network obtained by unrooting the network parameter (Gross & Long, 2018). For this reason, we define the models using phylogenetic semi-directed networks (as in Example 1.2) and work only with these structures for the rest of the paper.

Example 1.2. For the Jukes-Cantor phylogenetic model on the 4-leaf binary phylogenetic network depicted in Figure 2, the parameters of the model include the lengths of the edges and the two reticulation edge parameters, α_1 and $(1 - \alpha_1)$. As noted above, the root location of the phylogenetic network is not identifiable. Thus, we can equivalently construct the model by assigning edge lengths and reticulation edge parameters to the semi-directed network shown in the same figure. The site-pattern probability distribution from the network will then be a weighted sum of the two site-pattern probability distributions coming from the tree-based Markov model with the edge lengths shown on the two trees at the right in the figure. The weight of the distribution from the left tree will be $(1 - \alpha_1)$, and the weight from the right tree will be α_1.

1.3 Phylogenetic Network Invariants

There has been much work previously on identifying *phylogenetic invariants* for Markov models of DNA sequence evolution (Allman & Rhodes, 2007). The phylogenetic invariants for a tree-based Markov model on a tree T are the polynomials that vanish on the model $\mathcal{M}_T$. That is, letting $[n]$ be the set of leaf labels for T and letting $p_{i_1 i_2 \ldots i_n} = P(X_1 = i_1, \ldots, X_n = i_n)$, they are the polynomials contained in the ring

$$\mathbb{C}[p_{i_1 i_2 \ldots i_n} : i_1, i_2, \ldots, i_n \in \{A, C, G, T\}]$$

that evaluate to zero when the entries of any site-pattern probability distribution from the model are substituted. The set of all such polynomials that vanish on the model $\mathcal{M}_T$ is the ideal

$$I_T \subseteq \mathbb{C}[p_{i_1 i_2 \ldots i_n} : i_1, i_2, \ldots, i_n \in \{A, C, G, T\}],$$

which is called the *ideal of phylogenetic invariants* for T. One motivation for computing these ideals is that they can be used to show that the models for different trees are not contained in one another. For example, showing $I_{T_1} \nsubseteq I_{T_2}$ proves the reverse non-containment for the models, $\mathcal{M}_{T_2} \nsubseteq \mathcal{M}_{T_1}$. This observation has been used to show that the tree parameter of several tree-based Markov models is *generically identifiable*. That is, for a generic probability distribution coming from a tree-based Markov model, it is possible to recover the tree parameter(s) of the model (e.g., Allman et al., 2011; Allman & Rhodes, 2006; Rhodes & Sullivant, 2012).

In the same manner as for trees, we can consider the ideal of phylogenetic invariants for the network N, the set of all polynomials that vanish on the model $\mathcal{M}_N$. In (Gross & Long, 2018) and (Gross et al., 2021), the ideals for several small networks were computed in order to show that the network parameter of certain network-based Markov models is identifiable. This approach to establishing identifiability also suggests a method for phylogenetic inference. Specifically, suppose we have shown that for two networks N_1 and N_2 that $I_{N_1} \nsubseteq I_{N_2}$ and $I_{N_2} \nsubseteq I_{N_1}$. Then, if the network parameter is generically identifiable, given a generic site-pattern probability distribution p from either $\mathcal{M}_{N_1}$ or $\mathcal{M}_{N_2}$, the distribution p belongs to only one of these models. We can then substitute p into a set of polynomials that generate I_{N_1} and a set of polynomials that generate I_{N_2}. The result will be zero for every polynomial in the generating set of the ideal of the network with model containing p, and non-zero for at least one of the polynomials in the ideal of the network with model not containing p.

Theoretically, this same principle should allow us to infer phylogenetic networks from the observed site-pattern probability distributions coming from the aligned DNA sequences of a set of species. In practice, none of the phylogenetic invariants for any network are likely to evaluate to zero on observed data. This is not just because of the simplifying assumptions in our models. Even if we simulate from a network-based Markov model to obtain an observed site-pattern probability distribution, because the models are stochastic, we are likely to get small non-zero values when we substitute the entries of this distribution into the phylogenetic network invariants. Thus, using network in-

variants for inferring phylogenetic networks requires some means of interpreting the residuals, or deviations from zero, when observed site-pattern frequencies are substituted into the invariants.

One approach to doing this is to compute a score for each possible network by adding up the absolute value of the residuals of the invariants in a generating set for the ideal of the network. The inferred network would then be the one with the lowest score. This approach was used to infer phylogenetic trees in (Casanellas & Fernández-Sánchez, 2006; Ruskino & Hipp, 2012), though it is unlikely to be successful in our case. Those works considered only quartet trees, which all have the same unlabeled topology. This allows for a direct comparison of the computed scores since the generating sets of invariants used are related by a permutation of the variables. However, in our case, we require invariants for quarnets with four distinct unlabeled quarnet topologies (the different rows in Figure 3). It is unclear how one would compare scores for quarnets in different rows using different numbers of invariants of different degrees and with differing numbers of terms.

A more refined strategy for addressing this problem would be to apply statistical learning techniques to learn the patterns of residuals from observed phylogenetic data. However, this is not possible since there is a need for more reliable labeled data from which to learn. Yet another approach would be to derive expressions for the distribution of invariant residuals, assuming a certain phylogenetic network model of evolution. After all, for a fixed choice of model parameters, the invariants are polynomial functions of random variables. However, this would require specifying a prior distribution on the numerical parameters of the network models. Moreover, the number of variables involved and the correlation between them makes this infeasible.

For these reasons, we propose a method using support vector machines and random sampling to interpret the invariant residuals. We begin by constructing a set of invariants $\mathcal{S}$ that distinguishes between all twenty-four level-one quarnets as shown in Figure 3. We then sample from a large region of the numerical parameter space for each quarnet model. The labeled sampled data is then transformed and substituted into $\mathcal{S}$, and we train support vector machines on the invariant residuals. The support vector machines can then be used to infer quarnets for observed biological data. As previously noted, since all level-one phylogenetic networks can be constructed from their quarnets; this method can be paired with any method for constructing level-one networks from quarnets to infer level-one networks of arbitrary size.

1.4 Support Vector Machines

Our method relies on constructing a support vector machine (SVM), a supervised learning model, for classifying the invariant residuals. We provide here a brief overview of SVMs adapted from (James et al., 2013) and refer the reader there for more details.

A linear support vector machine uses separating hyperplanes for classification. In the simplest case, the training data consists of observations in a Euclidean space, each labeled as belonging to one of two classes. If the training data are separable, then there exists a hyperplane that perfectly separates the data so that the observations belonging to each class live on opposite sides of the hyperplane. In such a case, the separating hyperplane chosen is the *maximal margin classifier*, which is the separating hyperplane that maximizes the distance to the nearest point in the training data. Once the hyperplane is determined, new observations can be easily classified by determining on which side of the hyperplane they reside.

Of course, it is often not the case that the data used to train an SVM are perfectly separable. In these cases, the maximal margin classifier does not exist, and instead, we seek a *soft margin classifier*. A soft margin classifier is again a hyperplane trained on the data; however, since the data are not separable, it will necessarily misclassify some observations in the training data. The soft margin classifier is determined by choosing the hyperplane that best separates the training data according to some optimization criteria. Only now, when determining the optimal soft margin hyperplane, is a cost incurred for each misclassified observation, and the total allowable cost must remain below a chosen threshold. For example, if the allowable cost were zero, then the soft margin classifier would only exist if the data were perfectly separable, and in that case, it would be the maximal margin hyperplane. As is typical with statistical modeling, the cost parameter reflects a tradeoff between bias and variance, and the optimal cost is typically determined through cross-validation on the training set.

Two other issues distinguish most applications of SVMs from the simple case of two separable classes that we outlined above. The first issue is that even a soft margin classifier will only be effective at classifying new observations if the boundary between each pair of classes is approximately linear. However, it is easy to imagine applications where this is not the case. Consequently, SVMs are often constructed using *kernels*, which transform the original observations, possibly by embedding them in a higher dimensional space.

Non-linear decision boundaries in the original space can be represented as linear decision boundaries defined by hyperplanes in the transformed space. The second issue is that we may have observations belonging to one of several classes, rather than just two. There are a few ways to address this issue, one of which is the "one-versus-one" approach, which we use below. In this approach, given data belonging to m different classes, we construct $\binom{m}{2}$ hyperplanes, one for each pair of classes. New observations are then classified by allowing each hyperplane to "vote," and the ultimate classification is the class that receives the most votes.

In our application, the observations are vectors of invariant residuals obtained by sampling from a network-based Markov model. Each observation has a label from the set $\{1, 2, \ldots, 24\}$ based on the equivalence class of the semi-directed network that produced the observation. To build the SVM and classify points, we use the `R` package `e1071` (Meyer et al., 2019). By default, this package uses a one-versus-one approach to classify new observations. We

tune the cost parameter by trying several different values and evaluating the accuracy. Using residuals from a carefully constructed set of invariants, our data are already transformed in a way designed to separate the classes theoretically. Specifically, when comparing two quarnets, some of the coordinates correspond to distinguishing invariants for the two corresponding quarnets. We thus expect some of these coordinates to be near zero for one quarnet and nonzero for the other. Theoretically, the distinguishing invariants will be near zero for one quarnet and will lie above and below zero for the other in a way that makes a linear decision boundary a poor choice. However, this is not what we observe in practice, and typically the invariant residuals are near zero for the "correct" quarnet and bounded away from zero for the other. Thus, we have found that a linear kernel is effective on these transformed coordinates, which we use in the simulations below.

While one could apply several different supervised learning methods to the classified invariant residuals, support vector machines have a few properties that make them particularly appealing for this purpose. For one, they have an intrinsic geometric interpretation. Since we are viewing the phylogenetic network inference problem from an algebra-geometric vantage point, this may prove useful for further investigation. For example, the coefficients of the SVM hyperplanes may offer clues as to the relative importance of invariants for separating different models. Moreover, using results from (Lin et al., 2007), (Platt & others, 1999), and (Wu et al., 2003), one can modify this method to return probabilities for each class rather than a classification. This may be useful for developing methods for constructing larger networks from quarnets using a weighted quarnet scheme similar to those for trees (KS & Haeseler, 1996; Ranwez & Gascuel, 2001).

2 Constructing A Distinguishing Set of Phylogenetic Network Invariants for Jukes–Cantor Quarnets

As noted previously, our analysis will focus on inferring level-one quarnets, or 4-leaf semi-directed networks, since these can be used to build larger networks. We will also assume that the underlying DNA substitution process is the 4-state Jukes-Cantor model. In order to apply the invariants based inference method described in the previous section, we first need the following definition.

Definition 2.1. Let $\mathcal{N}$ be a set of phylogenetic networks and $\mathcal{M}$ a phylogenetic model. A set $\mathcal{S}$ is a *distinguishing set of invariants for $\mathcal{N}$ under $\mathcal{M}$*, if for all $N_1, N_2 \in \mathcal{N}$, there exists a polynomial invariant f in the vanishing ideal of I_{N_1} or I_{N_2}, but not both.

Thus, our first step is to construct a set of invariants $\mathcal{S}$ that is a distinguishing set of invariants for the set of quarnets under the Jukes-Cantor model.

2.1 Generating sets of network ideals and distinguishing invariants

Generating sets for the vanishing ideals of all level-one quarnets with a single cycle are known from (Gross & Long, 2018) and those of the three level-one quarnets with two reticulation vertices (the double-triangle networks) from (Gross et al., 2021). The computations for these ideals are contained in the supplemental materials of those works.

The ideals are computed in a set of transformed coordinates. For quarnets under the Jukes-Cantor model, the ring of transformed coordinates is the ring of *q-coordinates*, also referred to as *Fourier coordinates*,

$$\mathbb{C}[q_{i_1 i_2 i_3 i_4} : (i_1, i_2, i_3, i_4) \in \{A, C, G, T\}^4].$$

These q-coordinates are computed from the *probability coordinates* as follows. Letting χ be the matrix

$$\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}$$

with rows and columns indexed by A, C, G and T,

$$q_{i_1 i_2 i_3 i_4} = \sum \chi_{i_1, j_1} \chi_{i_2, j_2} \chi_{i_3, j_3} \chi_{i_4, j_4} p_{j_1 j_2 j_3 j_4} \tag{1}$$

where the index is over all $(j_1, j_2, j_3, j_4) \in \{A, C, G, T\}^4$. This linear change of coordinates, called the *Fourier transform*, was introduced in (Evans & Speed, 1993) and is common in phylogenetics when considering group-based models, such as the Jukes-Cantor model, because it simplifies the description of the models. For example, for group-based tree models, after applying the Fourier transform to both the domain and image spaces of the map ψ_T, the ideal I_T is generated by binomials in the q-coordinates. The details of the transform and group-based models are not particularly relevant to our analysis, so we refer the interested reader instead to (Evans & Speed, 1993; Sturmfels & Sullivant, 2005). For our purposes, the most important point is that since our invariants are expressed in the q-coordinates, we will need first to transform the observed site-pattern frequencies using (1).

Due to the symmetry of the Jukes-Cantor model, we will only need to compute part of the vector of 256 q-coordinates. This is because there are a number of site-patterns that are predicted to appear with the same frequency regardless of the network or tree parameter of the model. For example, because the substitution rate between all sites is the same under the Jukes-Cantor model, site-pattern probability distributions from every network will satisfy $p_{ACTA} = p_{GTAG}$. Put another way; this is a linear invariant in the probability coordinates that is contained in the ideal of every quarnet. This symmetry in the probability coordinates simplifies the resulting q-coordinates. For example, 128 of the q-coordinates are identically zero for every site-pattern probability distribution contained in a Jukes-Cantor model. Moreover, many of the other non-zero coordinates are identical. Working modulo these linear invariants, we can express the ideals for each of the quarnets in the ring contains only the following 15 variables,

$$\begin{aligned} \mathbb{C}[&q_{AAAA}, q_{AACC}, q_{ACAC}, q_{ACCA}, q_{ACGT}, \\ &q_{CAAC}, q_{CACA}, q_{CAGT}, q_{CCAA}, q_{CCCC}, \\ &q_{CGAT}, q_{CGCG}, q_{CGTA}, q_{CCGG}, q_{CGGC}]. \end{aligned} \tag{2}$$

The equivalence classes of these variables are listed in the catalog of Small Phylogenetic Trees (Garcia-Puente, 2007) under the Jukes-Cantor model for a 4-leaf unrooted tree. Note that the catalog assumes the additional symmetries of a particular tree, and so the equivalence classes of q_{CGCG} and q_{CGGC} as well as those of q_{CCCC} and q_{CCGG} are combined.

Computing the vanishing ideals for each level-one quarnet under the Jukes-Cantor model reveals an obstacle to computing a set of distinguishing invariants. First, as observed in (Gross & Long, 2018), there are many level-one quarnets with identical ideals. Specifically, two 3-cycle networks or two double-triangle networks with the same undirected skeleton have the same vanishing ideal under the Jukes–Cantor model. For example, consider the 4-leaf rooted binary phylogenetic network pictured in Figure 2. If we were to switch the leaf labels 3 and 4 in this network, the undirected skeleton of the associated phylogenetic semi-directed remains unchanged. The same is true if we instead switch the leaf vertex labeled by 3 and the cherry labeled by (12) in the original network.

It may be that the Markov models built on these three distinct phylogenetic semi-directed networks are in fact identical. It is also possible that they are distinguishable in some way; for example, they may satisfy certain inequalities or non-polynomial invariants. For these reasons, when we generate samples for the 3-cycles and double-triangles networks, we randomly choose one of the valid orientations for the reticulation edges. Still, since our method is based on theoretical results with polynomial invariants, we will only ever attempt to infer undirected 3-cycles. Thus, from this point forward, when we refer to quarnets, we refer to the 24 distinct topological structures depicted in Figure 3.

We also note that even among the quarnets depicted in Figure 3, the ideals of some quarnets are properly contained in the ideals of others. This containment is encoded in the figure, where a line between two quarnets indicates that the ideal of the quarnet above is contained in the ideal of the quarnet below. Thus, it is impossible to form a distinguishing set that contains an invariant that belongs to the vanishing ideal I_{N_1} but not to I_{N_2} for all quarnets N_1 and N_2.

Recalling that the containment of ideals is reverse to the containment of models, some of the above results come as no surprise. For example, it is clear that adding a reticulation edge to a tree results in a model on a 3-cycle network that must contain the tree model. This follows since any distribution from the tree model can be obtained from the 3-cycle network model by setting one of the reticulation edge parameters to 0 and the other to 1. Likewise, we would expect the models of the 3-cycle networks to be contained in the models of the double-triangle networks formed by adding an extra reticulation edge. However, it is perhaps more surprising that the models of the 3-cycle networks are contained in models for 4-cycle networks. This is not the case for the Kimura 2-parameter or Kimura 3-parameter model, so this appears to be a feature of the Jukes-Cantor model rather than a feature of the quarnets (Gross et al., 2021).

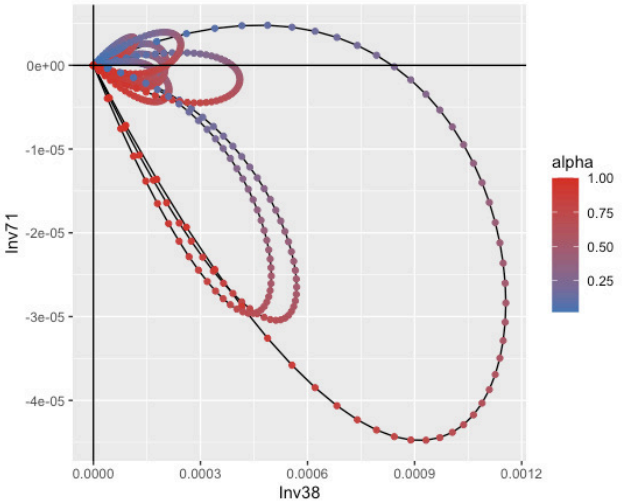


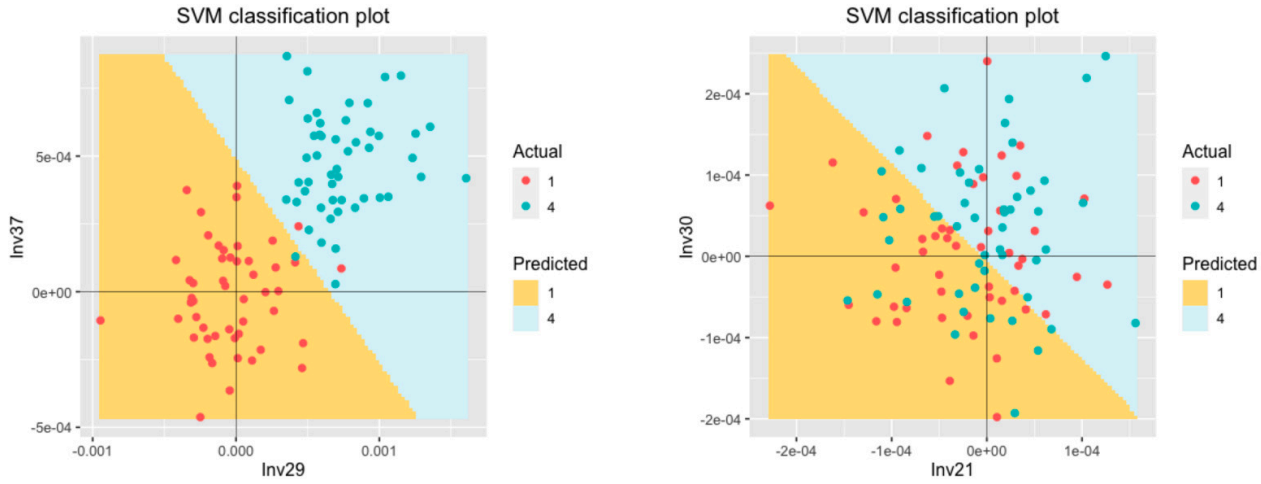
Figure 4. Plot of two invariant residuals in $I_{T_1} \setminus I_{N_4}$ as α varies for ten randomly chosen, fixed assignments of edge lengths to N_4.

This is the reason that our definition of a distinguishing set of invariants for $\mathcal{N}$ requires only that for every pair of quarnets, N_1 and N_2 in $\mathcal{N}$, the distinguishing set contains an invariant that belongs to either I_{N_1} or I_{N_2}, but not both. For example, consider the triangle quarnet N_4 and the 4-leaf tree T_1 with $I_{N_4} \subset I_{T_1}$. If $f \in I_{T_1} \setminus I_{N_4}$, then for a generic probability distribution p contained in either $\mathcal{M}_{N_4}$ or $\mathcal{M}_{T_1}$, $f(p) = 0$ if $p \in \mathcal{M}_{T_1}$ and $f(p) \neq 0$ if $p \in \mathcal{M}_{N_4}$. Indeed, this is the basis of using invariants for inference.

Example 2.2. The graph in Figure 4 shows the residuals of two invariants contained in $I_{T_1} \setminus I_{N_4}$. Each "loop" corresponds to a network obtained by randomly assigning edge lengths (between 0.1 and 0.2) to the 4-leaf semi-directed network depicted in Figure 2. Each point represents the q-coordinates of a theoretical site-pattern probability distribution in the model $\mathcal{M}_{N_4}$ evaluated at the two invariants as the reticulation edge parameter α on the edge labeled by f goes from 0 and 1. Hence, the loops start and end at the origin, since site-pattern probability distributions in the model $\mathcal{M}_{N_4}$ corresponding to reticulation edge parameters of 0 or 1 are also contained in the model $\mathcal{M}_{T_1}$.

Example 2.3. The data in Figure 5 correspond to 100 DNA sequence alignments of 10^6 sites for each taxon. Half of the sequence alignments were sampled from a site-pattern probability distribution in $\mathcal{M}_{T_1}$ and half from a site-pattern probability distribution in $\mathcal{M}_{N_4}$. We randomly selected the model parameters for sampling using the methods described in Section 3.1.

In each of the plots, the plotted points correspond to the residuals of two different invariants and the colored regions correspond to the decision boundary of the SVM trained on the plotted data. The invariant residuals shown in Figure 5 (A) are from two invariants in $\mathcal{S}$ that distinguish these quarnets (i.e., they belong to $I_{T_1} \setminus I_{N_4}$). The invariant residuals shown in Figure 5 (B) are from two invariants in $\mathcal{S}$ that do not distinguish these quarnets since they each vanish on both models (i.e., they belong to $I_{T_1} \cap I_{N_4}$).



(A) Invariants contained in $I_{T_1} \setminus I_{N_4}$. (B) Invariants contained in $I_{T_1} \cap I_{N_4}$.

Figure 5. These figures show the SVM regions using the residuals of two invariants. The training data consists of sequences sampled from site-pattern probability distributions in $\mathcal{M}_{T_1}$ and $\mathcal{M}_{N_4}$.

The theoretical basis for our method is depicted in Figure 5 (A). The distinguishing invariants are much better able to distinguish between the two quarnets when compared to the invariants contained in both ideals. We quantify this by applying these SVMs to an independently generated test set of 200 samples for T_1 and 200 samples for N_4. The accuracy of the model trained on distinguishing invariants is 92.25% compared to 51.25% for the model trained on invariants belonging to both ideals. Though not shown, we also applied the same method with two invariants in $\mathcal{S}$ that do not belong to either ideal. Even though there is no theoretical basis for these invariants to perform well, it would not be surprising if some polynomial transformations of the data had some power to distinguish between quarnets. In this case, the accuracy was 53.00%–slightly better than the two invariants contained in both ideals (as shown in Figure 5 (B)) but still less than those that theoretically distinguish the network.

Different pairs of invariants may behave differently, but to illustrate the principle, we chose pairs of invariants of low degree (so that the residuals were relatively large) that were uncorrelated (so they did not contain the same information). To do this, we converted $\mathcal{S}$ to a list and selected the first five low-degree invariants of each type from $\mathcal{S}$. For each type, we selected the pair of invariants with the lowest absolute correlation of the residuals in the training set.

2.2 Permutation Invariance

So far, we have specified only that the set $\mathcal{S}$ be a distinguishing set. However, it is an open question in phylogenetics as to which invariants are the most effective for inferring phylogenies. One way to construct a distinguishing set is to find a generating set of each quarnet ideal and then take the union of these sets of polynomial invariants. This still leaves many possible options for the distinguishing set since the generating set of an ideal is not unique. The computer algebra systems used to compute vanishing ideals return generating sets that satisfy nice mathematical properties, such as being minimal generating sets or being Gröbner bases with respect to certain term orders. These properties do not, however, have a clear biological interpretation and may not be well suited for distinguishing phylogenetic models.

A desirable property of any phylogenetic inference algorithm is that it is invariant under permutations of the data. That is, if the algorithm returns network N for input p, then it should return network $\sigma(N)$ for input $\sigma(p)$. As an example, suppose we input the set of aligned sequences for taxa $ABCD$ in that order into a phylogenetic inference algorithm and that the algorithm returns N_{10}, the 4-cycle network at the far left in Figure 3. This indicates that taxon B is a hybrid of taxa A and C. Now suppose that instead we input the aligned sequences in the order $ACDB$. Since the relationship between the taxa has not changed, we expect the algorithm to return N_{21}, indicating that the fourth taxon (B) is a hybrid of the first and second taxa (A and C in the new ordering).

This issue for invariants-based algorithms was first identified and addressed in the case of tree invariants in (Ruskino & Hipp, 2012). In order to ensure that our algorithm is also invariant under permutations of the data, we adopt a similar approach and construct $\mathcal{S}$ itself to be *permutation invariant* as we define below. Before we define this concept more formally, we need two small pieces of notation. One, for $\sigma \in S_n$ and a variable $q_{i_1 i_2 \dots i_n}$, $\sigma(q_{i_1 i_2 \dots i_n}) = q_{\sigma(i_1 i_2 \dots i_n)}$. And two, for a polynomial $f \in \mathbb{C}[q_{i_1 i_2 \dots i_n} : i_1, i_2, \dots, i_n \in \{A, C, G, T\}]$, $\sigma(f)$ is the polynomial that results from applying σ to every variable in f. As an example, let

$$f = q_{ACCA}q_{CAGT} - q_{CGTA}q_{CACA}$$

and let σ be the transposition $(13) \in S_4$. Then

$$\sigma(f) = q_{CCAA}q_{GACT} - q_{TGCA}q_{CACA}.$$

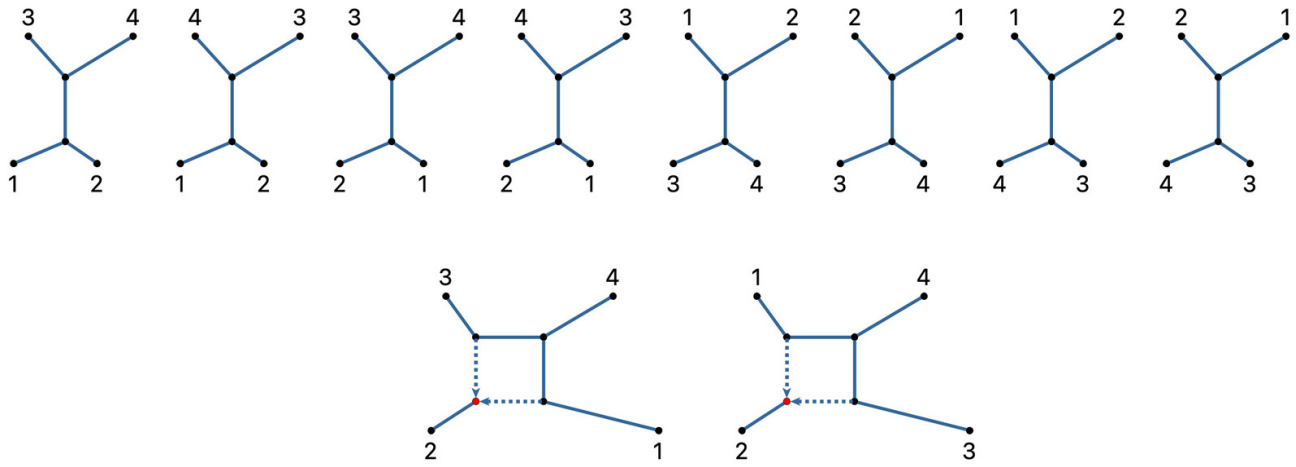


Figure 6. Applying all 24 possible label permutations to any of the trees below results in 24 trees, eight of which are topologically identical to the original. However, doing the same to either of the 4-cycles below results in 24 quarnets, only two of which are topologically identical to the original. Note that although the permuted quarnets are topologically identical, they have different branch lengths and hence correspond to different theoretical site-pattern probability distributions.

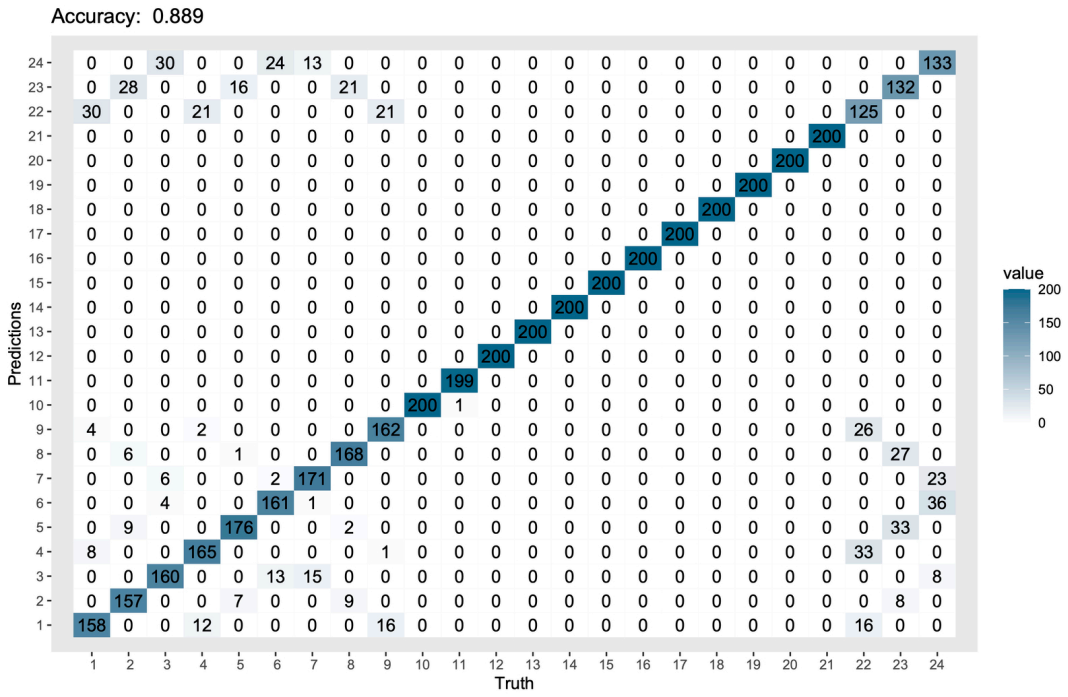


Figure 7. Confusion matrix for 4800 test observations, 200 for each quarnet. The true quarnet labels for the simulated observations are listed along the bottom of the plot, while the predicted quarnet labels are listed along the left side of the plot. A 200 in a diagonal entry means that 100% of the points sampled from the corresponding quarnet were classified correctly.

Definition 2.4. A set of polynomials

$$\mathcal{S} = \{f_1, f_2, \cdots, f_l\} \subseteq \mathbb{C}[q_{i_1 i_2 \ldots i_n} : i_1, i_2, \ldots, i_n \in \{A, C, G, T\}]$$

is *permutation invariant* if for any permutation σ in S_n and any $1 \leq j \leq l$, $\sigma(f) \in \mathcal{S}$.

In order to construct a permutation invariant set of distinguishing invariants, we first let $\mathcal{S}$ be the union of generating sets for the ideals of the quarnets T_1, N_4, N_{10}, and N_{22}. That is, we include in $\mathcal{S}$ polynomials sufficient to generate the ideal for one quarnet from each row of Figure 3. We then apply a permutation $\sigma \in S_4$ to all of the polynomials in $\mathcal{S}$, and add these polynomials to $\mathcal{S}$. We repeat this process until $\mathcal{S}$ is permutation invariant (which occurs after applying just the six transpositions in S_4). Note that we still embed our distinguishing set in the ring of q-coordinate equivalence classes from (2). For example, if we determine $\sigma(f) = q_{CCAA}q_{GACT} - q_{TGCA}q_{CACA}$

should be in $\mathcal{S}$, then we replace q_{GACT} and q_{TGCA} by their equivalence class representatives and add the polynomial $q_{CCAA}q_{CAGT} - q_{CGTA}q_{CACA}$ to $\mathcal{S}$.

The set $\mathcal{S}$ we construct contains polynomials sufficient to generate the ideal for every quarnet in Figure 3. This is because applying a permutation to a generating set for the ideal of a quarnet produces a generating set for the ideal of another quarnet in the same row. This implies that $\mathcal{S}$ is a distinguishing set. We have already shown that if N and N' are any two of the 24 quarnets, then without loss of generality, we may assume that $I_N \nsubseteq I_{N'}$. It follows then that any generating set of $I_{N'}$ contains a polynomial $f \in I_{N'} \setminus I_N$, and so $\mathcal{S}$ is a permutation invariant distinguishing set as desired. The set $\mathcal{S}$ contains 1126 polynomials and is available in the supplemental materials in a text file which can be read into Macaulay2. Each line of this text file is a separate invariant , and all specific invariants referenced herein are referenced by line number.

Constructing such a sizeable distinguishing set increases the run time of our algorithm, particularly in the construction of the support vector machines. However given the theoretical and practical implications of potentially returning different answers to an inference problem given equivalent inputs, we deemed the increase in run time from insisting $\mathcal{S}$ be permutation invariant worth the cost. Note that $\mathcal{S}$ is not the smallest permutation invariant distinguishing set we could construct. For one, we could reduce the size of $\mathcal{S}$ by beginning with a *minimal* generating set for the ideals T_1, N_4, N_{10}, and N_{22}. Moreover, it is not even strictly necessary for $\mathcal{S}$ to contain generators of every ideal in order for it to be a distinguishing set. For example, we could construct a distinguishing set that contains just a few invariants (at most $\binom{24}{2}$), and then expand this set until it is symmetric. However, previous results have shown that some invariants perform much better than others at distinguishing between phylogenetic structures from data (Casanellas et al., 2015; Casanellas & Fernández-Sánchez, 2011). A priori, it is unclear which invariants will perform best, and it may be that certain invariants perform better than others at distinguishing between quarnets over different regions of parameter space. For these reasons, we chose to include a large number of invariants in $\mathcal{S}$; which invariants are important is determined through the training process and the construction of the support vector hyperplanes.

3 Quarnet Network Reconstruction using Support Vector Machines (QNR-SVM).

We call our invariants-based algorithm for inferring phylogenetic networks Quarnet Network Reconstruction using Support Vector Machines (QNR-SVM). The algorithm takes as input the aligned DNA sequences for a set of four taxa, and then uses support vector classifiers to classify the input data as belonging to one of the 24 quarnet models. The algorithm is implemented in `R`, and the output is the quarnet associated with this model. In this section, we describe this algorithm in further detail by first describing the training data and the process used to construct the support vector classifiers.

3.1 Training the Support Vector Machine

In order to construct a point in our training data set, we begin by sampling a site-pattern probability distribution from a quarnet model. For a fixed quarnet N, a site-pattern probability distribution $p \in \mathcal{M}_N$ is determined by the numerical parameters of the model, which include the edge lengths of the quarnet, and for all quarnets that are not trees, the reticulation edge parameters.

For a fixed quarnet, we select each of the reticulation edge parameters uniformly at random from the closed interval $[0.25, 0.75]$. We select edges uniformly at random from one of two different intervals. If the edge is adjacent to a degree-2 vertex in a displayed tree of the quarnet, we use $[0.05, 0.2]$, and if it is not, then we use $[0.05, 0.4]$. We use different intervals for these edges to ensure that sampling from a network with a reticulation edge parameter set to 0 approximates sampling from a quarnet without that reticulation edge. This way the distributions for each nested chain of quarnets lie roughly in the same region of probability space. To see why this is desirable, consider the 3-cycle network pictured in Figure 2 and the displayed tree constructed by removing the reticulation edge e. The central edge of this tree is the concatenation of edges h and g, and so it could be twice as long as the maximum edge length in the interval. Thus, if we sampled these edges from the same interval we used to sample edges of the tree $12|34$, then on average, this network would reflect greater distances between taxa on opposite sides of the split $12|34$ than would the tree. Thus the low accuracy risk of the SVM classifying based on this feature of the data rather than on the topology of the underlying quarnet.

We selected this branch length range so that the displayed quartet trees have branch lengths for which invariants-based and classical gene tree estimation methods reconstruct the associated trees with very high probability (see, for example (Fernández-Sánchez & Casanellas, 2016)). Thus, we focus on distinguishing among networks rather than the well-documented challenges of phylogenetic inference in the presence of extremely short or long branches.

Once the choice of numerical parameters is made, we sample a DNA sequence alignment from the model consisting of 10^6 sites for each taxon using the function `gen.seq.HKY` (which calls the function `seqgen`) from the `R` package `phyclust` (Chen, 2011). As noted previously, a branch length ℓ in this function corresponds to a value of $\beta = \frac{1}{4}(1 - \exp(-\frac{4\ell}{3}))$ in the Jukes-Cantor transition matrix on an edge. Note that we do not scale edges to model different rates across sites, so if two sites are generated by the same displayed tree of the network, then they are generated on identical trees with the same branch lengths.

We next convert the alignment to a length 256 empirical site-pattern probability distribution. Then, we average over Jukes-Cantor equivalence classes so that entries in the same equivalence class are equal. This step is necessary for our method to be permutation invariant since we use equiv-

alence class representatives of the q-coordinates. Finally, the resulting distribution is converted to q-coordinates using Equation 1 and these 15 entries are substituted into the invariants in $\mathcal{S}$ (with a fixed ordering). The result is a vector of length 1126 labeled by the quarnet.

Instead of repeating this process for each quarnet, we fix one representative of the four unlabeled quarnet topologies. After the aligned DNA sequences are generated, we permute the sequences by each of the 24 permutations in S_4. The resulting sequences are then converted into length 1126 vectors as described and labeled with the quarnet obtained after applying the same permutation to the leaf labels. As a result, the entire set of training data is also invariant under permutation of the q-coordinate indices, a condition that ensures our algorithm is permutation invariant.

Each independent sample from each of the four unlabeled quarnet topologies results in $|S_4| = 24$ observations added to our training set. One important thing to note is the difference in the number of label permutations that fix each quarnet. For example, eight label permutations fix a quarnet tree, but only two that fix a 4-cycle (see Figure 6). Therefore, a single independent sample from a quarnet tree, once permuted, will generate eight observations labeled by each of the three quarnet trees, whereas a single sample from a 4-cycle will generate only two observations labeled by each 4-cycle.

In order to avoid a class imbalance, which can bias an SVM model towards the majority class (Batuwita & Palade, 2013), we sample so that there are 9600 observations labeled by each quarnet. While this balances the number of observations labeled by each quarnet, the number of independent samples corresponding to each quarnet will differ. For example, the observations corresponding to a 4-cycle represent four times as many independent samples as those corresponding to a tree. It is unclear exactly how this affects the construction of the support vector classifiers, but it is a necessary byproduct of our insistence on a training set with balanced classes that is invariant under permutation.

After the training data is constructed, we use the function `svm` from the `R` package `e1071` to construct our model. We use a linear kernel and a `cost` parameter of 0.1. During preliminary investigations, we tuned the `cost` parameter of the soft-margin classifier by building several models with different `cost` parameters (0.0001, 0.001, 0.01, 0.1, 1, 10) and comparing the returned accuracy of each model on an independent test set. In these tests, there was a leveling off of accuracy at the 0.1 cost level as the cost parameter decreased. However, due to computational considerations, this aspect was not explored deeply, and thus, tuning could be further explored to improve accuracy. By default, `svm` scales each column, which allows us to compare the residuals from invariants of different degrees. Also, by default, the function uses a one-versus-one approach, which we describe in Section 1.4. With $230400 = 9600 \times 24$ training points, construction of the support vector classifiers takes 21.4 hours on a iMac Pro with a 2.5 GHz Intel Xeon W processor. The resulting SVM model is available in the supplementary material.

3.2 The QNR-SVM Algorithm and Simulation Results

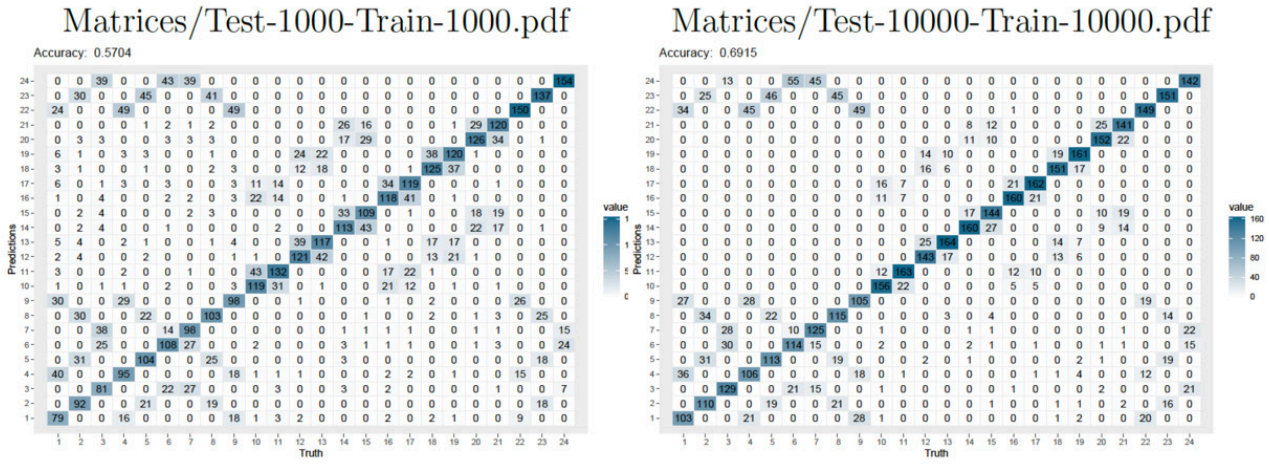
Almost all of the computational cost of the QNR-SVM algorithm comes from the generation of the training data and the construction of the support vector classifiers, which both need only be performed once. Assuming this has been done, applying the algorithm is simply a matter of transforming the input data appropriately for use in the SVM model. We summarize each step of the process here. These steps are implemented in the supplementary files (`SamplingFunctions.R` and `TrainingAndTesting.R`).

- **Step 1:** Given the aligned DNA sequences for four taxa, compute the length 256 site-pattern frequency vector p.
- **Step 2:** Replace each coordinate of p with the average count for its Jukes-Cantor equivalence class.
- **Step 3:** Use the discrete Fourier transform to convert the vector p from probability coordinates to q-coordinates.
- **Step 4:** Construct the residual vector $r = (f_1(q), \cdots, f_{1126}(q))$ for $f_i \in \mathcal{S}$.
- **Step 5:** Use the previously built SVM classifier to classify r as belonging to one of the 24 network models.

With an SVM classifier already in hand, the five steps listed above complete rather quickly. For example, for an alignment of 10^6 sites, the five steps complete in 22.21 seconds with Step 1, converting the aligned sequences into a site-pattern frequency vector, taking the most time (21.06 seconds), and Step 5, the classification step taking only 1.35 seconds. The speed at which a quarnet is recovered is one of the benefits of this method; however, as with other methods, we expect scalability to remain an issue since the number of quarnets to classify grows combinatorially with the number of leaves.

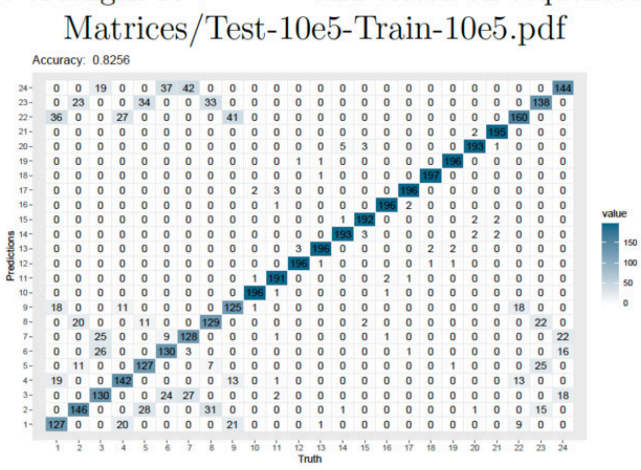
We first demonstrate the performance of the QNR-SVM algorithm on data generated according to the model on which the method is based. This at least offers evidence that the invariants-based SVM can learn general features of the network models that are not specific to the training set. Next, in Section 3.3, we apply our method to a misspecified model involving incomplete lineage sorting where we observe QNR-SVM to be robust to modest levels of gene tree discordance. Finally, in Section 4, we apply the algorithm to a biological data set consisting of primate data.

Figure 7 displays the prediction results using QNR-SVM on data generated according to the model on which it is based. The classifier was trained according to the specifications detailed in Section 3.1. The test set consists of 4800 observations from independently sampling 200 observations from each of the 24 quarnets. Each independent sample was obtained by generating an alignment of 10^6 sites for each taxon from one of the quarnet models following the same branch length sampling regime used for the training set.



(A) Confusion matrix for the classifier trained and tested on sequences of length 10^3.

(B) Confusion matrix for the classifier trained and tested on sequences of length 10^4.



(C) Confusion matrix for the classifier trained and tested on sequences of length 10^5.

Figure 8. Confusion matrices for classifiers trained on data generated from sequences of varying lengths. The true quarnet labels for the sim-ulated observations are listed along the bottom of the plot, while the predicted quarnet labels are listed along the left side of the plot. Each confusion matrix displays the results for 4800 test observations, 200 for each quarnet. The test observations are generated from sequences of the same length as the training data.

The confusion matrix in Figure 7 reveals the strengths and weaknesses of the method. The overall accuracy on our test set is 88.9%. The method performs particularly well with 4-cycles—more than 99% of the observations generated by a 4-cycle quarnet are correctly classified. Likewise, very few observations generated by one of the other quarnets are misclassified as having been generated by a 4-cycle. The method has more difficulty classifying observations sampled from trees, 3-cycles, and double-triangles. However, a closer look reveals that the misclassifications follow a consistent pattern. In almost all cases, the misclassified observations are incorrectly assigned to a submodel or supermodel of the correct model. Put another way; the observations are assigned to a quarnet that results from adding a reticulation edge to or deleting a reticulation edge from the correct quarnet. Indeed, this phenomenon explains the symmetric patterns of misclassified quarnets we see off the diagonal in the confusion matrix. This pattern is not surprising. For example, correctly classifying these observations requires in some cases, determining if an observation was sampled from a 3-cycle with a reticulation edge parameter near 0 or 1, or a tree that is essentially that same 3-cycle with reticulation edge parameter exactly 0 or 1. More precisely, we see that, in this test set, all but ten, i.e. 99.6%, of observations from trees, 3-cycles, and double-triangles are either correctly classified, or classified as a quarnet with an additional or missing reticulation.

Due to the underlying nested geometry of the models, we expected the classifier to have difficulty with triangles, and, as noted in the introduction, it is known that triangles are hard to identify in many network inference methods. In the experiment reported here, we attempted to avoid the degeneration issue that happens as reticulation edge parameters approach 0 and 1 by bounding these parameters away from 0 and 1 when generating our data for the training set and test set. As a consequence, there is some built-

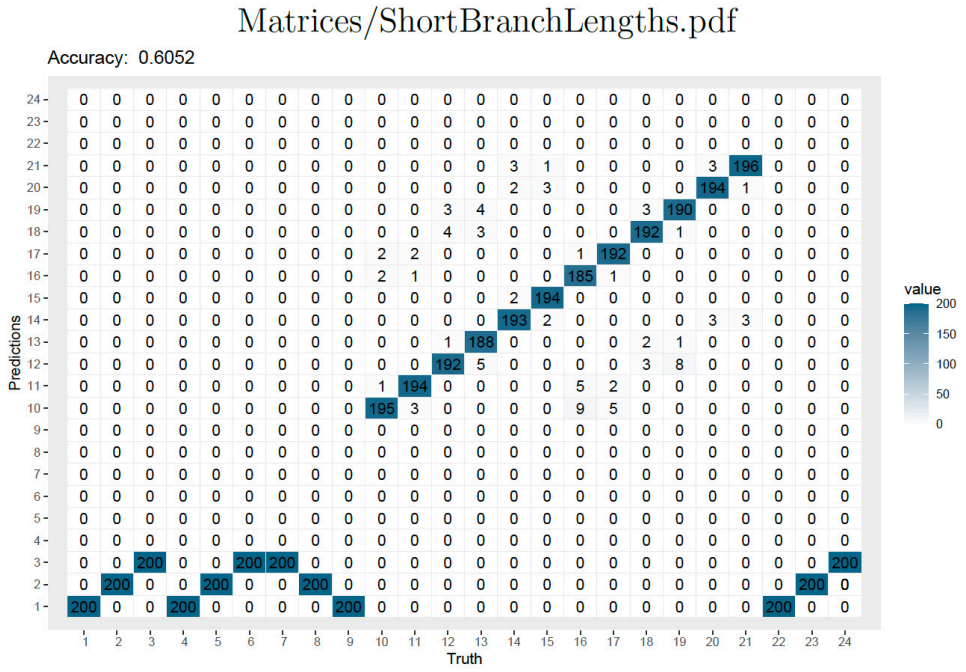


Figure 9. Confusion matrix for test observations generated from quar-nets with branch lengths selected in [0.01, 0.05] using support vector machine classifier trained on data generated from quarnets with branch lengths selected in [0.05, 0.4]

in preference for the method to default to a quarnet without a specific reticulation unless there is strong evidence for it, which is in line with the general practice of inferring the most parsimonious model in order to explain the data. However, as we see from the confusion matrix, there is room to strengthen this preference. One way this can be done is by putting a threshold on the votes or outputted probabilities. The outputted class probabilities for a SVM in the `e1071` package are obtained according to the methods in (Wu et al., 2003), where pairwise probabilities are estimated by fitting a logistic distribution to the decision values and then the class probabilities are obtained by reducing the problem to a quadratic optimization problem. For example, for the test points corresponding to the 3-cycle N_4 classified in Figure 7, the outputted probabilities when the network was classified correctly displayed a strong signal with probabilities in the $80-90\%$ range for N_4 whereas the probabilities when the network was misclassified indicated more uncertainty, generally with the probability mass concentrated on both the incorrect and correct networks (with the incorrect network having probabilities in the $50-70\%$ range).

Since we expect the variance of the residuals to shrink as the number of sites increases, the trained support vector machine is sensitive to the number of sites. For example, if we use the above-trained support vector machine on test sets constructed by generating alignments of lengths 10^3, 10^4, and 10^5, then the accuracies are 22.8%, 50.9%, and 77.5% respectively. Thus, if using shorter sequences, we suggest training the classifier according to the number of sites. We trained three additional support vector machines using $230400 = 9600 \times 24$ training points generated by alignments of length 10^3, 10^4, 10^5 and tested the classifiers using 4800 test points generated by alignments of the same length. The confusion matrices are in Figure 8, and the corresponding accuracies are 57.0%, 69.2%, and 82.5%. While the accuracy drops significantly for the classifier trained and tested on data generated by sequences of length 1000, the misclassified points follow the same pattern described above.

Finally, we aimed for our trained classifier to cover a large span of possible edge lengths, however, in order to do this, we needed to bound the training data away from zero. Thus, some care should be taken using the classifier if it is expected that the true phylogenetic network has branch lengths less than 0.05 or greater than 0.4. In one experiment summarized in Figure 9, where we constructed a test set by selecting branch lengths in the range $[0.01, 0.05]$, the accuracy decreased to 60.5%. In this case, the classifier accuracy was 100% for trees and 96% for 4-cycles, but 0% for 3-cycles and double-triangles. For the 3-cycles and double-triangles, the classifier selected the corresponding displayed tree 100% of the time. One possible way to address this sensitivity to branch lengths outside of the trained region is to train a support vector machine according to the expected branch lengths of the data set. We describe one way to do this in Section 4.2.

3.3 Simulation under the Coalescent

The QNR-SVM algorithm is based on a model that does not account for the coalescent process or other factors beyond hybridization that may result in variation of phylogenies across the genome. However, from a practical perspec-

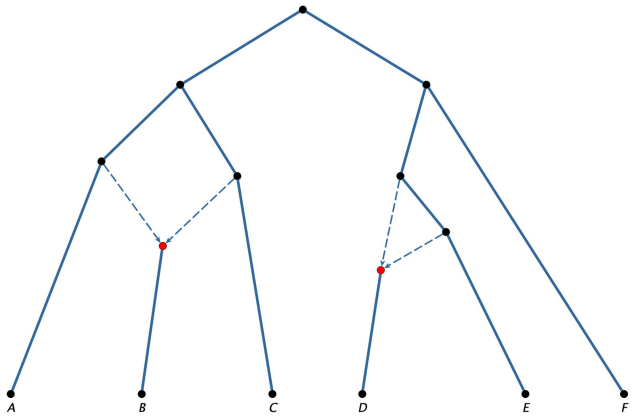


Figure 10. Network Topology for Coalescent Simulation.

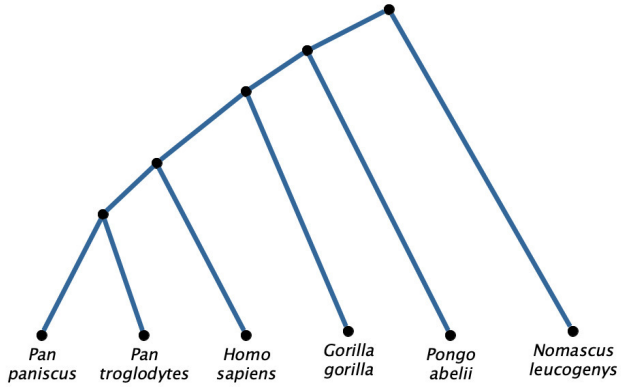


Figure 11. QNR-SVM estimated evolutionary scenario for the *Hominidae* clade. The tree has been rooted using the outgroup *Nomascus leucogenys*. All edges have 100% bootstrap support.

tive, it is easy to apply QNR-SVM to concatenated sequence data. The complexities of when and how one best combines multi-locus datasets to infer phylogenetic networks are beyond the scope of this work. However, we conducted a brief simulation to examine how QNR-SVM performs in the presence of incomplete lineage sorting under the network multispecies coalescent (NMSC) model. For details on the NMSC model, see (Degnan, 2018). While there is evidence that NMSC reconstruction may be enhanced by analyzing multiple individuals per species, for example (Cai & Ané, 2021; Rabier et al., 2021; S. Zhu & Degnan, 2017), QNR-SVM is limited to a single individual per species.

In the network multispecies coalescent model, it is frequently assumed that hybridization events happen instantaneously, and such events are modeled with reticulation edges of length zero (Blischak et al., 2018). This model choice is meant to ensure that the hybrid parent and child co-exist at the same point in time. However, it is also possible that one or both of the hybrid parents came from a lineage that went extinct or was not sampled. In these cases, it would make sense for the reticulation edge in the network to have non-zero length. Given that 4-taxa networks are frequently considered restrictions of a larger network, we believe that, in practice, most applications will involve reticulation edges of non-zero length.

With this in mind, we performed simulations using two different phylogenetic networks, one with length zero reticulation edges (N_0) and one with reticulation edges of non-zero length (N_ϵ). The 6-leaf rooted phylogenetic network topology of both networks is shown in Figure 10. This network displays all four possible unlabeled quarnet network topologies among its $\binom{6}{4} = 15$ displayed quarnets.

We assume that each edge of the species networks has the same population size. To measure the impact of incomplete lineage sorting, we test three effective population sizes ($Pop_e = 100, 1000, 10000$). The mutation rate is fixed throughout these experiments at 2.5×10^{-5} substitutions per generation.

There is no simple formula for selecting branch lengths to train a QNR-SVM model, which would be appropriate across the quarnet restrictions of a specified network under the multispecies network coalescent model. The QNR-SVM algorithm requires that we restrict to 4-taxa subsets. Thus the resulting quarnets will have branch lengths that are sums of branch lengths in the original 6-taxa network. Moreover, under the multispecies coalescent process, branches on the gene trees may be shorter or longer than corresponding branches in the species tree. To work around these difficulties, we used a range of branch lengths for the species tree such that the majority of the displayed gene trees would have branch lengths within the range of $[0.05, 0.4]$, the same range used to train our QNR-SVM model.

For each population size, we construct 100 species networks. For each simulated network, we simulate 1000 gene trees under the multispecies coalescent model with branch lengths in generations using the `PhyloCoalSimulations` package in `Julia` (Fogg et al., 2023). We simulate a sequence of length 1000 bases pairs for each gene tree under the Jukes-Cantor model using `Seq-Gen`. We concatenate these sequences such that each data point is an alignment of 10^6 sites. Associated code and detailed simulation results are available in the Supplemental Materials. As noted, the effective population size impacts the amount of incomplete lineage sorting and so the range of branch lengths in the gene trees. Table 1 summarizes how these features change under the three effective population size conditions.

We note here that large population sizes can cause two challenges in the application of QNR-SVM. As shown in Table 1, an increase in population size has the direct effect of increasing incomplete lineage sorting. As QNR-SVM does not assume a model that includes incomplete lineage sorting, we expect performance to decrease as population size increases. However, we note a secondary impact: increasing population size also increases the range of branch lengths on the gene-trees, which increases the percentage of gene tree branch lengths that fall outside of our training dataset, which, in turn, can negatively impact our estimates.

We ran the model of QNR-SVM from Section 3.1 on each 4-taxa subset. Table 2 shows the results classified by unla-

beled quarnet topology across the three effective population sizes.

When the branch length of the reticulation edge is 0, QNR-SVM does very well at detecting trees, 3-cycles, and 4-cycles, with reduced accuracy for trees and 3-cycles for the highest population size. This is not surprising, given that a higher population size means more incomplete lineage sorting. In this case, QRN-SVM often estimates the tree and three cycles as 4-cycles, with elements of the tree or hybrid cherries appearing as neighbors in the 4-cycle. The low accuracy for QNR-SVM for double-triangles is misleading. For instance, on network N_0, for effective population sizes $Pop_e = 10^2$ and $Pop_e = 10^3$, QNR-SVM always estimates the three-cycle with hybrid cherry containing B and whichever of A or C is present. With these population sizes, the hybrid cherry DE is mistaken for a tree-cherry. With the population size $Pop_e = 10^3$, QNR-SVM estimates the double 3-cycle as a single 3-cycle over 90% of the time, this time detecting each of the hybrid cherries in roughly 100 samples each. This suggests a potential benefit of using the QNR-SVM algorithm when ILS is present, especially considering that 3-cycles and double-triangle networks are not detectable from gene concordance factors (Solís-Lemus & Ané, 2016).

When the reticulation edges have non-zero branch lengths, the accuracy of QNR-SVM decreases on the trees. At the two lower population sizes QNR-SVM, all the errors in reconstructing the trees involve mistaking one of the tree cherries as a hybrid cherry. The decrease in accuracy in estimating the 3-cycles is the result of mis-estimating a tree-cherry as a hybrid cherry or vice-versa. The non-zero branch length of the reticulation edges does not impact QNR-SVM's estimation of 4-cycles and dramatically improves the estimation of double 3-cycles. When the population size is $Pop_e = 10^4$, QNR-SVM misidentifies all trees, 3-cycles, and double 3-cycles as 4-cycles but continues to place the tree and hybrid cherries as neighbors in the 4-cycle. All errant 4-cycles were still identified as 4-cycles, with most of the errors maintaining the correct ordering of the samples but misidentifying the reticulation.

Overall, this simulation shows that QNR-SVM is relatively robust to the presence of incomplete lineage sorting. However, the accuracy does decrease for very large levels of incomplete linage sorting. This simulation is not expansive enough to identify which misspecification (model or gene tree branch length distribution) is driving the reduction in accuracy. We do not recommend that QNR-SVM be used as a definitive source in constructing phylogenetic networks in the presence of incomplete lineage sorting, as the simulation provided here is small in scale. However, it does have the potential to be a helpful tool, especially when the QNR-SVM is tailored to the data set. The example in Section 4 demonstrates such an application.

4 Biological Examples

Quartets and quarnets can play an essential role in reconstructing trees and networks since the topological structure of individual quartet trees or networks can encode the topological structure of a larger tree or network from which it is sampled. For instance, SVDQuartets estimates a single quartet for each 4-taxa subset and combines these quartets into a larger species tree (Julia Chifman & Kubatko, 2014). Such a network inference process has two key components: estimating the individual quarnets and combining the quarnet information into a larger network. Currently, the QNR-SVM algorithm only addresses the first component of this problem. Still, for a small number of taxa, ad hoc approaches for assembling quarnets can provide a useful picture of how the algorithm might be applied to a data

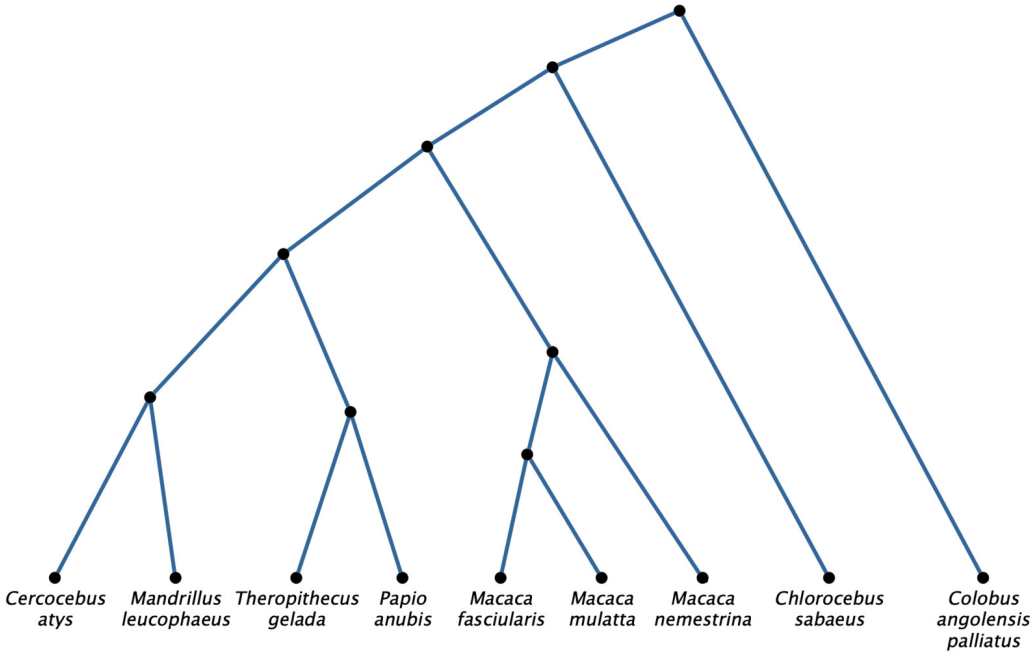


Figure 12. Network of the *Cercopithecinae* backbone tree as estimated with QNR-SVM.

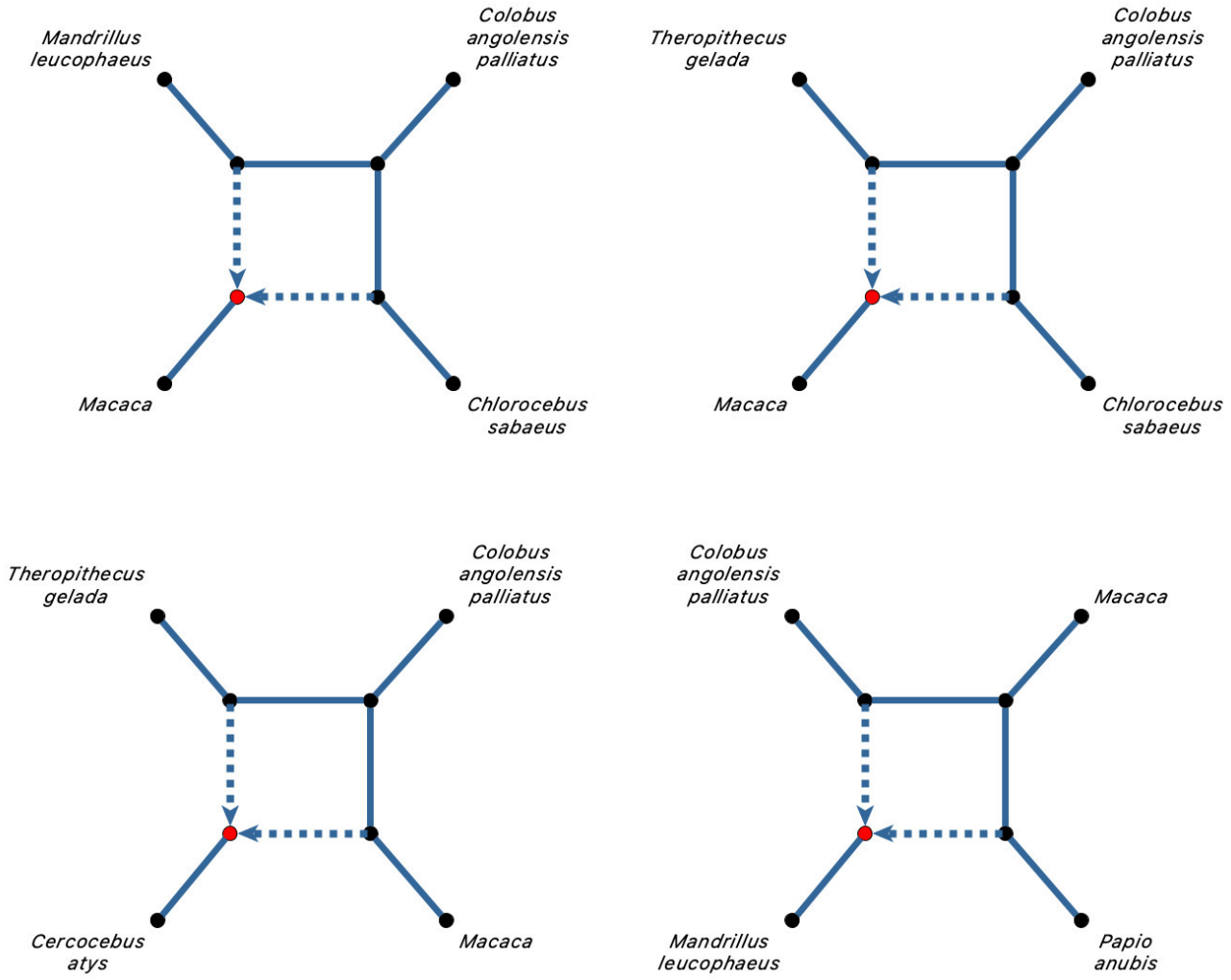


Figure 13. QNR-SVM estimated 4-cycles. Each 4-cycle represents a collection of three estimated quarnets with each of the individual macaque species placed in the leaf labeled Macaca.

set with more taxa. In the future, we envision software that combines semi-directed quarnets into a single larger network (analogous to QuartetsMaxCut for trees (Snir & Rao, 2012)) or which returns the level-one network that maximizes the number of displayed quarnets.

4.1 Description of Primate Data

In (Vanderpool et al. 2020a), Vanderpool et al. examined the phylogenomic history of 26 primate species and 3 non-primate outgroup species using various phylogenomic methods designed to reconstruct a species tree. In addition, the authors sought to detect interspecific introgression by using a version of the Δ test (Huson et al., 2005), an extension of the D-statistic test, more commonly known as the "ABBA-BABA" test (Durand et al., 2011; Green et al., 2010; Kulathinal et al., 2009), but which uses gene concordance factors as input. Using these methods, they identified six cases of introgression, primarily among the monkeys in the *Cercopithecinea* clade. It is not feasible for us to reconstruct the species network for all 29 species. Indeed, this would require us to resolve the potential conflicts among all $\binom{29}{4} = 23751$ quarnets. So instead, we apply QNR-SVM to two subsets of the primate data representing the *Hominidae* and *Cercopithecinae* clades. Here, we describe each of the examples.

Example 1: *Hominidae* clade. The first data set consists of five primates from the *Hominidae* clade; *Pan paniscus* (bonobo), *Pan troglodytes* (chimpanzee), *Pongo abelii* (orangutan), *Gorilla gorilla* (gorilla), and *Homo sapiens* (human) and the outgroup *Nomascus leucogenys* (northern white-cheeked gibbon).

Example 2: *Cercopithecinae* clade. The second example consists of the eight primates in the *Cercopithecinae* clade; *Chlorocebus sabaeus* (green monkey), *Cercocebus atys* (sooty mangabey), *Mandrillus leucophaeus* (drill), *Papio anubis* (olive baboon), *Theropithiecus gelada* (gelada baboon), *Macaca nemestrina* (southern pig-tailed macaqeu), *Macaca fascicularis*(crab-eating macaqeu), and *Macaca mulatta* (rhesus macaques), as well as the outgroup *Colobusangolensis palliatus* (black and white colobus).

We used DNA sequence data referenced in the supplementary materials of (Vanderpool et al. 2020a) and accessible on Dryad (Vanderpool et al. 2020b). This data contains the concatenated coding sequences of 1730 single-copy or-

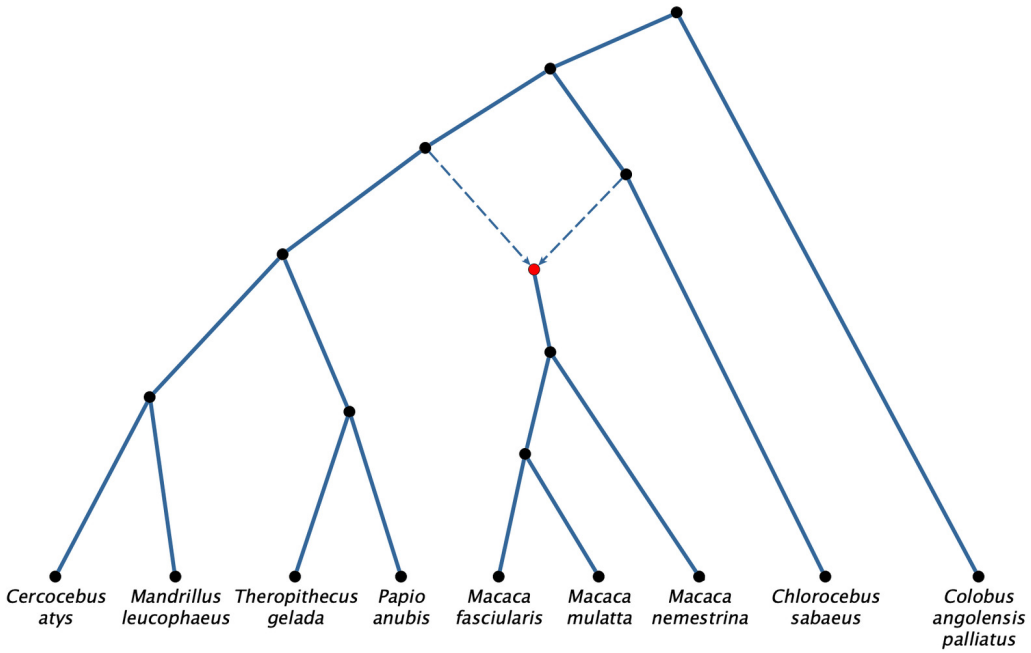


Figure 14. Best approximation of a level-one network which accounts for estimated 4-cycles.

Table 1. Impact of effective population size on gene tree topologies and branch lengths. The top two rows of the table show the observed incomplete lineage sorting (ILS) for networks N_0 and N_ϵ under each effective population size. We measure ILS as the percent of gene trees that do not match the species tree when both are restricted to $\{A, C, D, F\}$ and $\{A, C, E, F\}$ (the two topologies for which the restricted species network is a tree). The bottom two rows show the percent of quarnet branch lengths that fall within $[0.05, 0.40]$, the range of branch lengths used to train the SVM.

Pop_e	10^2	10^3	10^4
Observed ILS N_0	0.00%	4.78 %	50.2 %
Observed ILS N_ϵ	0.00%	0.35%	51.69%
Pct. of QBL in Training Range N_0	99.0%	94.7%	67.9%
Pct. of QBL in Training Range N_ϵ	99.3%	96.6%	77.8%

thologs present in at least 27 of the 29 species. This results in a sequence length of 1761114 bp for each species. For each 4-taxa subset, we examined only gap-free sites.

4.2 Methods

Vanderpool et al. found evidence of incomplete lineage sorting and hybridization in the primate data (Vanderpool et al. 2020a). As noted in Section 3.3, incomplete lineage sorting can impact the accuracy of QNR-SVM through both gene tree discordance and by shifting the gene tree branch lengths away from those found on the species tree. To help mitigate this impact, we train a new SVM on a targeted training set. To get branch lengths for our targeted training set, we first constructed a neighbor joining tree on all 29 species with distances estimated using the Jukes–Cantor model. From this tree, we then determined the lengths of the branches in each of the $\binom{29}{4}$ quartets. We then built our targeted SVM using 4000 samples from each of the 24 quarnets, where branch lengths were sampled uniformly at random from the set of quartet branch lengths found from the neighbor joining tree. Following the methodology in Section 3.1, we then simulated DNA sequences of length $500,000$ for each quarnet, and then trained a new SVM on this data.

This newly trained model was used to estimate networks for all 4-taxa subsets of the Hominidae and Cercopithecinae species. For each 4-taxa subset, we computed 100 bootstrap estimates by re-sampling sites from the gap-free alignment for those four species.

4.3 Results

The complete listing of QNR-SVM estimates and associated bootstrap support can be found in the `GitHub` repository listed in the Supplemental Materials.

4.3.1 Example 1: *Hominidae* clade

The QNR-SVM algorithm returns a tree structure with 100% bootstrap support for all fifteen subsets of the six species. The tree in Figure 11 displays all estimated quartet trees. This is consistent with the findings of (Vanderpool

Table 2. Accuracy of quarnet reconstruction for different effective population sizes. Here, accuracy refers to how often QNR-SVM correctly classified each type of quarnet in the 6-taxa network in Figure 10. For each population size and network, the averages are across 200 trees, 800 3-cycles, 300 4-cycles, and 200 double-triangles, reflecting the distribution of displayed quarnets in the species network.

Pop_e	10^2	10^3	10^4
Tree N_0	98.5%	91.5%	16.0%
Tree N_ϵ	59%	24.5%	0.0%
3-cycle N_0	99.5%	99.6%	25.9%
3-cycle N_ϵ	75%	80.0%	0.0%
4-cycle N_0	100%	100 %	100%
4-cycle N_ϵ	100%	100%	54.3 %
double-triangle N_0	0.0%	0.5 %	0.0%
double-triangle N_ϵ	97.0%	96.5%	0%

et al. 2020a) where the same tree structure received 100% bootstrap support and a posterior probability of 1.0. It is interesting to note that QNR-SVM returned the same findings despite the relatively low gene and site concordance factors in this clade found in (Vanderpool et al. 2020a).

4.3.2 Example 2: *Cercopithecinae* clade

Among the 126 subsets of 4 taxa in the *Cercopithecinae* clade example, the QNR-SVM algorithm estimated 103 underlying quarnet tree structures with 100% bootstrap support, 9 tree structures with less than 100% bootstrap support, three 4-cycle networks with 100% bootstrap support and twelve 4-cycle networks with less than 100% bootstrap support. The estimated networks are not consistent with a single level-one network. Although we lack a formal algorithm for combining these incompatible semi-directed level-one quarnets, we can still discuss the key observations supported by the QNR-SVM estimated quarnets.

The QNR-SVM algorithm strongly supports the tree in Figure 12 as a backbone on which to examine potential hybridization events. Of the 112 trees estimated by QNR-SVM, 111 are compatible with this backbone tree, with the sole exception having a bootstrap value of only 29 out of 100. This is the only tree that displays this collection of quartet trees. The backbone tree matches the underlying tree structure estimated by Vanderpool et al. (Vanderpool et al., 2020) and by Kong et al. using PhyNest (Kong et al., 2022).

Fourteen of the estimated quarnets were 4-cycles. There are 4 groups of three quarnets, in which each group places the three macaques in the same relative position to 3 other species as shown in Figure 13. Given the fixed location of the macaques in all of the estimates, it seems likely that these twelve estimates indicate at most four hybridization events involving the ancestor of the macaques rather than individual events. One of the two remaining 4-cycles has low bootstrap support (43%), in which more of the bootstrap estimates supported a tree topology. Another 4-cycle contains a single macaque, but the subsets containing the two other macaque and the remaining three samples were estimated as trees. Any representation of this 4-cycle would suggest that many other 4-cycles should have been detected but instead were estimated as trees.

The quarnets in Figure 14 are incompatible with a semi-directed level-one network. The backbone tree in Figure 12 is the level-one network which maximizes the number of displayed quarnets that match the QNR-SVM estimates. The semi-directed level-one network in Figure 14 displays fewer overall observed quarnets but captures a significant portion of the estimated 4-cycles.

Since level-one networks are a restrictive class of networks, the evolutionary events that gave rise to the macaques were likely more complex than can be described with a level-one network. In particular, we note that for this study, if we assume the semi-directed level-one network in Figure 14, the estimated quarnets that would be considered as errors do not match the error patterns that we found in the simulation studies. While our findings, as well as those of (Kong et al., 2022; Vanderpool et al., 2020), all indicate ancestral hybridization, the specific resolution of this evolutionary history remains unclear. Finally, we did not detect the within-macaque hybridization identified in (Kong et al., 2022; Vanderpool et al., 2020). We note that the branch lengths associated with quarnets containing all three macaques would be on the very short end of the branch lengths in our training set, which samples across the entire primate tree.

4 Conclusion and Discussion

In this manuscript, we explore how algebraic phylogenetic invariants can be paired with support vector machines to infer phylogenetic networks. As we see in Section 3.2, the results on simulated data are promising, as the QNR-SVM algorithm achieves overall accuracy above 88% on the test set. Moreover, almost all of the errors fit a predictable pattern in which the predicted network differs from the true network by the insertion or deletion of a reticulation edge in a triangle (or 3-cycle). While this is a novel and promising approach, it is only a first step in developing an efficient and effective tool for inference. Indeed, many questions can be explored that are likely to lead to improvements.

One of these questions regards the set of phylogenetic invariants that we used to transform our data. As discussed in Section 2.1, one of our main criteria for our method was permutation invariance, so we constructed a permutation invariant set of phylogenetic invariants. However, this set contains 1126 polynomials, meaning our training data is in $\mathbb{R}^{1126}$. The same method developed with a smaller set of invariants may be as effective and offer additional savings in the time required to train the model and classify observations.

One way to cull the set we used here would be to use some measure of variable importance and retain only those variables with demonstrated power to distinguish between quarnets. Of course, it would still be desirable for the remaining set of phylogenetic invariants to be permutation invariant. It may also be possible to use some algebraic or geometric principles to determine theoretically which invariants should perform best at distinguishing between certain quarnets, and then to permute this set to obtain a permutation invariant set of phylogenetic invariants. Doing so would likely not only improve this method but could also provide a blueprint for utilizing invariants and algebraic statistics more effectively in model selection.

A second question is how to make the method more robust with respect to the sequence length and branch lengths. In the main training and test data in our simulation study in Section 3.2, each sequence in each alignment consists of 10^6 sites. While we attain some promising results, we also saw a decrease in accuracy as we decreased the number of sites, presumably since this increases the variance of each invariant residual. This suggests that this method may be more appropriate when there is a sufficient number of sites, for example, when the data consists of a whole genome alignment rather than data from an individual gene. A separate but related question is how a *mismatch* in the number of sites used in the training and test data affects model accuracy. For example, as shown in Section 3.2, it may be desirable to pretrain several models on alignments of different lengths (e.g., 10^4, 10^5, etc.) Then, one could choose the model trained on sequences of a similar length to the sequences one wished to classify. We have yet to explore this question in depth.

A similar issue to the number of sites is the choice of branch lengths. Our experiments suggest performance of the model decreases rapidly for networks with branch lengths outside of the range of those in the training set. One possible fix for this is to retrain the model in the probable range of the branch lengths, as we did, for example, in Section 3.3 using branch length estimates from the neighbor joining tree. Then, the model can be retrained with branch lengths chosen from intervals that will result in quarnets with the same approximate pairwise distances.

We noticed better accuracy when we worked on narrower branch length intervals, so in some cases, the computational cost of training the classifier may be worth the improved accuracy. Similar to the above suggestion, one could also pretrain several models on different ranges of branch lengths appropriate for different sets of taxa as described in 4.2. In any case, though, it becomes challenging to infer quarnets with weak phylogenetic signal as the branch lengths go to zero.

Another issue that warrants discussion is the assumption of level-one networks. By combining inferred level-one quarnets, it is possible QNR-SVM can be used to reconstruct level-one networks of arbitrary size. However, because the model only returns level-one networks, it is not possible for the model to correctly infer the underlying network from data generated by a network of level-two or greater. In order to extend this methodology to general level-k networks, algebraic studies that determine distinguishing sets of invariants for level-k networks would be needed.

Finally, our method is designed to work with data generated according to the Jukes-Cantor model of DNA sequence evolution on a level-one network. It may perform less well when the substitution model is misspecified. However, it is likely possible to adapt the method to work with more general group-based models of DNA sequence evolution. For example, several invariants have been found for the level-one quarnets for the Kimura 2-parameter and 3-parameter models (Gross et al., 2021), so it seems feasible to develop a similar method.

Of course, it may even be possible to train a classifier using a more complicated model with this same set of invariants. Although this lacks the theoretical justification we offer here, it may be that the invariants here are sufficiently general to distinguish points coming from other phylogenetic models, for example, one with a coalescent process or a general time-reversible substitution process.

Supplementary Material

The supplementary files mentioned in this paper can be found here:
https://github.com/lizgross/Inferring-Phylogenetic-Networks-with-QNR-SVM

Acknowledgment

Work on this project was supported by the Mathematical Biosciences Institute and the National Science Foundation under grant DMS-1440386 for CL. EG was supported by the National Science Foundation under grant DMS-1945584. JR was supported by the National Science Foundation under Grant No. DMS-1616186.

A special thanks to the three anonymous reviewers whose detailed and thorough feedback helped strengthen this article.

Submitted: January 08, 2025 EST. Accepted: February 11, 2025 EST. Published: October 30, 2025 EST.

References

Allman, E. S., Baños, H., & Rhodes, J. A. (2022). Identifiability of species network topologies from genomic sequences using the logdet distance. *J. Math Biol.*, *84*(5), 35. https://doi.org/10.1007/s00285-022-01734-2

Allman, E. S., Petrović, S., Rhodes, J. A., & Sullivant, S. (2011). Identifiability of 2-tree mixtures for group-based models. *IEEE/ACM Trans. Comp. Biol. Bioinformatics*, *8*(3), 710–722. https://doi.org/10.1109/TCBB.2010.79

Allman, E. S., & Rhodes, J. A. (2006). The identifiability of tree topology for phylogenetic models, including covarion and mixture models. *J. Comp. Biol.*, *13*(5), 1101–1113. https://doi.org/10.1089/cmb.2006.13.1101

Allman, E. S., & Rhodes, J. A. (2007). Phylogenetic invariants. In O. Gascuel & M. A. Steel (Eds.), *Reconstructing evolution: New mathematical and computational advances*. Oxford University Press, New York.

Allman, E. S., Rhodes, J. A., & Sullivant, S. (2012). When do phylogenetic mixture models mimic other phylogenetic models? *Syst. Biol.*, *61*(6), 1049–1059. https://doi.org/10.1093/sysbio/sys064

Baños, H. (2019). Identifying species network features from gene tree quartets under the coalescent model. *Bulletin of Mathematical Biology*, *81*(2), 494–534. https://doi.org/10.1007/s11538-018-0485-4

Baroni, M., Semple, C., & Steel, M. (2005). A framework for representing reticulate evolution. *Annals of Combinatorics*, *8*(4), 391–408. https://doi.org/10.1007/s00026-004-0228-0

Batuwita, R., & Palade, V. (2013). Class imbalance learning methods for support vector machines. In H. He & Y. Ma (Eds.), *Imbalanced learning: Foundations, algorithms, and applications*. Wiley-IEEE Press. https://doi.org/10.1002/9781118646106.ch5

Blischak, P. D., Chifman, J., Wolfe, A. D., & Kubatko, L. S. (2018). HyDe: A python package for genome-scale hybridization detection. *Systematic Biology*, *67*(5), 821–829.

Bordewich, M., Huber, K. T., Moulton, V., & Semple, C. (2018). Recovering normal networks from shortest inter-taxa distance information. *Journal of Mathematical Biology*, 1–24. https://doi.org/10.1007/s00285-018-1218-x

Bordewich, M., Semple, C., & Tokac, N. (2018). Constructing tree-child networks from distance matrices. *Algorithmica*, *80*(8), 2240–2259. https://doi.org/10.1007/s00453-017-0320-6

Bryant, D., & Moulton, V. (2004). Neighbor-net: An agglomerative method for the construction of phylogenetic networks. *Molecular Biology and Evolution*, *21*(2), 255–265. https://doi.org/10.1093/molbev/msh018

Cai, R., & Ané, C. (2021). Assessing the fit of the multi-species network coalescent to multi-locus data. *Bioinformatics*, *37*(5), 634–641. https://doi.org/10.1093/bioinformatics/btaa863

Casanellas, M., & Fernández-Sánchez, J. (2006). Performance of a new invariants method on homogeneous and nonhomogeneous quartet trees. *Molecular Biology and Evolution*, *24*(1), 288–293. https://doi.org/10.1093/molbev/msl153

Casanellas, M., & Fernández-Sánchez, J. (2011). Relevant phylogenetic invariants of evolutionary models. *Journal de Mathématiques Pures et Appliqueés*, *96*(3), 207–229. https://doi.org/10.1016/j.matpur.2010.11.002

Casanellas, M., & Fernández-Sánchez, J. (2021). Rank conditions on phylogenetic networks. *Extended Abstracts GEOMVAP 2019: Geometry, Topology, Algebra, and Applications; Women in Geometry and Topology*, 65–69. https://doi.org/10.1007/978-3-030-84800-2_11

Casanellas, M., Fernández-Sánchez, J., & Michałek, M. (2015). Low degree equations for phylogenetic group-based models. *Collectanea Mathematica*, *66*(2), 203–225. https://doi.org/10.1007/s13348-014-0120-0

Cavender, J. A., & Felsenstein, J. (1987). Invariants of phylogenies in a simple case with discrete states. *J. of Class.*, *4*, 57–71. https://doi.org/10.1007/BF01890075

Chen, W.-C. (2011). *Overlapping codon model, phylogenetic clustering, and alternative partial expectation conditional maximization algorithm* [PhD thesis]. Iowa State University.

Chifman, J., & Kubatko, L. (2014). Quartet inference from SNP data under the coalescent model. *Bioinformatics*, *30*(23), 3317–3324. https://doi.org/10.1093/bioinformatics/btu530

Chifman, J., & Kubatko, L. (2015). Identifiability of the unrooted species tree topology under the coalescent model with time specific rate variation and invariable sites. *Journal of Theoretical Biology*, *374*, 35–47. https://doi.org/10.1016/j.jtbi.2015.03.006

Cummings, J., Hollering, B., & Manon, C. (2021). Invariants for level-1 phylogenetic networks under the cavendar-farris-neyman model. *arXiv:2102.03431*. https://doi.org/10.1016/j.aam.2023.102633

Degnan, J. H. (2018). Modeling hybridization under the network multispecies coalescent. *Systematic Biology*, *67*(5), 786–799. https://doi.org/10.1093/sysbio/syy040

Durand, E. Y., Patterson, N., Reich, D., & Slatkin, M. (2011). Testing for ancient admixture between closely related populations. *Molecular Biology and Evolution*, *28*(8), 2239–2252. https://doi.org/10.1093/molbev/msr048

Evans, S. N., & Speed, T. P. (1993). Invariants of some probability models used in phylogenetic inference. *Ann. Statist.*, *21*(1), 355–377. https://doi.org/10.1214/aos/1176349030

Felsenstein, J. (1981). Evolutionary trees from DNA sequences: A maximum likelihood approach. *J. Mol. Evol.*, *17*, 368–376. https://doi.org/10.1007/BF01734359

Fernández-Sánchez, J., & Casanellas, M. (2016). Invariant versus classical quartet inference when evolution is heterogeneous across sites and lineages. *Systematic Biology*, *65*(2), 280–291. https://doi.org/10.1093/sysbio/syv086

Fogg, J., Allman, E. S., & Ané, C. (2023). PhyloCoalSimulations: A simulator for network multispecies coalescent models, including a new extension for the inheritance of gene flow. *bioRxiv*, 2023–01. https://doi.org/10.1093/sysbio/syad030

Francis, A. R., & Steel, M. (2015). Which phylogenetic networks are merely trees with additional arcs? *Systematic Biology*, *64*(5), 768–777. https://doi.org/10.1093/sysbio/syv037

Garcia-Puente, L. D. (2007). *Small phylogenetic trees*. Sam Houston State University. https://www.coloradocollege.edu/aapps/ldg/small-trees/small-trees_5.html

Green, R. E., Krause, J., Briggs, A. W., Maricic, T., Stenzel, U., Kircher, M., Patterson, N., Li, H., Zhai, W., Fritz, M. H.-Y., & others. (2010). A draft sequence of the neandertal genome. *Science*, *328*(5979), 710–722. https://doi.org/10.1126/science.1188021

Gross, E., Iersel, L. van, Janssen, R., Jones, M., Long, C., & Murakami, Y. (2021). Distinguishing level-1 phylogenetic networks on the basis of data generated by markov processes. *Journal of Mathematical Biology*, *83*(3), 1–24. https://doi.org/10.1007/s00285-021-01653-8

Gross, E., & Long, C. (2018). Distinguishing phylogenetic networks. *SIAM J. Appl. Algebra Geometry*, *2*(1), 72–93. https://doi.org/10.48550/arXiv.1706.03060

Hejase, H. A., & Liu, K. J. (2016). A scalability study of phylogenetic network inference methods using empirical datasets and simulations involving a single reticulation. *BMC Bioinformatics*, *17*(1), 422. https://doi.org/10.1186/s12859-016-1277-1

Hollering, B., & Sullivant, S. (2021). Identifiability in phylogenetics using algebraic matroids. *Journal of Symbolic Computation*, *104*, 142–158. https://doi.org/10.48550/arXiv.1909.13754

Huber, K. T., Iersel, L. van, Kelk, S., & Suchecki, R. (2011). A practical algorithm for reconstructing level-1 phylogenetic networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, *8*(3), 635–649. https://doi.org/10.1007/s11538-018-0450-2

Huber, K. T., Moulton, V., Semple, C., & Wu, T. (2018). Quarnet inference rules for level-1 networks. *Bulletin of Mathematical Biology*, *80*(8), 2137–2153. https://doi.org/10.1007/s11538-018-0450-2

Huber, K. T., Van Iersel, L., Moulton, V., Scornavacca, C., & Wu, T. (2017). Reconstructing phylogenetic level-1 networks from nondense binet and trinet sets. *Algorithmica*, *77*(1), 173–200.

Huebler, S., Morris, R., Rusinko, J., & Tao, Y. (2019). Constructing semi-directed level-1 phylogenetic networks from quarnets. *arXiv Preprint arXiv:1910.00048*. https://doi.org/10.48550/arXiv.1910.00048

Huson, D. H., Klöpper, T., Lockhart, P. J., & Steel, M. A. (2005). Reconstruction of reticulate networks from gene trees. *Annual International Conference on Research in Computational Molecular Biology*, 233–249. https://doi.org/10.1007/11415770_18

Iersel, L. V., & Moulton, V. (2014). Trinets encode tree-child and level-2 phylogenetic networks. *J. Math Biol.*, *68*(7), 1707–1729. https://doi.org/10.1007/s00285-013-0683-5

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An introduction to statistical learning with applications in R*. Springer Science+Business Media. https://doi.org/10.1007/978-1-0716-1418-1

Jin, G., Nakhleh, L., Snir, S., & Tuller, T. (2007). Efficient parsimony-based methods for phylogenetic network reconstruction. *Bioinformatics*, *23*(2), e123–e128. https://doi.org/10.1093/bioinformatics/btl313

Kong, S., Swofford, D., & Kubatko, L. (2022). Inference of phylogenetic networks from sequence data using composite likelihood. *bioRxiv*, 2022–11. https://doi.org/10.1093/sysbio/syae054

KS, S., & Haeseler, A. von. (1996). Quartet puzzling: A quartet maximum-likelihood method for reconstructing tree topologies. *Mol Biol Evolution*, *13*. https://doi.org/10.1093/oxfordjournals.molbev.a025664

Kubatko, L. S., & Chifman, J. (2019). An invariants-based method for efficient identification of hybrid species from large-scale genomic data. *BMC Evolutionary Biology*, *19*(1), 1–13. https://doi.org/10.1186/s12862-019-1439-7

Kulathinal, R. J., Stevison, L. S., & Noor, M. A. (2009). The genomics of speciation in drosophila: Diversity, divergence, and introgression estimated using low-coverage genome sequencing. *PLoS Genetics*, *5*(7), e1000550. https://doi.org/10.1371/journal.pgen.1000550

Lake, J. A. (1987). A RATE-INDEPENDENT TECHNIQUE FOR ANALYSIS OF NUCLEIC-ACID SEQUENCES - EVOLUTIONARY PARSIMONY. *Molecular Biology and Evolution*, *4*(2), 167–191. https://doi.org/10.1101/2023.09.11.557152

Lin, H.-T., Lin, C.-J., & Weng, R. C. (2007). A note on platt's probabilistic outputs for support vector machines. *Machine Learning*, *68*(3), 267–276. https://doi.org/10.1093/oxfordjournals.molbev.a040433

Martin, S., Moulton, V., & Leggett, R. M. (2023). Algebraic invariants for inferring 4-leaf semi-directed phylogenetic networks. *bioRxiv*, 2023–09.

Matsen, F. A., Mossel, E., & Steel, M. (2008). Mixed-up trees: The structure of phylogenetic mixtures. *Bulletin of Mathematical Biology*, *70*(4), 1115–1139. https://doi.org/10.1007/s11538-007-9293-y

Meyer, D., Dimitriadou, E., Hornik, K., Weingessel, A., Leisch, F., Chang, C.-C., Lin, C.-C., & Meyer, M. D. (2019). Package "e1071." *The R Journal*.

Nakhleh, L. (2011). *Problem solving handbook in computational biology and bioinformatics* (pp. 125–158). Springer Science+Business Media, LLC. https://doi.org/10.1007/978-0-387-09760-2

Nakhleh, L., Ruths, D., & Wang, L.-S. (2005). RIATA-HGT: A fast and accurate heuristic for reconstructing horizontal gene transfer. *International Computing and Combinatorics Conference*, 84–93. https://doi.org/10.1007/11533719_11

Park, H. J., Jin, G., & Nakhleh, L. (2010). Bootstrap-based support of HGT inferred by maximum parsimony. *BMC Evolutionary Biology*, *10*, 131. https://doi.org/10.1186/1471-2148-10-131

Platt, J. & others. (1999). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in Large Margin Classifiers*, *10*(3), 61–74.

Rabier, C.-E., Berry, V., Stoltz, M., Santos, J. D., Wang, W., Glaszmann, J.-C., Pardi, F., & Scornavacca, C. (2021). On the inference of complex phylogenetic networks by markov chain monte-carlo. *PLoS Computational Biology*, *17*(9), e1008380. https://doi.org/10.1371/journal.pcbi.1008380

Ranwez, V., & Gascuel, O. (2001). Quartet-based phylogenetic inference: Improvements and limits. *Molecular Biology and Evolution*, *18*, 1103–1116. https://doi.org/10.1093/oxfordjournals.molbev.a003881

Rhodes, J. A., Baños, H., Mitchell, J. D., & Allman, E. S. (2021). MSCquartets 1.0: Quartet methods for species trees and networks under the multispecies coalescent model in r. *Bioinformatics*, *37*(12), 1766–1768. https://doi.org/10.1093/bioinformatics/btaa868

Rhodes, J. A., & Sullivant, S. (2012). Identifiability of large phylogenetic mixtures. *Bulletin of Mathematical Biology*, *74*(1), 212–231. https://doi.org/10.1007/s11538-011-9672-2

Ruskino, J. P., & Hipp, B. (2012). Invariant based quartet puzzling. *Algorithms Mol Biol.*, *7*(35). https://doi.org/10.1186/1748-7188-7-35

Snir, S., & Rao, S. (2012). Quartet MaxCut: A fast algorithm for amalgamating quartet trees. *Molecular Phylogenetics and Evolution*, *62*(1), 1–8. https://doi.org/10.1016/j.ympev.2011.06.021

Solís-Lemus, C., & Ané, C. (2016). Inferring phylogenetic networks with maximum pseudolikelihood under incomplete lineage sorting. *PLoS Genetics*, *12*(3), e1005896. https://doi.org/10.1371/journal.pgen.1005896

Sturmfels, B., & Sullivant, S. (2005). Toric ideals of phylogenetic invariants. *J. Comp. Biol.*, *12*(2), 204–228. https://doi.org/10.48550/arXiv.q-bio/0402015

Vanderpool, D., Minh, B. Q., Lanfear, R., Hughes, D., Murali, S., Harris, R. A., Raveendran, M., Muzny, D. M., Hibbins, M. S., Williamson, R. J., & others. (2020). *Supplementary data for: Primate phylogenomics uncovers multiple rapid radiations and ancient interspecific introgression* (No. DOE-SLC-6903-1). Dryad. https://doi.org/10.5061/dryad.rfj6q577d

Wen, D., Yu, Y., & Nakhleh, L. (2016). Bayesian inference of reticulate phylogenies under the multispecies network coalescent. *PLoS Genetics*, *12*(5), e1006006. https://doi.org/10.1371/journal.pgen.1006006

Wen, D., Yu, Y., Zhu, J., & Nakhleh, L. (2018). Inferring phylogenetic networks using PhyloNet. *Systematic Biology*, *67*(4), 735–340. https://doi.org/10.1093/sysbio/syy015

Wu, T.-F., Lin, C.-J., & Weng, R. (2003). Probability estimates for multi-class classification by pairwise coupling. *Advances in Neural Information Processing Systems*, *16*.

Yang, J., Grünewald, S., Xu, Y., & Wan, X.-F. (2014). Quartet-based methods to reconstruct phylogenetic networks. *BMC Systems Biology*, *8*(1), 21. https://doi.org/10.1093/molbev/mst040

Yu, Y., Than, C., Degnan, J. H., & Nakhleh, L. (2011). Coalescent histories on phylogenetic networks and detection of hybridization despite incomplete lineage sorting. *Syst. Biol.*, *60*(2), 138–149. https://doi.org/10.1093/sysbio/syq084

Zhu, J., Wen, D., Yu, Y., Meudt, H. M., & Nakhleh, L. (2018). Bayesian inference of phylogenetic networks from bi-allelic genetic markers. *PLoS Computational Biology*, *14*(1), e1005932. https://doi.org/10.1371/journal.pcbi.1005932

Zhu, S., & Degnan, J. H. (2017). Displayed trees do not determine distinguishability under the network multispecies coalescent. *Systematic Biology*, *66*(2), 283–298. https://doi.org/10.1093/sysbio/syy019