Bioinformatics, 37, 2021, i142–i150
doi: 10.1093/bioinformatics/btab267
ISMB/ECCB 2021

Advancing admixture graph estimation via maximum likelihood network orientation

Erin K. Molloy[1,2,*], Arun Durvasula[3] and Sriram Sankararaman[1,3,4,5,*]

[1]Department of Computer Science, University of California, Los Angeles, LA 90095, USA, [2]Institute for Advanced Computer Studies, University of Maryland, College Park, College Park, MD 20740, USA, [3]Department of Human Genetics, David Geffen School of Medicine, University of California, Los Angeles, LA 90095, USA, [4]Bioinformatics Interdepartmental Program, University of California, Los Angeles, LA 90095, USA and [5]Department of Computational Medicine, University of California, Los Angeles, LA 90095, USA

*To whom correspondence should be addressed.

Abstract

Motivation: Admixture, the interbreeding between previously distinct populations, is a pervasive force in evolution. The evolutionary history of populations in the presence of admixture can be modeled by augmenting phylogenetic trees with additional nodes that represent admixture events. While enabling a more faithful representation of evolutionary history, *admixture graphs* present formidable inferential challenges, and there is an increasing need for methods that are accurate, fully automated and computationally efficient. One key challenge arises from the size of the space of admixture graphs. Given that exhaustively evaluating all admixture graphs can be prohibitively expensive, heuristics have been developed to enable efficient search over this space. One heuristic, implemented in the popular method TreeMix, consists of adding edges to a starting tree while optimizing a suitable objective function.

Results: Here, we present a demographic model (with one admixed population incident to a leaf) where TreeMix and any other starting-tree-based maximum likelihood heuristic using its likelihood function is *guaranteed* to get stuck in a local optimum and return an incorrect network topology. To address this issue, we propose a new search strategy that we term maximum likelihood network orientation (MLNO). We augment TreeMix with an exhaustive search for an MLNO, referring to this approach as OrientAGraph. In evaluations including previously published admixture graphs, OrientAGraph outperformed TreeMix on 4/8 models (there are no differences in the other cases). Overall, OrientAGraph found graphs with higher likelihood scores and topological accuracy while remaining computationally efficient. Lastly, our study reveals several directions for improving maximum likelihood admixture graph estimation.

Availability and implementation: OrientAGraph is available on Github (https://github.com/sriramlab/OrientAGraph) under the GNU General Public License v3.0.

Contact: ekmolloy@cs.ucla.edu or sriram@cs.ucla.edu

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Admixture, the exchange of genes between distinct populations, has emerged as an important process in shaping genetic diversity within and between populations. Evidence of gene flow events in many species, including humans (Green *et al.*, 2010), wolves (Pilot *et al.*, 2019) and butterflies (Edelman *et al.*, 2019), has led to increasing interest in estimation of the demographic history of populations while accounting for admixture.

The demographic history of populations in the presence of admixture can be naturally modeled by augmenting phylogenetic trees with additional nodes that represent admixture events. These *admixture graphs* cannot be directly observed and need to be estimated from genomic data. Admixture graphs model changes in allele frequencies as a Wright–Fisher process, parametrized by the pair $(N, \boldsymbol{\Theta})$, where N is a *phylogenetic network* on a set S of populations (Definition 1) and $\boldsymbol{\Theta}$ is a real-valued vector describing parameters associated with the edges of N, namely branch lengths in units of genetic drift and admixture proportions. The objective of admixture graph inference is to estimate $(N, \boldsymbol{\Theta})$ from genetic variation data measured across the populations in S. Many popular methods utilize summary statistics, such as the vector $\boldsymbol{X}$ of f-statistics, computed from allele frequencies in each of the S populations.

One of the most widely used tools, qpGraph (Patterson *et al.*, 2012), takes $\boldsymbol{X}$ as well as a network topology N as input and then fits the numerical parameters $\boldsymbol{\Theta}$. When the number of populations is very small, it is possible to evaluate the likelihood of all network topologies with a fixed number of admixture events (e.g. using the admixturegraph R package from Leppälä *et al.*, 2017). Otherwise, the researcher must propose potential networks and evaluate them individually (see Harney *et al.*, 2021 for discussion).

Brute force and manual procedures quickly become intractable, as the number of possible network topologies grows superexponentially in the number of populations (McDiarmid *et al.*, 2015). This has motivated the development of heuristics for estimating admixture graphs from *f*-statistics. For example, MixMapper (Lipson *et al.*, 2013, 2014) is a pipeline in which the user identifies a set of unadmixed populations $R \subseteq S$, estimates a submodel (i.e. a population tree) on R, and then sequentially adds populations in $S \setminus R$ to the submodel. This approach works best when there are only a few admixed populations and can be viewed as a 'semi-automated' version of qpGraph (Lipson, 2020).

A recent approach to automating admixture graph inference, called miqograph (Yan *et al.*, 2020), is unique from prior methods in that it simultaneously estimates the network topology N and numerical parameters $\mathbf{\Theta}$, requiring the user to specify some additional parameters. miqograph is similar to admixturegraph in that it is guaranteed to find the highest scoring admixture graph with a user-specified number of admixture events; however, unlike admixturegraph, the network space evaluated by miqograph is constrained to networks in which all admixture nodes are either a leaf node or are the parent of an admixture node (see Yan *et al.*, 2020 for further discussion).

To the best of our knowledge, the only method that is fully automated and (at least in principle) allows for the greatest flexibility in exploring network topologies is TreeMix (Pickrell and Pritchard, 2012). The only topological constraint imposed by TreeMix is that N is *tree-based*, meaning that it can be drawn as a tree annotated with additional edges representing gene flow (Francis and Steel, 2015; see Definition 4). At a high level, TreeMix uses a maximum likelihood (ML) search heuristic that operates by searching for an ML population tree T on S and then adding edges sequentially provided the edge additions improve a measure of model fit (hill climbing). We refer to methods that operate in this fashion as starting-tree-based maximum likelihood (STB-ML) methods. Empirical evidence suggests that TreeMix can be biased by its starting tree, especially when many populations are admixed (Lipson *et al.*, 2013), so researchers may be wary of relying solely on TreeMix, despite its scalability relative to other approaches (Lipson, 2020).

In this paper, we consider the problem of admixture graph estimation from *f*-statistics using STB-ML methods. To explore the limitations of this class of methods, we study a simple admixture graph $(N^*, \mathbf{\Theta}^*)$ with just one admixture event incident to a leaf. Given the *f*-statistics implied by the true model $(N^*, \mathbf{\Theta}^*)$ (which is equivalent to assuming that the number of independent sites goes to infinity), any STB-ML method that succeeds in finding the ML tree [which in this case is also the Neighbor Joining (NJ) tree] as its starting tree, is guaranteed to get stuck in a local optimum and return an incorrect network topology N'. Our case study highlights that STB-ML methods can be inconsistent even under a demographic model with a single admixture event.

Examining our case study in more detail, we observe that the incorrect topology N' can be transformed into the correct topology N^* simply by changing which population is admixed and redirecting the edges of N' accordingly. This graph transformation is performed by researchers manually searching for the admixture graph topology using qpGraph (Lipson, 2020) and is also related to recent theoretical results on phylogenetic network orientation by Huber *et al.* (2019). Utilizing the terminology of Huber *et al.* (2019), we say that N' and N^* are two different *orientations* of the same undirected network (Definition 5), with N^* yielding in a higher likelihood score than N'. While Huber *et al.* (2019) look at network orientation from a graph theoretic perspective (e.g. addressing questions such as: is the orientation of an undirected network uniquely determined by the position of the root and admixed populations?), here we explore network orientation as a search strategy, leading us to propose the *maximum likelihood network orientation* (MLNO) problem.

To evaluate the utility of MLNO for improving the accuracy of STB-ML methods based on *f*-statistics, we augment TreeMix with an exhaustive search for an MLNO after every edge addition. In line with our theoretical expectations, this approach, which we refer to as OrientAGraph, recovers the correct network in our motivating case study, whereas the original TreeMix method does not. We also benchmark OrientAGraph against TreeMix and miqograph on 7 model admixture graphs, 6 of which were estimated from biological datasets in recent studies. We find that OrientAGraph improves the accuracy of the original TreeMix method and, in two scenarios, is more accurate than miqograph. The first scenario occurs when an admixed population is not incident to a leaf (as this violates the topological constraints employed by miqograph); the second scenario occurs when the dataset is large enough (e.g. 10 populations and 2 admixture events) so that miqograph is unable to solve its problem to optimality within our specified maximum running time. We conclude by discussing future directions as well as the potential utility of MLNO to STB-ML methods that take estimated gene genealogies as input (e.g. Wen *et al.*, 2018; Wu, 2020; Yu *et al.*, 2014).

2 Terminology and background

Throughout this paper, we discuss results for phylogenetic networks and the approximate likelihood function computed by TreeMix (Pickrell and Pritchard, 2012); the relevant background is provided in this section.

2.1 Phylogenetic networks

We use the term *phylogenetic network* to refer to a graphical object, which we now define.

Definition 1 (Phylogenetic network). A *phylogenetic network* N is a triplet (n, S, ϕ), where n is a graph, S is a set of labels (typically denoting species or populations) and ϕ is a bijection mapping the leaves (i.e. vertices with out-degree 0) of n to the labels in S. N is *directed*, meaning that n is a directed acyclic graph with a directed path between the root (i.e. a special vertex with in-degree 0) and all other vertices in n and without any parallel arcs or self-loops. For convenience, we typically do not make an explicit distinction between a phylogenetic network N and its graph n. Instead, we say that N is a network on S and denote its vertices as $V(N)$, edges as $E(N)$ and leaves as $L(N)$.

Henceforth, we assume that N is *binary*, meaning the root has out-degree 2, leaf vertices have in-degree 1 and all other vertices, referred to as internal vertices, have in-degree and out-degree summing to 3. This is consistent with Francis and Steel (2015), Huber *et al.* (2019), and much of the literature on phylogenetic networks. However, we use the terms admixture or gene flow instead of reticulation or hybridization. Specifically, we refer to any internal vertex with in-degree ≥ 1 as an *admixture node*. An *admixture edge* is an arc whose head (target vertex) is an admixture node; all other arcs are *tree edges*. A directed phylogenetic network with zero admixture nodes is a directed *phylogenetic tree*.

TreeMix and related methods search the space of directed phylogenetic networks using operations, such as *edge additions* and *tail moves* (Fig. 2). Such operations are *legal* if they produce a directed phylogenetic network. We refer to the set of networks that can be created by performing one legal edge addition to N as the edge addition *neighborhood* of N; we use similar terminology for other operations, such as tail moves. It is worth noting that the space of all directed phylogenetic networks is connected under tail moves (Gambette *et al.*, 2017; Janssen *et al.*, 2018).

Definition 2 (Edge addition). An *edge addition* between two edges e and f in a directed phylogenetic network N involves subdividing them with new vertices s and t, respectively, and then adding an edge, referred to as the linking arc, from s to t.

Definition 3 (Tail move). A *tail move* between two edges e and f in a directed phylogenetic network N involves relocating the tail (source vertex) of edge e so that it subdivides edge f.

Many methods require networks to have specific properties; TreeMix, in particular, searches for *tree-based* networks, a class of networks introduced by Francis and Steel (2015).

Definition 4 (Tree-based). We say that a directed phylogenetic network N is *tree-based* if it can be constructed from a phylogenetic tree T, referred to as the base tree, via a sequence of edge additions, with the linking arcs representing gene flow, that can be performed in any order.

As shown by Francis and Steel (2015), this implies the existence of labeling $\psi : E(N) \rightarrow \{0, 1\}$ with the following properties: first, any nonroot vertex has *exactly* one of its incoming edges labeled 0, second, any internal vertex has *at least* one of its outgoing edges labeled 0 and third, the root has both of its outgoing edges labeled 0 (note this last requirement is specific to TreeMix). We say that edges labeled 0 are 'part of the base tree', and edges labeled 1 are 'gene flow'. Labelings with these properties are called tree-based (note that there can be multiple tree-based labelings of N).

We conclude this section by discussing *undirected* phylogenetic networks. A phylogenetic network $N' = (n', S, \theta)$ is undirected if n' is undirected and connected, with no parallel edges or self-loops (Gambette *et al.*, 2012). Again, we assume that N' is binary, so leaves are vertices with degree 1 and all other vertices have degree 3. It is easy to transform a directed phylogenetic network N into an undirected network N' simply by ignoring edge directions and suppressing the vertex previously designated as the root (Gambette *et al.*, 2012).

Definition 5 (Network orientation). We say that a directed network N is an *orientation* of an undirected network N' if the undirected version of N, denoted $N|_u$, is isomorphic to N'.

Note that two undirected networks $N'_1 = (n_1, S, \phi_1)$ and $N'_2 = (n_2, S, \phi_2)$ are *isomorphic* if there exists a bijection $f : V(N'_1) \rightarrow V(N'_2)$ such that $(u, v) \in E(N'_1)$ if and only if $(f(u), f(v)) \in E(N'_2)$ and $\phi_1(u) = \phi_2(f(u))$ for all $u \in L(N)$.

The reverse operation of transforming an undirected network N' into a directed network N, referred to as orienting the network, is not so straightforward, because the location of the root does not uniquely determine the direction of all edges in an unrooted network (Gambette *et al.*, 2012; Huber *et al.*, 2019). Notably, Huber *et al.* (2019) recently showed that specifying the the location of the root (i.e. the edge that the root subdivides) and the admixture nodes in an undirected phylogenetic network N' results in a unique orientation of N, provided that an orientation exists (Theorem 2 in Huber *et al.*, 2019). Furthermore, given this information, this orientation can be found in linear time in the number of edges (Algorithm 1 in Huber *et al.*, 2019).

In the remainder of this paper, all phylogenetic networks are directed, unless otherwise noted, and the term 'phylogenetic' is often omitted when referring to phylogenetic trees and networks.

2.2 Likelihood function

A phylogenetic network N is just one parameter in an admixture graph $(N, \mathbf{\Theta})$. In order to search network space, we must define a procedures for estimating numerical parameters $\mathbf{\Theta}$ (i.e. the branch lengths in drift units and admixture proportions) and computing the likelihood of the resulting model given the input data.

Without loss of generality, we can define the likelihood function used by TreeMix (and related methods) in terms of f_2-statistics. Given a model admixture graph $(N, \mathbf{\Theta})$, we can compute the expected value of the f_2-statistics directly, up to a scaling factor (Patterson *et al.*, 2012; Peter, 2016; Pickrell and Pritchard, 2012). If N is a tree, then the f_2-statistic for populations $i, j \in S$ is the sum of the lengths (in units of genetic drift) of edges on the path from the leaf labeled i to the leaf labeled j in N. If there is admixture, then this formula is more complex, as we must consider all paths between i and j and use the admixture proportions to weight these paths appropriately.

In practice, $(N, \mathbf{\Theta})$ are unknown, and each f_2-statistic is estimated from allele frequency data, along its standard error. We let $\mathbf{X}$ and $\mathbf{Z}$ denote the vector of observed f-statistics and the variances, respectively. Given our input $\mathbf{X}$ and a network N, we can optimize $\mathbf{\Theta}$ to minimize the difference between the observed f_2-statistics $\mathbf{X}$ and the expected value of the f_2-statistics for $(N, \mathbf{\Theta})$, denoted $\mathbf{Y}$. Assuming X_i is normally distributed with mean Y_i and variance Z_i, the composite likelihood function is

$$\log \ell(N) = -\frac{1}{2}\sum_i \left(\frac{(X_i - Y_i)^2}{Z_i^2} + 2\log(Z_i) + \log(2\pi) \right) \quad (1)$$

. This (approximate) log-likelihood function used by TreeMix is related to the log-likelihood score used by qpGraph and miqograph.

As discussed by Pickrell and Pritchard (2012) and others (e.g. Lipson, 2020), the three branch lengths incident to an admixture node cannot be estimated simultaneously. Automated methods typically estimate (i.e. fit) exactly one of the three branch lengths, e.g. admixturegraph estimates the length of the outgoing edge, setting the lengths of the two (incoming) admixture edges to 0. In contrast, TreeMix sets the length of the outgoing edge and one of the two incoming admixture edges to 0 (note that TreeMix uses a tree-based labeling ψ of N to determine which admixture edges are set to 0 and which are estimated). Lastly, we note that the position of the root is not identifiable, except that it must be ancestral to admixture nodes.

3 Pitfalls of STB-ML methods

In this section, we show that there exist simple demographic models for which STB-ML methods using the likelihood function given Equation (1) are guaranteed to get trapped in a local optimum and thus return an incorrect network.

Our case study model ($M1$), shown in Figure 1a, has five populations at the leaves, one of which is admixed. The network topology can be created by forming a caterpillar tree with leaves alphabetically labeled by the set $S = \{A, B, C, D, E\}$, rooting the tree at population E, and then performing an edge addition (Definition 2) on the edges incident to D and A so that A is admixed. In the remainder of this section, we use $(N^*, \mathbf{\Theta}^*)$ to denote the network topology and numerical parameters for model $M1$.

Given $(N^*, \mathbf{\Theta}^*)$, it is possible write down the f_2-statistics implied by this model, denoted $\mathbf{X}^*$. As the amount of data generated under $(N^*, \mathbf{\Theta}^*)$ goes to infinity, the estimated f_2-statistics converge to $\mathbf{X}^*$, up to a scaling factor, and the standard error for every f_2-statistic goes to zero. (Note that it is typically shifted by a small constant to avoid numerical issues.)

Evaluating the likelihood of a candidate network N requires optimizing the numerical parameters $\mathbf{\Theta}$. To verify that our results were agnostic to the choices in parameter estimation by performing computational experiments using admixturegraph and TreeMix. Recall that admixturegraph optimizes the lengths of tree edges only, whereas TreeMix uses a tree-based labeling ψ, optimizing the lengths of the edges that are part of the base tree and whose tails (source vertices) are not admixture nodes (note that we evaluated likelihood using all tree-based labelings for a network). Scripts used in this study are available on Github: https://github.com/ekmolloy/mlno-study.

We begin our computational study by addressing whether the network topology N^* has the highest likelihood of all phylogenetic networks on S with one admixture event. We found that N^* has the highest likelihood through brute force, using all_graphs function in admixturegraph to generate all such topologies and computing their likelihoods with both admixturegraph and TreeMix.

We now turn to the performance of STB-ML methods given $\mathbf{X}^*$ as input. Because $M1$ contains a single admixture event, an STB-ML method would operate by (1) searching for an ML starting tree N_0, (2) searching for an ML network N_1 in the edge addition neighborhood of N_0 and (3) searching from N_1 for a network with a higher likelihood by applying other search moves. Below, we use brute force to determine the outcome of these three steps.

Step 1:

The first step of an STB-ML method is to estimate a starting tree on the full set of populations; this is typically done using an ML heuristic or the NJ algorithm (Saitou and Nei, 1987). In our case study, the NJ tree for $\mathbf{X}^*$ is the same as the ML tree for $\mathbf{X}^*$, which

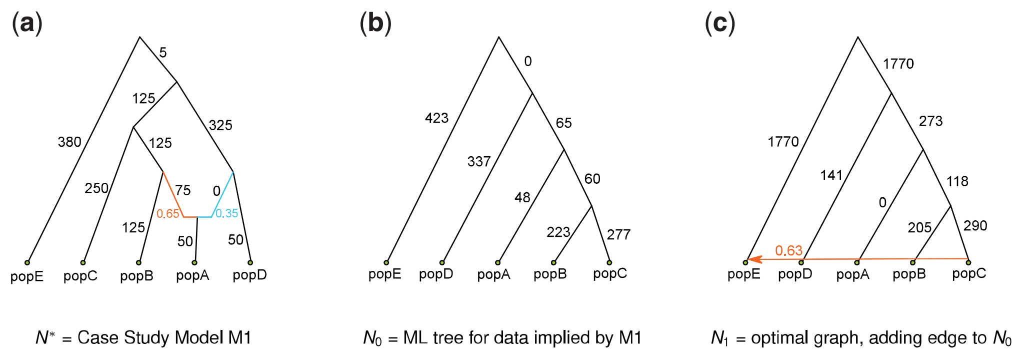

Fig. 1. Subfigure (a) shows the model admixture graph (N^*, Θ^*) discussed in Section 3. We refer to this model as M1, and let $\mathbf{X}^*$ denote the vector of f_2-statistics implied by this model. Subfigure (b) shows N_0: the ML (and NJ) tree for $\mathbf{X}^*$ rooted at the outgroup E. Subfigure (c) shows N_1: the ML network in the edge addition neighborhood of N_0. Although N^* has the highest likelihood of all networks with one admixture node, TreeMix and related STB-ML methods get stuck in a local optimum, returning N_1. Numerical parameters shown in subfigure (b) and (c) were computed using TreeMix. Log-likelihood scores, also computed using TreeMix, of N_0, N_1 and N^* are −1 302 625, −366 944 and 83, respectively. Branch lengths are shown multiplied by 1000

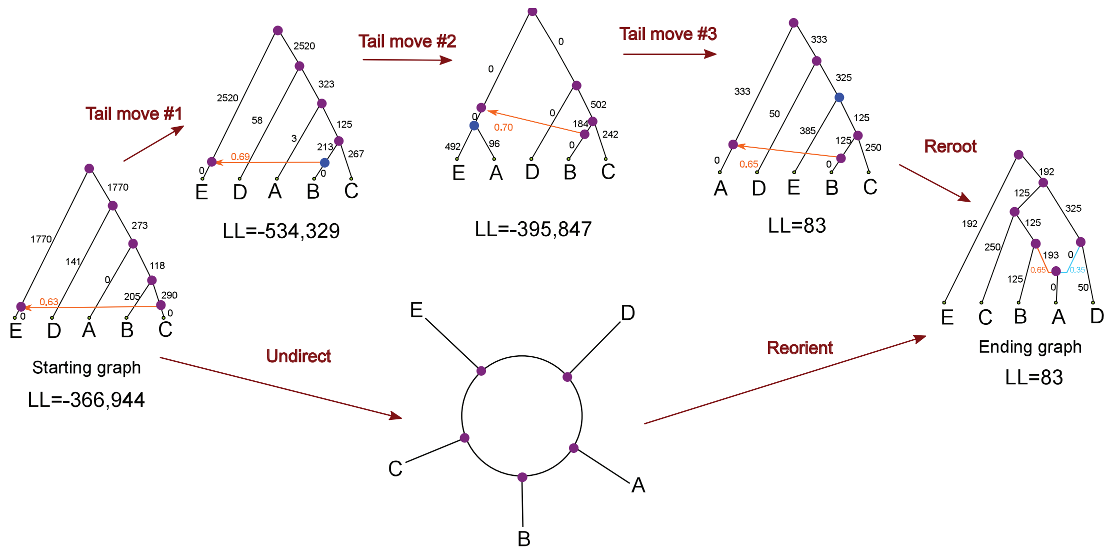

Fig. 2. The top of the figure shows a minimal sequence of three tail moves that transforms the starting network N_1 computed in step 2 into the ending network N^*. We found, through brute force analyses, that any sequence of tail moves from N_1 to N^* must traverse through a network with a lower likelihood score than N_1. Alternatively, N_1 can be transformed into N^* by selecting A as the admixed population (instead of E) and redirecting the edges of the network accordingly. Adopting the language of Huber *et al.* (2019), we refer to this as reorientating the network. This is not to be confused with rerooting the network, which simply relocates the position of the root and does not change which populations are admixed. Numerical parameters and log-likelihood scores shown above were computed using TreeMix. Branch lengths shown are multiplied by 1000

we found using brute force as described above. The NJ/ML tree rooted at outgroup E, denoted N_0 and shown in Figure 1b, is *not* a base tree of N^*. Note that TreeMix uses a random population heuristic to build a starting tree and may not be successful in finding the optimal tree in practice.

Step 2:

The second step of an STB-ML method is to search for an ML network in the edge addition neighborhood of N_0, which has size $O(|E(N_0)|^2)$ (Definition 2). We used brute force to find the optimal network topology in this neighborhood. The ML network, denoted N_1 and shown in Figure 1c, has E as the admixed population (note that we confirmed this outcome for all possible rootings of N_0).

Step 3:

The third step of an STB-ML method is to search from N_1 for a network with a higher likelihood via hill climbing. In our brute force evaluation of all networks with one admixture node, we found that N_1 has the second highest likelihood score, second only to N^*; therefore, it is sufficient to show that N_1 cannot be transformed into N^* via a single search move. We consider search moves that are commonly implemented in STB-ML methods, namely tail moves (Definition 3) and head moves. A head move relocates the head (target vertex) of an admixture edge to some other edge in the network, so we can verify by inspection that N_1 cannot be transformed into N^* with head moves. To evaluate whether this can be achieved with tail moves, we implemented a subroutine that given a network N finds all networks in its tail move neighborhood. Applying this subroutine iteratively, we found that at least three tail moves are required to transform N_1 into N^*, modulo the position of the root, which as previously mentioned does not impact the likelihood score (note that we confirmed this outcome for all possible rootings of N_1). As expected from our brute force analysis, this sequence of tail moves, shown in Figure 2, requires moving from N_1 to graphs of lower likelihood, so it is impossible to reach N^* from N_1 via hill climbing with head moves and tail moves. We conclude that our STB-ML method returns N_1.

4 The MLNO problem and OrientAGraph

In the previous section, we showed through a series of computational experiments that even for a simple model admixture graph, with just one admixed population, typical STB-ML methods can get trapped in a local optimum and return an incorrect network. In a purely hill climbing approach, we must move from incorrect network N_1 to the correct network N^* in a single step. One possible solution is to define a search move consisting of three tail moves. The 3-tail move neighborhood of N_1 has size $O(|E(N_1)|^6)$, and it seems unlikely that a random 3-tail move would take us from N_1 to N^*. Interestingly, we can transform N_1 into N^* simply by selecting A to be the admixed population and redirecting the edges of the network accordingly (Fig. 2). This search strategy of considering different populations as admixed has been used by researchers (e.g. Lipson, 2020) to manually explore network space. There is a connection between this approach and network orientation, because the admixture nodes (along with a valid root position) uniquely determine the orientation of an undirected binary network (Huber *et al.*, 2019). In other words, researchers manually search for a network orientation with the highest likelihood. Integrating this manual process into automated methods is appealing and leads us to propose the MLNO problem.

Definition 6 (MLNO problem). Let N' be an undirected phylogenetic network, and let $\mathcal{N}(N')$ be the orientation neighborhood of N' i.e. the set of directed phylogenetic networks such that $N \in \mathcal{N}(N')$ implies that the undirected version of N, denoted $N|_u$, is isomorphic to N'. We say that a directed network N^* is a maximum likelihood orientation of N', if N^* is in the set $\text{argmax}_{N \in \mathcal{N}(N')} \ell(N)$.

Here, we take $\ell(N)$ to be the likelihood function given in Equation (1).

The most straightforward approach to finding the MLNO is exhaustive search, typically initiated from some directed network N with h admixture nodes. The orientation neighborhood of $N|_u$ is defined by all ways of selecting h admixture nodes from the set $V(N|_u) \setminus L(N|_u)$ and all ways of selecting a root edge from $E(N|_u)$ (Huber *et al.*, 2019). We typically consider networks that are rooted at the outgroup g, denoting this set $\mathcal{N}_g(N|_u) \subset \mathcal{N}(N|_u)$. In a brute force approach, we evaluate the likelihood of every directed network $M \in \mathcal{N}_g(N|_u)$; this requires optimizing the parameters for M, denoted $\mathbf{\Theta}_M$.

As discussed in Sections 2.2 and 3, methods, such as admixturegraph and TreeMix, differ in how they optimize parameters, with TreeMix requiring a tree-based labeling ψ for M. This labeling is lost in reorientating a network, but a tree-based labeling, if one exists, can be found in linear time (Francis and Steel, 2015). An issue here is whether different tree-based labelings will yield different likelihood scores or have downstream effects on the search algorithm; however, when h is small, it is possible to evaluate all 2^h labelings (whether or not a given labeling is tree-based can be verified in linear time), as described by Francis and Steel (2015). We discuss this issue further in Sections 6 and 7.

In any case, an exhaustive search for an MLNO is only feasible when h and $|V(N|_u)|$ are sufficiently small. When $h = 1$, the orientation neighborhood $\mathcal{N}_g(N|_u)$ can be generated in $O(|E(N|_u)| \times |V(N|_u) \setminus L(N|_u)|$ time, as the algorithm from Huber *et al.* (2019), which scales linearly in the $|E(N|_u)|$, can be used to reorient $N|_u$ for all ways of selecting one admixture node from the set $V(N|_u) \setminus L(N|_u)$. For each $M \in \mathcal{N}_g(N|_u)$, we can compute the likelihood for all tree-based labelings, as there are at most two. When h is not fixed, we conjecture that finding the MLNO of $N|_u$ is NP-hard, in which case heuristic search will be necessary.

We conclude this section by describing how we incorporate MLNO within the latest version of TreeMix (v1.13 Revision 231), referring to our implementation as OrientAGraph. To estimate an admixture graph $(N, \mathbf{\Theta})$ with h admixture events, the following steps are taken.

```
1. Search for an ML starting tree $N_0$, rooting it at the outgroup.
2. For $i = 1, 2, \ldots, h$:
   a. $(N_i, \mathbf{\Theta}_{N_i}) \leftarrow$ Search for an ML network in the (gene flow) edge addition neighborhood of $N_{i-1}$.
   b. $(N_i, \mathbf{\Theta}_{N_i}) \leftarrow$ Search from $N_i$ for a network of a higher likelihood using tails moves.
   c. $(N_i, \mathbf{\Theta}_{N_i}) \leftarrow$ Search for an ML network in the (outgroup-rooted) orientation neighborhood of $N_i|_u$.
3. Return $(N_i, \mathbf{\Theta}_{N_i})$.
```

All three parts of step 2 apply operations (edge addition, tail moves, reorientations) to a network with the goal of searching for an ML network. Our case study focuses on the utility of MLNO; however, as observed in our experimental study, the relative effectiveness of these operations depends on the dataset. The details of this approach are provided in Supplementary Materials, but we summarize the differences between TreeMix and OrientAGraph below (note that steps 1, 2b, and 3 are the same in both methods).

For step 2a, TreeMix searches a subset of the edge addition neighborhood, which we refer to as the gene flow edge addition neighborhood. Specifically, edge additions must occur between pairs of edges that are both labeled as part of the base tree, with the linking arc labeled as gene flow; the tree-based labeling of N_{i-1} is then extended to N_i in the natural way. There are still $O(|E(N_i^2)|)$ networks in this space, and TreeMix uses a heuristic to identify a subset of them that seem promising. In OrientAGraph, we enable an exhaustive search of the gene flow edge addition neighborhood. Regardless of whether a heuristic or exhaustive search is performed, the legality of edge additions still needs to be evaluated (e.g. an edge addition cannot produce cycles). Testing for legality can be sped up by performing a dynamic programming preprocessing phase prior to initializing the exhaustive search (Algorithm 1 in the Supplementary Materials). However, an exhaustive search will still be prohibitively expensive when the number of populations is large.

For Step 2c, OrientAGraph executes an exhaustive search for an MLNO. [Note this step can be performed by evaluating a single tree-based labeling found by applying the algorithm of Francis and Steel (2015) or by evaluating all possible labelings.] In the next section, we evaluate the utility of MLNO for enabling STB-ML methods using f-statistics to escape local optima on modestly-sized datasets. An exhaustive search for the MLNO will not scale to datasets with a large number of admixture events; we discuss how scalability might be addressed and related considerations in Section 7.

5 Experimental study

In our experimental study, we utilized admixture graph models published in prior studies to benchmark TreeMix, OrientAGraph and miqograph, all of which take f-statistics as input. The exact f-statistics implied by the models were given to these methods as input, representing the case of infinite data. Method performance on these model datasets can be attributed to the method itself rather than error or bias in the input. To evaluate whether the trends observed for TreeMix and OrientAGraph extended to finite data, we benchmarked these methods on genomes simulated under a subset of the models.

5.1 Model datasets

We used the admixturegraph R package (Leppälä *et al.*, 2017) to create the f-statistic datasets implied by our case study model (M1) as well as seven other models (Fig. 3 and Supplementary Fig. S1). Model M2 is based on the toy example shown in Figure 5 of Patterson *et al.* (2012) used to benchmark qpGraph. All other models are based on admixture graphs estimated from biological datasets in prior studies; see Figure 3 for details. Note that M5 has the same network topology as M1; however, M1 is a toy example, whereas M5 was estimated on a biological dataset by Lipson (2020). For these experiments, the standard error of the f-statistic was set to 0.0001 to avoid numerical problems.

5.2 Simulated datasets

We simulated datasets from three demographies from models (M1, M5 and M6) using *ms* (Hudson, 2002) with the number of loci set to 3000, the number of sites per locus set to 1 Megabase, the effective population size above the root set to 10 000, the sample size for

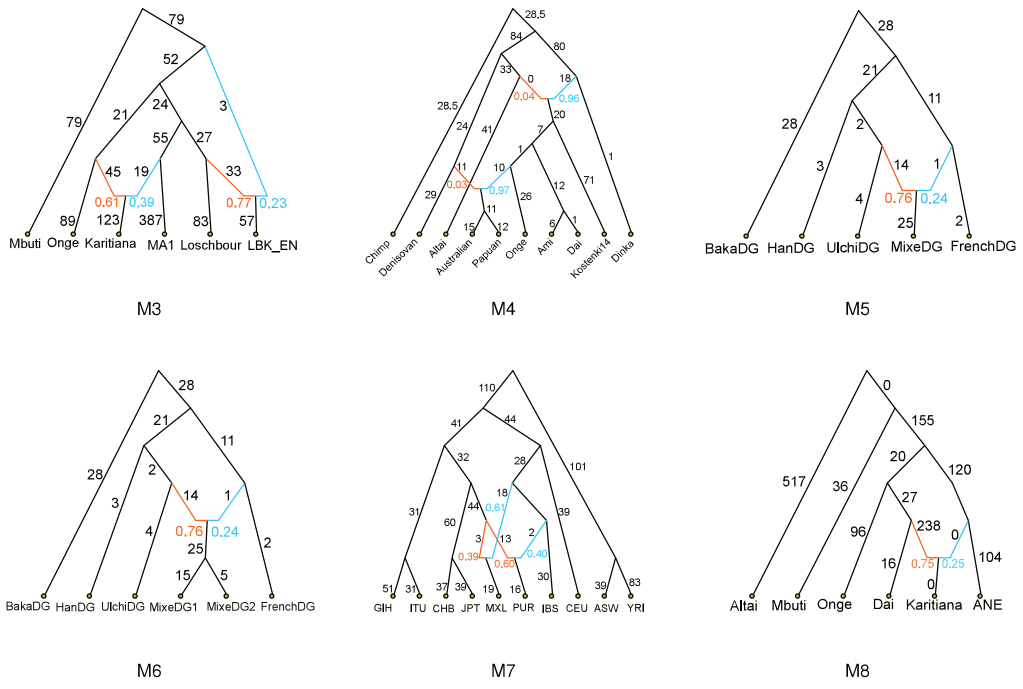

Fig. 3. This figure shows the model demographies evaluated in our experimental study. M3 is based on Figure S8.1 from Haak *et al.* (2015), M4 is based on Figure S11.1 from Mallick *et al.* (2016) but simplified so that there are only two admixture events, M5 is based on Figure 5a from Lipson (2020), M6 is M5 but extended so that Mixe split into two populations, M7 is based on Figure 7a from Wu (2020) and M8 is based on Figure 2 from Yan *et al.* (2020). All of these admixture graphs were estimated from biological datasets. Note that the branch lengths in these subfigures are not drawn to scale and are shown multiplied by 1000

each population set to 20, the mutation rate set to 1.25×10^{-8}, and the recombination rate set to 1×10^{-8}. The models based on biological datasets had branch lengths given in units of genetic drift and were not dated, so we did not know the timing of each internal node. To simulate data with *ms*, we assigned timings at the internal nodes and then rescaled the population size on each branch so that it had the correct length in drift units. The maximum number of SNPs per locus was approximately 6000, 3000 and 3500 for M1, M5 and M6, and we set the block size accordingly, when we used TreeMix to estimate f_2-statistics, along with their standard error. (Note that we also performed experiments computing the covariance matrix instead of f_2-statistics to confirm that this did not impact results.) Lastly, we repeated this process of estimating matrices with TreeMix using SNPs from the first 100, 500, 1000, 1500 and 2000 loci. This produced a total of 18 simulated datasets.

5.3 Method evaluation

Our primary focus was comparing TreeMix and OrientAGraph. Recall that OrientAGraph is implemented on top of TreeMix with two different subroutines: one for exhaustive gene flow edge additions and one for MLNO (Section 4). We executed these subroutines together as well as individually to explore their relative impact. Note that we exhaustively evaluated potential tree-based labelings during the MLNO subroutine. Otherwise, methods were run with the same options: specifying the outgroup, specifying the (correct) number of admixture events and implementing additional search moves from the starting tree, after reconstructing it with random population addition. In addition, we generated all possible population addition orders, selected 100 uniformly at random and ran all TreeMix and OrientAGraph using each of these orders.

Unlike TreeMix or OrientAGraph, miqograph is guaranteed to find the true admixture graph topology N^*, provided that N^* meets some topological constraints. We could not run miqograph on our cluster (because of how the industrial solver used by miqograph handles academic licenses), so we ran miqograph on the model datasets only. To compare running times, all analyses of model datasets were performed on the same shared computing resource [Intel(R) Xeon(R) CPU 2.10 GHz server with 128 GB RAM and 32 cores]. We allowed miqograph to use all available threads (note that TreeMix and OrientAGraph are single threaded) and gave it the same information as TreeMix and OrientAGraph, e.g. the outgroup and the (correct) number of admixture events. miqograph also requires the user to specify the granularity with which admixture proportions can be estimated (we set this value to 4) and the depth of its 'search tree' (we set this value to half the number of leaves in a given model).

Methods were compared in terms of three measures: log-likelihood score of the estimated admixture graph (for TreeMix and OrientAGraph only), topological accuracy, specifically triplet distance between the true and estimated admixture graph topology (Jansson *et al.*, 2019) and runtime (in seconds). For TreeMix and OrientAGraph only, we also plotted the estimated admixture graphs (for one population order) and its residuals for each model dataset (see Supplementary Materials). This was done to verify that different likelihood scores corresponded to different residuals and that a triplet distance of 0 corresponded to the true admixture graph being returned.

6 Results and discussion

6.1 Model datasets

In this section, we report the results of running TreeMix, OrientAGraph and miqograph on model datasets (Table 1).

Models M2, M3 and M8: For these three model datasets, all methods recovered the true admixture graph topology and did so quickly.

Model M1: For our case study model dataset, we observed the results expected based on Section 3. TreeMix recovered an incorrect topology, whereas OrientAGraph recovered the true topology. miqograph also recovered the true topology but was 2 orders of magnitude slower than OrientAGraph.

Model M4: Model M4 is an admixture graph presented with the publication of Simmons Genome Diversity dataset (Mallick *et al.*, 2016); however, we simplified the published model so that it only

Table 1. Results for estimating admixture graphs given the *f*-statistics implied by the models in Figure 3 using TreeMix, OrientAGraph (OAG) and miqograph (miqo)

Model	Log-likelihood		Correct topology?			Topological error			Running time (s)		
	TreeMix	OAG	TreeMix	OAG	miqo	TreeMix	OAG	miqo	TreeMix	OAG	miqo
M1	−366 944	83	No	Yes	Yes	9	0	0	0.06	0.19	398
M2	83	83	Yes	Yes	Yes	0	0	0	0.04	0.15	802
M3	124	124	Yes	Yes	Yes	0	0	0	0.28	2.12	2340
M4[a]	103[b]	373	No	Yes	No	14	0	81[c]	1.28	11.72	10 000
M5	−33	83	No	Yes	Yes	9	0	0	0.05	0.16	170
M6[a]	−30	124	No	Yes	No	15	0	2	0.11	0.31	556
M7	−2883	−2883	No	No	No	18.6[b]	20	82[c]	5.58	13.20	10 000
M8	124	124	Yes	Yes	Yes	0	0	0	0.09	0.34	264

Note: We report the running time in seconds, the log-likelihood score (for TreeMix and OrientAGraph methods only) and the topological error (triplet distance) between the model admixture graph and the estimated admixture graph (we also note whether the correct admixture graph topology was returned). Running time is averaged across 100 runs for TreeMix and OrientAGraph.

[a]Indicates that miqograph cannot get the topology correct, because of the position of the admixture event(s).

[b]Indicates there was a difference in the log-likelihood score or triplet distance for the graphs returned on different population orders, in which case, we report the average.

[c]Indicates that miqograph did not complete within the time limit of 10 000 s (2.78 h) and thus was not solved to optimality.

had two admixture events: gene flow from Denisova into the ancestors of Australians and Papuans as well as gene flow from Neanderthals into modern human, non-Africans. After 2.78 h, the graph returned by miqograph graph had a triplet distance of 81 to the true topology. It is unclear how much this score would improve with a longer running time, given that both admixture events in M7 are not incident to the leaves, so miqograph is guaranteed to return an incorrect topology. In contrast, TreeMix and OrientAGraph completed in <15 s, with a triplet distance of 15 and 0 to the correct topology, respectively. This difference was also reflected in the log-likelihood scores of the returned graph. These results were driven by the exhaustive search for an ML gene flow edge addition rather than an MLNO, suggesting that heuristic employed by TreeMix to identify candidate edge additions was ineffective for M4.

Models M5 and M6: On model datasets for M5 and M6, we observed similar trends to our case study (M1), that is, TreeMix returned an incorrect topology, which had a lower log-likelihood score than the true topology. Both OrientAGraph and miqograph returned the true topology, but again OrientAGraph was faster than miqograph.

Model M5 was estimated from Simmons Genome Diversity Project data (Mallick *et al.*, 2016) by Lipson (2020), who explored the space of admixture graphs manually using qpGraph. We were not aware of M5 when we designed our case study, but after finding this result, we created model M6 by extending M5 to have two populations descending from the admixed population (Mixe). This enabled us to evaluate whether the trends observed were unique to the depth of the admixed population; this was not the case as the trends observed for M6 were the same as for M5. However, miqograph is guaranteed to recover an incorrect topology for M6 (because of its topological constraints); this was reflected in our results.

Summary of models M1, M5 and M6: We confirmed that the trends observed for these three model datasets were driven by MLNO (and not an exhaustive search for an ML gene flow edge addition) by running these subroutines separately (Supplementary Table S1). We also performed exploratory analyses on these model datasets. For each model $(N^*, \boldsymbol{\Theta}^*)$ and input data $\boldsymbol{X}^*$ pair, we ran OrientAGraph given a base tree for N^* as its starting tree and using exhaustive search for ML gene flow edge additions. In this setting, OrientAGraph should recover the true admixture graph topology even without the MLNO; we confirmed that this was the case. We also computed the NJ tree for $\boldsymbol{X}^*$ to check that it was *not* a base tree of N^*; again, this was the case, so TreeMix could not be improved simply by taking the NJ tree as its starting tree.

Lastly, for each model, we scored the true admixture graph topology N^*, using all tree-based labelings. While the parameters around the admixture node differed (as expected), the likelihood scores were very similar. While some model parameters were not identifiable by TreeMix, the admixed population was identifiable. Specifically, in the orientation neighborhood of N^*, we found that OrientAGraph (which returned the true topology) and TreeMix (which returned an incorrect topology) achieved the highest and second highest likelihood scores, respectively. Importantly, the residuals showed that the true topology was a better fit to the data (Supplementary Materials Figs S3, S8 and S10). Note that these results also suggest that a single tree-based labeling could be used during MLNO, at least for simple models.

Model M7: Model M7 was estimated by Wu (2020), who ran GTMix given from data from the third phase of the 1000 Genomes project (The 1000 Genomes Project Consortium, 2015). Wu (2020) found that running TreeMix on a related dataset produced a different admixture graph topology than GTMix. However, GTMix and TreeMix use different inputs (gene genealogies versus *f*-statistics), different likelihood functions and different search heuristics (although both are STB-ML methods). Our question is whether TreeMix's performance could be related to its search procedure.

In principle, TreeMix can recover the true topology, as model M7 is tree-based; however, both TreeMix and OrientAGraph failed to recover the true admixture graph topology on this model dataset, returning graphs with a log-likelihood score of −2883 for all 100 different population orders. The triplet distance between the estimated graphs to the true graph varied (either 16 or 20). We scored the true admixture graph topology given the model dataset. This yielded a log-likelihood score of 373 (which is higher than −2883), so TreeMix is getting stuck local optimum. (Note that MLNO is ineffective in this scenario.)

miqograph also fails, returning a topology with a triplet distance of 82 from the correct topology. Although miqograph can recover the correct graph as both admixture nodes are incident to leaves, it did not solve its problem to optimality within our allowed time frame of 10 000 s. This suggests that users may want to be wary of the results produced by miqograph, when it does not succeed in solving its problem to optimality. Interestingly, M7 is the only model that we studied that is where a single vertex is a source for two different admixture edges (i.e. it is tree-child); this model may be of particular interest to method developers.

Running time comparisons:

The difference in running time between OrientAGraph and TreeMix was not pronounced on the six model datasets with at

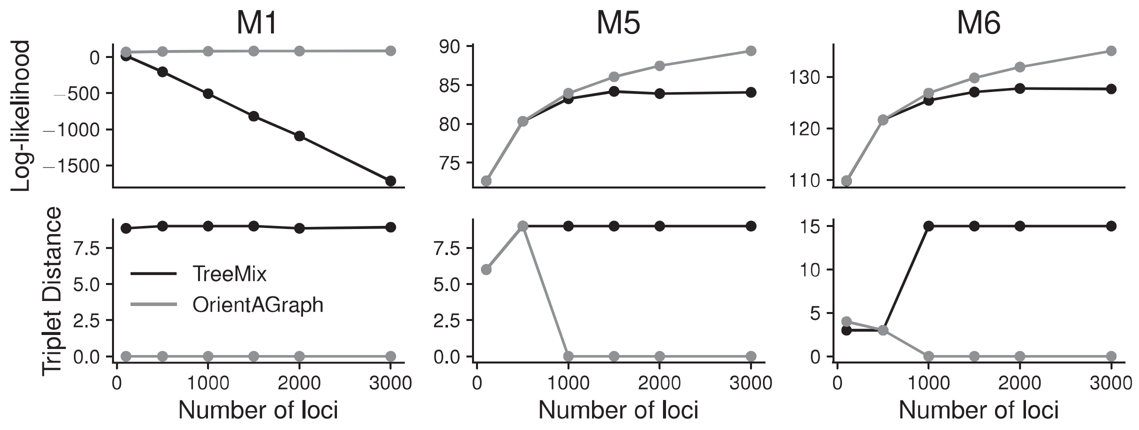

Fig. 4. Results from simulated data ranging from 100 loci to 3000 loci. Note that we are showing averages across runs of TreeMix for each population order (120 combinations for M1 and M5 and 720 combinations for M6). On these datasets, the results are consistent across runs so we only show the average. While we show both log-likelihood and triplet distance, we do not expect these measures to be fully correlated (i.e. lower triplet score may not imply lower log-likelihood.)

most 7 populations at the leaves and 2 admixture events (see Table 1). On these datasets, miqograph was slower than either OrientAGraph and TreeMix but still solved its problem to optimality. miqograph did not solve its problem to optimality within the allowed time frame for our study (10 000 s) on the remaining two model datasets, with 10 populations and 2 admixture events. On these datasets, there was noticeable slow down in the running time of OrientAGraph compared to TreeMix. These results are expected as OrientAGraph is the same as TreeMix but does more work; we discuss the issue of scalability in Section 7.

6.2 Simulated datasets

We used genome-scale datasets simulated from models M1, M5 and M6 to evaluate whether the trends observed for TreeMix and OrientAGraph on model datasets (representing infinite data) extended to the case of finite data (Fig. 4). This was the case: OrientAGraph recovered the true topology, and TreeMix recovered an incorrect topology, on simulated datasets with >1000 loci (Fig. 4). For M5 and M6, both TreeMix and OrientAGraph sometimes failed to recover the true topology for smaller numbers of loci ($\leq$1000 loci), likely due to differences between the estimated f_2-statistics and the expected f_2-statistics for the true admixture graph.

7 Conclusions and future work

In this work, we proposed a new search strategy based on network orientation and explored its utility in the context of STB-ML methods that take *f*-statistics as input. Our current implementation, referred to as OrientAGraph, relies on an exhaustive search for an MLNO. This is compounded by finding a tree-based labeling, as OrientAGraph is implemented on top of TreeMix, which requires such a labeling to evaluate its likelihood function. This points to the broader challenge of fitting numerical parameters from *f*-statistics in an automated fashion. Scalability might be improved by implementing a procedure for fitting model parameters that does not require a base tree (e.g. admixturegraph). However, this does not address the fact that there are inherent limitations in terms of parameter identifiability. As shown by Lipson (2020), it is possible for two admixture graphs with different topologies to fit the observed *f*-statistics perfectly and thus have the same likelihood score [Equation (1)]. We worked to mitigate the issue of parameter identifiability in our case study and computational experiments in two ways: first, we evaluated whether differences in topological accuracy corresponded to differences in likelihood scores and residuals, and second, we selected model admixture graphs based on the topological considerations discussed by Lipson (2020).

Beyond the likelihood function, method performance can be impacted by error or bias in the input data and the search heuristic. In this work, we focused on the search heuristic by considering the case of infinite data (as well as finite data) and by performing some additional experiments. For example, we found that MLNO was unnecessary to recover true admixture graph topology N^* when TreeMix was given a base tree of N^* as its starting tree. We also confirmed that the NJ tree for the f_2-statistics implied by the true admixture graph was not a base tree of N^*, so the issue is not specific to searching for an ML starting tree. One possibility is that methods based f_2-statistics are particularly susceptible to bad starting trees given the information content of their input. This could explain why a recent study by Cao *et al.* (2019) found an STB-ML method that takes estimated genealogical trees as input to be relatively accurate. Genealogical trees have more information content, especially when rooted; however, estimating them accurately is challenging from both a computational and statistical perspective. Furthermore, likelihood functions based genealogical trees are far more computationally intensive than those based on *f*-statistics. In any case, exploring the utility of MLNO for ML methods that use different inputs and/or Bayesian methods that sample (rather than hill climb) network space are interesting directions for future research.

Another important direction is scalability. OrientAGraph, while efficient for the admixture graphs considered here, will not scale to large numbers of populations and/or admixture events. In this case, we need to constrain the search for an MLNO, perhaps by maintaining a set of nodes that are required to be admixed across iterations. For example, in iteration *i*, our search of the orientation neighborhood of N_i can be constrained to $O(|V(N_i)|)$ networks, if the subset of nodes that are admixed in network N_{i-1} are required to be admixed in network N_i. This amounts to reorientating network N_i 'around' the *i*th edge addition only. Such a heuristic seems promising in the scenario that admixture events are sufficiently decoupled (e.g. the network is level-1) but may not be effective otherwise. Broadly speaking, developing fully automated inference methods that are accurate yet scalable under complex admixture scenarios is an important direction for future work.

Acknowledgements

We thank the four anonymous reviewers for their comments that led to improvements in this work.

Funding

This work was funded by the following grants: NSF GRFP DGE-1650604 (A.D.), NIH R35GM125055 (S.S. and E.K.M.) and an Alfred P. Sloan Research Fellowship (S.S.).

Conflict of Interest: none declared.

References

Cao,Z. *et al.* (2019) Empirical performance of tree-based inference of phylogenetic networks. In: Huber K.T. and Gusfield,D. (eds.), *19th International Workshop on Algorithms in Bioinformatics, WABI 2019, September 8–10, 2019*, Niagara Falls, NY, USA, Vol. **143,** *LIPIcs*. Schloss Dagstuhl—Leibniz-Zentrum für Informatik, pp. 21:1–21:13.

Edelman,N.B. *et al.* (2019) Genomic architecture and introgression shape a butterfly radiation. *Science*, **366**, 594–599.

Francis,A.R. and Steel,M. (2015) Which phylogenetic networks are merely trees with additional arcs? *Syst. Biol.*, **64**, 768–777.

Gambette,P. *et al.* (2012) Quartets and unrooted phylogenetic networks. *J. Bioinform. Comput. Biol.*, **10**, 1250004.

Gambette,P. *et al.* (2017) Rearrangement moves on rooted phylogenetic networks. *PLoS Comput. Biol.*, **13**, e1005611.

Green,R.E. *et al.* (2010) A draft sequence of the neandertal genome. *Science*, **328**, 710–722.

Haak,W. *et al.* (2015) Massive migration from the steppe was a source for Indo-European languages in Europe. *Nature*, **522**, 207–211.

Harney,É. *et al.* (2021) Assessing the performance of qpAdm: a statistical tool for studying population admixture. *Genetics*, **217**, iyaa045.

Huber,K.T. *et al.* (2019) Rooting for phylogenetic networks. *arXiv* abs/1906.07430.

Hudson,R.R. (2002) Generating samples under a Wright-Fisher neutral model of genetic variation. *Bioinformatics*, **18**, 337–338.

Janssen,R. *et al.* (2018) Exploring the tiers of rooted phylogenetic network space using tail moves. *Bull. Math. Biol.*, **80**, 2177–2208.

Jansson,J. *et al.* (2019). Computing the rooted triplet distance between phylogenetic networks. In Colbourn,C. J., Grossi,R. and Pisanti, N. (eds.), *Combinatorial Algorithms*. Springer International Publishing, Cham, pp. 290–303.

Leppälä,K. *et al.* (2017) admixturegraph: an R package for admixture graph manipulation and fitting. *Bioinformatics*, **33**, 1738–1740.

Lipson,M. (2020) Applying f4-statistics and admixture graphs: theory and examples. *Mol. Ecol. Resour.*, **20**, 1658–1667.

Lipson,M. *et al.* (2013) Efficient moment-based inference of admixture parameters and sources of gene flow. *Mol. Biol. Evol.*, **30**, 1788–1802.

Lipson,M. *et al.* (2014) Reconstructing Austronesian population history in Island Southeast Asia. *Nat. Commun.*, **5**, 4689.

Mallick,S. *et al.* (2016) The Simons Genome Diversity Project: 300 genomes from 142 diverse populations. *Nature*, **538**, 201–206.

McDiarmid,C. *et al.* (2015) Counting phylogenetic networks. *Ann. Combinat.*, **19**, 205–224.

Patterson,N. *et al.* (2012) Ancient admixture in human history. *Genetics*, **192**, 1065–1093.

Peter,B.M. (2016) Admixture, population structure, and F-statistics. *Genetics*, **202**, 1485–1501.

Pickrell,J.K. and Pritchard,J.K. (2012) Inference of population splits and mixtures from genome-wide allele frequency data. *PLoS Genet.*, **8**, e1002967.

Pilot,M. *et al.* (2019) Global phylogeographic and admixture patterns in grey wolves and genetic legacy of an Ancient Siberian lineage. *Sci. Rep.*, **9**, 17328.

Saitou,N. and Nei,M. (1987) The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Mol. Biol. Evol.*, **4**, 406–425.

The 1000 Genomes Project Consortium (2015) A global reference for human genetic variation. *Nature*, **526**, 68–74.

Wen,D. *et al.* (2018) Inferring phylogenetic networks using PhyloNet. *System. Biol.*, **67**, 735–740.

Wu,Y. (2020) Inference of population admixture network from local gene genealogies: a coalescent-based maximum likelihood approach. *Bioinformatics*, **36**, i326–i334.

Yan,J. *et al.* (2020) miqoGraph: fitting admixture graphs using mixed-integer quadratic optimization. *Bioinformatics*, btaa988.

Yu,Y. *et al.* (2014) Maximum likelihood inference of reticulate evolutionary histories. *Proc. Natl. Acad. Sci. USA*, **111**, 16448–16453.