JOURNAL OF COMPUTATIONAL BIOLOGY
Volume 17, Number 6, 2010
© Mary Ann Liebert, Inc.
Pp. 799–818
DOI: 10.1089/cmb.2009.0201

`constNJ`: An Algorithm to Reconstruct Sets of Phylogenetic Trees Satisfying Pairwise Topological Constraints

FREDERICK A. MATSEN

ABSTRACT

This article introduces `constNJ` (constrained neighbor-joining), an algorithm for phylogenetic reconstruction of sets of trees with constrained pairwise rooted subtree-prune-regraft (rSPR) distance. We are motivated by the problem of constructing sets of trees that must fit into a recombination, hybridization, or similar network. Rather than first finding a set of trees that are optimal according to a phylogenetic criterion (e.g., likelihood or parsimony) and then attempting to fit them into a network, `constNJ` estimates the trees while enforcing specified rSPR distance constraints. The primary input for `constNJ` is a collection of distance matrices derived from sequence blocks which are assumed to have evolved in a tree-like manner, such as blocks of an alignment which do not contain any recombination breakpoints. The other input is a set of rSPR constraint inequalities for any set of pairs of trees. `constNJ` is consistent and a strict generalization of the neighbor-joining algorithm; it uses the new notion of maximum agreement partitions (MAPs) to assure that the resulting trees satisfy the given rSPR distance constraints.

Key words: evolution, molecular evolution, phylogenetic trees, viruses.

1. INTRODUCTION

Since the pioneering article of Sneath (1975), tens of thousands of articles have been published on the subject of "reticulate evolution." "Reticulate evolution" has generally come to mean evolution where genetic material for a new lineage may come from two or more sources, as in the case of recombination and hybridization. The Oxford English Dictionary (OED, 1989) defines "reticulated" to mean "constructed or arranged like a net; made or marked so as to resemble a net or network." Correspondingly, rather than evolutionary history being representable as a tree, a network is more appropriate. A considerable amount of effort has gone into the phylogenetic reconstruction of these networks.

Algorithms for phylogenetics in the presence of reticulation have followed a different path than the mainstream of phylogenetics. Current algorithms fall into three types. First, there are algorithms that attempt to find the phylogenetic network displaying some fixed characteristics (such as splits in an alignment or some set of trees) that contain the minimum number of reticulation events. Second, there are

Program in Computational Biology, Fred Hutchinson Cancer Research Center 1100, Seattle, Washington, USA.

algorithms to construct "splits networks," which do an excellent job of representing conflicting signals in the data, but do not give an explicit evolutionary history. Third is the full Bayesian approach to sampling trees and recombination breakpoints given the data.

None of these approaches yet furnish a practical solution for certain cases, such as HIV researchers who would like to reconstruct the evolutionary history of a reasonably large alignment which includes recombinant sequences. Indeed, first fixing a set of characteristics and that then minimizing the number of reticulation events ignores the balance between the number of reticulation events and phylogenetic optimality; the splits network approach does not tell a complete evolutionary story; and fully Bayesian algorithms are not yet sufficiently fast for DNA sequence datasets that have thousands of nucleotides and hundreds of taxa. HIV researchers who wish to reconstruct evolutionary history typically proceed in one of two ways: they either treat the whole alignment as having a single tree-like history, which cannot possibly be correct, or they build trees on sub-alignments independently, which does not take into account the underlying network structure. These two extremes, of assuming all trees have the same topology or allowing their topologies to differ in arbitrary ways, leave a substantial gap in the middle, where the correct balance of optimality and discord should be found.

The goal of the algorithm `constNJ` (constrained neighbor-joining) is to begin filling this gap in a manner analogous to classical phylogenetic inference algorithms. To do so, we make a different set of assumptions than is typically done considering the input and desired output. Regarding the data, we assume that that the given alignment has been segmented into "alignment blocks," each of which can be described by a single tree. For example, in the case of recombination, the alignment blocks are the segments of an alignment that do not contain recombination breakpoints. (Note that for the purposes of this article we will be using the word "recombination" in the general sense, including processes such as gene conversion.) Although the assumption that the data comes pre-segmented is a substantial one, we don't think that it is unreasonable. Sometimes a segmentation is clear, such as the distinct RNA strands of the influenza genome. Other times, such as for recombination, it is not so clear, but even in this more difficult case the inference of recombination breakpoints has seen significant progress in the last 10 years. We will also assume that an outgroup has been selected. Such a choice is crucial, as it establishes directionality for reticulation events.

Regarding desired output, rather than actually building a single reticulate network, this article will focus on building "correlated sets of trees," which display the sorts of constraints found on trees that fit in a reticulate network. We are focused on building trees because each alignment block is correctly described by a single tree. However, this set of trees must fit into a network, which forces constraints on their topology. Specifically, the trees which sit in these networks must be related by rooted subtree-prune-regraft (rSPR) moves, whereby a rooted subtree is cut from the original tree and then re-attached in another location (Fig. 1). We describe below how it is necessary for trees sitting in a reticulate network to be related by rSPR moves, though this is not a sufficient condition.

For `constNJ`, we assume that the user can supply a series of constraints describing the number of rSPR moves allowable between pairs of alignment blocks. For example, if the alignment contains "pure" types and a single class of recombinants that are derived from a pair of types, then there should be two alignment blocks, and the trees for those blocks should be related by one rSPR move, as in Figure 1. The

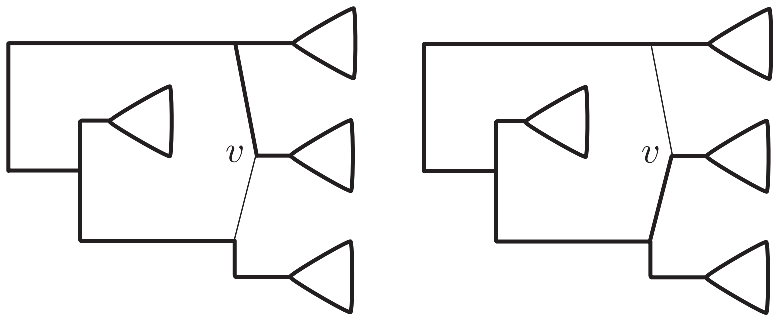

FIG. 1. An example "reticulate" network and the two trees that it contains. Those two trees are related by a single rooted subtree-prune-regraft (rSPR) move, whereby the middle subtree is cut off of the tree and reattached at another location. The node v will be called a reticulation node.

challenge, then, is to reconstruct a set of trees which satisfy the constraints and which together optimize some phylogenetically relevant criterion, such as likelihood, parsimony, or balanced minimum evolution. Note that constNJ actually constructs a number of such sets of trees, in order to display the balance between optimality of the individual trees and the number of reticulation events needed to fit the trees together into a network.

We now present a motivating example. The CRF14_BG circulating recombinant form (CRF) of HIV is known to be a mosaic of subtype B and subtype G viruses, and the breakpoints of the recombination events are known (Thomson et al., 2001). We will call the region of the alignment where BG derives from the G subtype the "G region," and the region where BG derives from the B subtype the "B region." As in Thomson et al. (2001) and all similar articles we could find on this subject, researchers build trees independently on the no-recombination blocks. We have repeated such an analysis in Figure 2, building PHYML maximum likelihood phylogenetic trees using the F84 model and rooted using CPZ.CD.90.ANT.U42720 (removed from tree for clarity). As one would hope, the trees do indeed show that the BG CRF derives one portion of its RNA from the G subtype, and the other part from the B subtype. However, there are many more differences between the two trees than should occur for an alignment with a single recombinant strain. For example, the rooting changes between the two trees, as does the location of the C and the F–K clades. Building a recombination network out of these trees would lead to a number of spurious hypothesized recombinations.

In contrast, for this dataset, constNJ returns a collection of pairs of trees displaying the balance between the number of allowed rSPR moves between pairs of trees and phylogenetic optimality. This balance is described in the text output of constNJ, which is shown in Table 1 for the BG dataset. The first column shows the rSPR distance between the two reconstructed trees (in this case, the G region tree and the B region tree). As described below, the notion of optimality for constNJ is total tree length, which is a trivial generalization of the balanced minimum evolution (BME) criterion (Desper and Gascuel, 2004). It is displayed in the second column for the pairs of trees. Thus, the second line states that constNJ found a pair of trees which differ by a single rSPR move, and which have total tree length of about 7.119. The third column shows the difference between the second column values between rows. Thus, 0.0942 signifies that there is a decrease of magnitude 0.0942 in total tree length by allowing a single rSPR difference between the two trees.

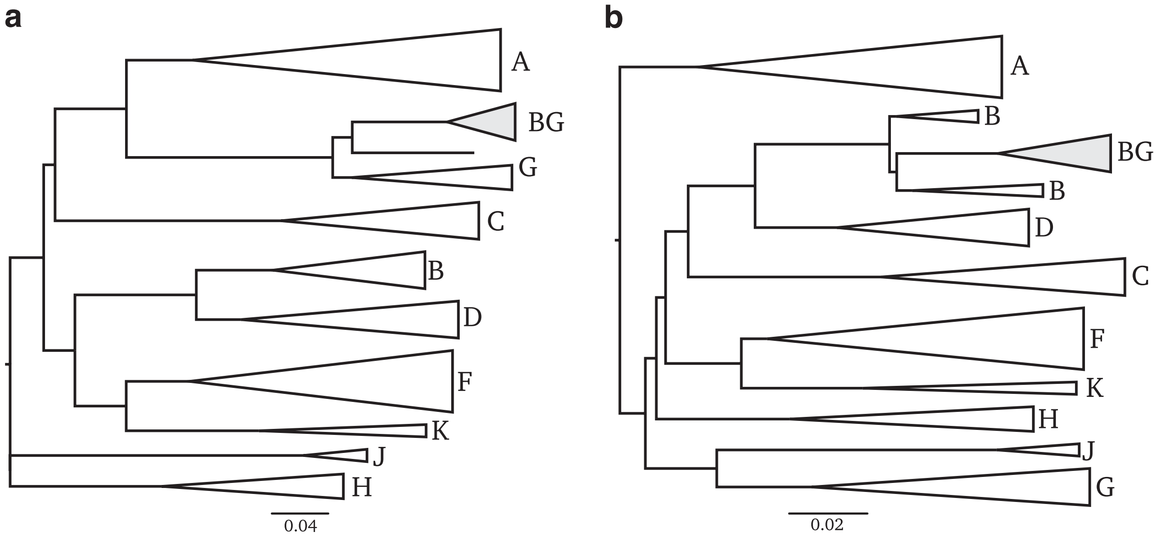

FIG. 2. Phylogenetic trees of the pure subtypes of human immunodeficiency virus (HIV) and the BG recombinant clade constructed independently using the no-recombination blocks of the HIV genome. The single letters (e.g., A, B, C) label clades of subtypes, and BG denotes a clade of circulating recombinant forms (CRFs) made from B and G subtypes. Tree (**a**) is built from the "G region," i.e., the region where the BG CRF derives from the G subtype. Tree (**b**) is built from the "B region," where BG derives from the B subtype. As noted in the text, although these trees do place the recombinant strains in the correct locations, they differ in a number of important ways that are not explained by recombination events. It is the perspective of this article that these extra differences represent phylogenetic error, and that accuracy can be improved by constraining the trees to fit into a recombination network.

Table 1. Balance between Discord and Optimality for the Example HIV Dataset

rSPR distance	*Total tree length*	*Tree length difference*
0	7.213	0.0942
1	7.119	0.0076
2	7.111	0.0149
3	7.096	0.0139
9 (independent NJ)	7.082	

On the left side is the number of SPR moves required to go from the tree built on the G region to the tree built on the B region. In the center is the total tree length (see Equation 2), which is our notion of optimality. On the right is the difference of the total tree length between the rows. As described in the text, the largest drop in total tree length comes when allowing a single rSPR move (i.e., recombination event) between the two trees, indicating that one recombination event is needed to explain the data.

HIV, human immunodeficiency virus; rSPR, rooted subtree-prune-regraft; NJ, neighbor-joining.

In this way, we can achieve an understanding of the balance between phylogenetic optimality and number of recombination events. For example, we can see that the decrease in allowing a single recombination event is significantly greater in magnitude than that for allowing two rather than one. And surprisingly, allowing nine rSPR moves does not significantly decrease the total tree length compared to allowing three. Because the improvement in total tree length when allowing one rSPR move is significantly greater than that for any subsequent rSPR moves, we believe that Table 1 suggests that the data probably arose from one recombination event, which agrees with the established knowledge concerning these taxa.

Furthermore, the trees which `constNJ` finds assuming a single recombination event agree with the accepted recombination history of the BG recombinant circulating form (Fig. 3). In particular, the only difference between them is the location of the BG clade, which switches from the G to the B subclade depending on the region analyzed. Importantly, these two trees can fit into a recombination network with a single reticulation node, in contrast to those in Figure 2.

We will now review the relevant literature, starting with common terminology. The currently accepted term for the class of networks including both hybridization and recombination networks is "reticulate network."[1] If we consider a rooted tree to be a directed graph such that edges are directed away from the root, a reticulate network is a rooted phylogenetic tree with additional directed edges making a directed acyclic graph with "tree nodes" of in-degree one and "reticulation nodes" of in-degree two (Huson et al., 2005).

A considerable amount of work has gone into the problem of constructing a reticulate network given a set of phylogenetic trees which it must contain. This problem was initiated by Maddison (1997), and considerable progress has been made by Baroni et al. (2005), Nakhleh et al. (2005), Huson et al. (2005), and Bordewich and Semple (2007a, b). As described above, we differ from these approaches as we would like to estimate the trees while ensuring that they fit into a reticulate network.

A related problem (which was the original motivation for fitting trees into a network) is to reconcile distinct gene trees into a single species tree. This problem has received an appropriately large amount of attention, and has found a more realistic model-based formulation in Ané et al. (2007) and Edwards et al. (2007). These differ from the present articles because they assume that there is a single species tree, and that "correctness" of a gene tree should in part be judged by the degree to which it fits within a species tree due to a coalescent model. In our setting, however, there is no single species tree, and the coalescent model may not be appropriate.

Sometimes a related assumption is made, which is not that complete species trees are known, but that the resulting recombination network must display a specified collection of bipartitions, which are typically called splits. This is equivalent to assuming a supplied alignment evolves according to the infinite sites model of mutation. The problem again is to find a network that minimizes the number of recombination events. This problem was first formulated by Hudson and Kaplan (1985) and was shown to be NP-hard in Wang et al. (2001). Progress was made in a sub-case by Gusfield et al. (2004), and a simpler related (and in some ways more realistic) problem was solved by Song and Hein (2005). Huson and Kloepper (2005) note that the algorithm in Huson et al. (2005) can be extended to this case. Although a different formulation, this

[1]This terminology is redundant, as the word "reticulate" already means network-like.

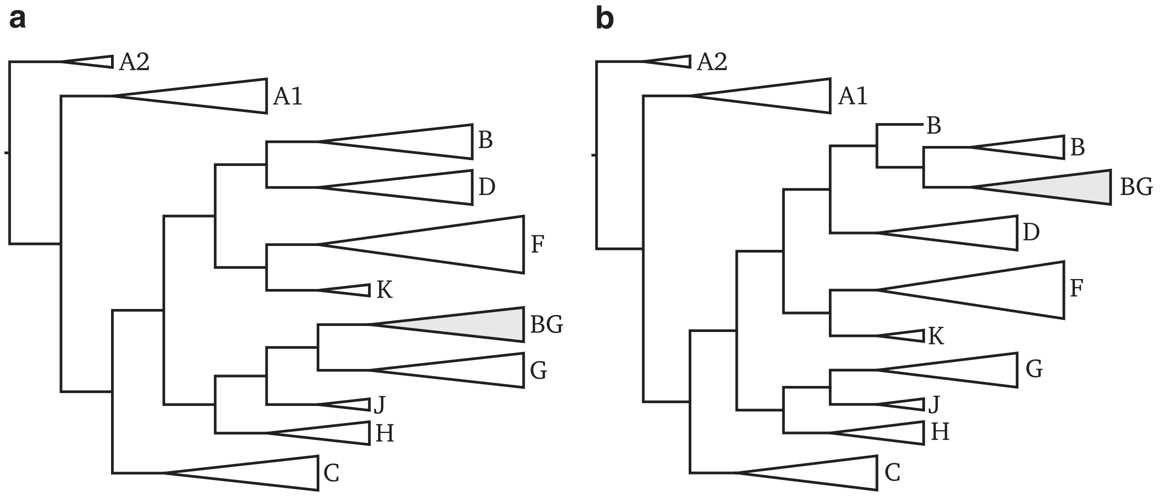

FIG. 3. One pair of phylogenetic trees (**a, b**) constructed for the same dataset using `constNJ`. In contrast to Figure 2, the only difference between the two trees is the location of the BG recombinant clade. These two trees fit into a recombination network with a single recombination event, as should be the case for a tree for the pure subtypes with a single recombinant strain like we have here. `constNJ` correctly identifies that the BG subtype is a recombinant of the B and G subtypes.

splits/infinite sites approach represents a different version of the same strategy: find the network displaying a certain set of characteristics that minimizes the number of reticulation events.

Splits network methods are a biologically useful and mathematically interesting way of understanding conflicting signals in phylogenetic data. The first method to construct splits networks from distance data was the split decomposition approach of Bandelt and Dress (1992a, b). Another successful approach has been the "neighbor-net" algorithm created by Bryant and Moulton (2004) and further analyzed by Levy and Pachter (2008). These methods form a useful complement to phylogenetic analysis in the traditional tree-based sense, but do not reconstruct an explicit evolutionary history. We also note that recombination networks need not be circular split systems, which are the sorts of splits networks returned by neighbor-net.

On the other end of the spectrum lie likelihood-based methods using the coalescent with recombination (Hudson, 1983). Major recent advances have been made in this area. The full likelihood is quite daunting to compute, but Lyngsø et al. (2008) have a parsimony-based approach that saves on computation by several orders of magnitude. Importance sampling (Griffiths et al., 2008) is also promising, but is not yet efficient enough for long alignments. Also, it is the intent of this article to construct a method that is independent of population genetics models such as the coalescent.

A related line of research is the inference of recombination breakpoints using phylogenetic methods. One of most basic and commonly used methods for the inference of recombination breakpoints is called "bootscanning", whereby a window is scanned along the alignment and a phylogenetic tree is built for each position of the window; a change in topology between sections of the window can be interpreted as evidence for a recombination breakpoint (Lole et al., 1999). Marc Suchard and collaborators apply multiple change-point models and reversible-jump MCMC to estimate trees and model parameters along the alignment, first without using the relationship between topologies on either side of a recombination breakpoint (Suchard et al., 2003; Minin et al., 2005), and more recently incorporating such information using a "random tips" model (Bloomquist et al., 2009). de Oliveira Martins et al. (2008) also propose a Bayesian algorithm, using an approximate SPR distance calculation to penalize discordant topologies. Westesson and Holmes (2009) develop a hidden Markov model for the phylogenetic action of recombination and use an EM-type algorithm to perform model selection, estimate parameters, and find optimal recombination breakpoints. Boussau et al. (2009) also investigate a phylogenetic hidden Markov model on tree topologies via more classical heuristic optimization of the likelihood.

Although it is common for breakpoint detection methods to use phylogenetic trees as part of their calculation, it is typically not the intent of these methods to accurately infer phylogeny. To do so, it can be helpful to use the correlation among trees between no-recombination blocks as in (de Oliveira Martins et al., 2008). An exciting recent development in this direction is the inference of ancestral recombination graphs

(ARGs) by Bloomquist and Suchard (2010) given an alignment which has been partitioned into no-recombination blocks. These authors develop new priors and transition kernels to obtain an MCMC method which is practically useful for moderately sized data sets. The ability to obtain dates and posterior probabilities for non-vertical evolutionary events ensures that this method will find wide use in practice; the difficulty of performing computation on ARGs will keep scientists working on theory and implementation busy for years to come.

In summary, there is a continued need for reticulate network inference algorithms that give an explicit rooted phylogenetic history for each column of the alignment, that elucidate the balance of discord between the trees and optimality for those trees, and that are efficient enough to be useful for modern data sets. The lack of practical phylogenetic algorithms in the presence of recombination was recently demonstrated in a simulation study by Woolley et al. (2008). Huggins et al. (2008) have noted the lack of useful reconstruction algorithms for host-parasite relationships and have noted the need for an algorithm that balances tree concordance and optimality as `constNJ` does. Although far from a complete solution for these cases, we believe that `constNJ` is one step towards fitting a substantial algorithmic need.

2. GENERAL DESCRIPTION OF `constNJ`

The primary input for `constNJ` is a collection of alignment blocks, which as described are disjoint subsets of columns of the alignment that are assumed to evolve in a tree-like manner. In the case of alignments with recombinant sequences, the alignment blocks are simply the no-recombination blocks. Note that the alignment blocks need not be contiguous; for example, a single recombination event with two recombination breakpoints will result in two, not three, alignment blocks. The other input for `constNJ` is a sequence of constraints on the rSPR distance between the trees constructed for the no-recombination blocks as described below. Given this input, the goal of `constNJ` is to exhibit the balance between discordance among the alignment-block trees on one hand, and optimality of the trees in some phylogenetic sense on the other.

`constNJ` is a deterministic distance-based approach to reconstruction; we chose this direction for several reasons. First, the underlying space for a likelihood optimization scheme is even larger than usual, making a heuristic search even less appealing: there are $[(2n-3)!!]^k$ k-tuples of rooted bifurcating phylogenetic trees on n taxa. The sorts of constraints we will be imposing reduces this number substantially, but little is known about the resulting graph under the sorts of moves typically used in heuristic phylogenetic searches. Furthermore, likelihood-based approaches are substantially improved by starting with a reasonable tree, which in modern applications is typically a distance-based tree. Thus, even if a likelihood-based approach was the eventual goal, a distance-based approach would be useful as a "seed" for the heuristic likelihood search. Finally, we feel that distance- and likelihood-based algorithms occupy distinct and complementary roles in the world of computational phylogenetics.

Our goal is to design an approach that generalizes the remarkably accurate and hugely popular neighbor-joining algorithm (Saitou and Nei, 1987). Surprisingly, it took almost 20 years for the phylogenetics community to learn the objective function of neighbor-joining; during that time, it was claimed that no such objective function existed. However, it is now known that neighbor-joining greedily optimizes the "tree length" $\ell(T, D)$ (defined below in Equation 1) for the given distance matrix D. `constNJ` generalizes this objective function, as it attempts to minimize the total length of all k trees (2) by a combination of greedy steps.

The trees resulting from `constNJ` are constrained by the user to be some specified number of rSPR moves from one to another. As displayed in Figure 1, reticulation events such as recombination and hybridization correspond to rSPR tree rearrangements. The converse is not true: arbitrary rSPR tree rearrangement events need not correspond to reticulation events. For recombination or hybridization to take place, the participants in the event need to exist at the same time; it is not hard to set up examples of rSPR move combinations that violate this fact; see, for example, Song and Hein (2005). Methods have been developed that take timing restrictions into account (Song and Hein, 2005; Bordewich and Semple, 2007a), but we do not incorporate these ideas into a phylogenetic reconstruction framework. This may be an interesting avenue for future research, but on the other hand seeing such timing violations can actually be informative. First, there may be something wrong with the data. Second, it has been noted by Baroni et al., (2006) that reticulation networks can appear to violate timing constraints if certain taxa are not sampled. The problem of determining the minimal number of "missing" taxa required to explain timing constraints has been analyzed by Linz et al. (2010). Therefore, we have left interpretation of timing issues up to the user of the program.

We now make a more formal statement of the problem that `constNJ` attempts to solve; note that a similar formulation was made independently by Huggins et al. (2008) in the context of host-parasite relationships.

Problem 1. (rSPR-constrained balanced minimum evolution). *Given k $n \times n$ distance matrices $D_1, \ldots, D_k$ and a symmetric $k \times k$ constraint matrix C, find the set of trees $T_1, \ldots, T_k$ minimizing $\sum_{i=1}^{k} \ell(T_i, D_i)$ such that $d_{\mathrm{rSPR}}(T_i, T_j) \leq C_{i,j}$ for each i and j.*

Theorem 2. `constNJ` *is a consistent algorithm to solve Problem 1.*

For `constNJ`, we proceed in a manner analogous to that for neighbor-joining. The neighbor-joining algorithm starts with all taxa connected to a central node and then, at every stage, chooses the "coalescence" (in other articles, "amalgamation") of trees which most decreases the value of the total tree length. We mimic this philosophy by evaluating coalescences based on how they affect the total tree length. However, in the end we must come up with a collection of trees $T_1, \ldots, T_k$ that satisfy the prescribed rSPR constraints. This raises the question of how one might bound the rSPR distance of the eventual trees "ahead of time," i.e., before the termination of the coalescence steps. For instance, if in the developing trees one has the subtrees (a, b) for the first distance matrix, and (a, c) for the second distance matrix, it is clear that the resulting trees must have rSPR distance of at least one between trees T_1 and T_2.

The question of how to bound eventual rSPR distance is solved by Theorem 19. Specifically, we generalize m, the size of the maximum agreement forest (Bordewich and Semple, 2005) to these partially coalesced trees, which forms a sharp bound. In short, the m value for a pair of partially coalesced trees T and S is the minimum rSPR distance possible among trees resulting from coalescences of T and S; thus, once a pair of partially coalesced trees achieves an m value above the corresponding constraint, we can throw that pair out, as the eventual resolved trees will never satisfy the constraints.

Using this m, we construct our greedy algorithm, as shown in Figure 4. Say that we only have two trees, and that we want to find the minimal-total-length pair of trees which are only one rSPR move apart. At every stage, we attempt to find the best pair of partially coalesced trees with m values zero and one. We start with two star trees; m applied to this pair is zero. The first-step coalescence must also lead to a pair of trees which have m value zero, as one of the trees is still completely unresolved. Say the optimal, in terms of total tree length, second step NJ-type coalescence leads to a pair of trees which have m value one (indicated by the first diagonal arrow in Fig. 4). Then we go down the list of second-step coalescences for the trees and find the best one that does not increase the m value at all (indicated by a horizontal arrow in Fig. 4). Next we repeat the process for each of the trees from the previous stage, saving the best pair of trees

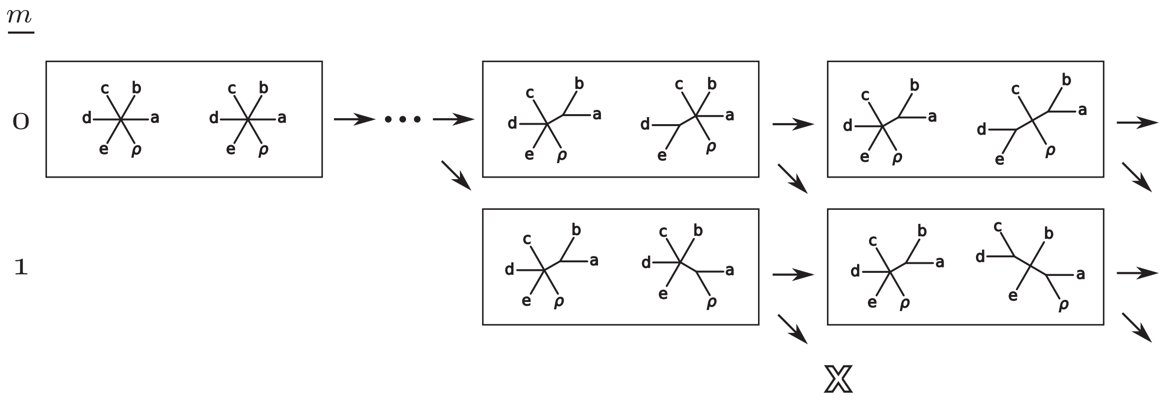

FIG. 4. Schematic diagram of the `constNJ` algorithm. As described in the text, at every stage we attempt to find the optimal pair of partially coalesced trees which could eventually be, at most, some fixed number of rooted subtree-prune-regraft (rSPR) moves apart. As shown in Theorem 19, the m value for a pair of partially coalesced trees forms a sharp lower bound for the eventual rSPR distance between those trees. Therefore, pairs of partially coalesced trees that have m value exceeding the constraint on rSPR distance can be thrown out, as shown by the X.

which have m values zero and one. In the end, we will have the best pair of trees which have rSPR distances zero and one which achievable via a series of greedy steps. Although not guaranteed to be the optimal pair of trees, the algorithm is consistent.

3. TECHNICAL PRELIMINARIES

In this section, we review some definitions and clarify our notion of optimality. As stated in the introduction, we will always assume that an outgroup taxon has been chosen and will label it ρ. Thus, we always assume that ρ is contained in any taxon set X. We will use the following definitions. For the purpose of this article, a *tree* on a finite taxon set X will be a rooted binary phylogenetic X-tree. A *forest* on taxon set X will be a collection of trees on disjoint taxon sets such that the union of the taxon sets is X. We will sometimes consider a tree on X to be a forest with a single tree. An *unrooted tree* on a finite taxon set X will be an unrooted phylogenetic X-tree (note that unrooted trees will be allowed to have multifurcating nodes). $\mathcal{L}(R)$, $\mathcal{E}(R)$, and $\mathcal{V}(R)$ will denote the leaves, edges, and vertices of a tree, unrooted tree, or forest R.

Although ρ represents the true rooting of the phylogenetic tree, we will not always assume that our trees or forests are rooted at ρ. We must do so because the NJ-type coalescences will not in general root the tree at the edge leading to ρ. Therefore, we must allow alternative rootings, but at the same time keep in mind that the rSPR distance between the trees must be calculated with respect to the edge leading to ρ. Thus, we use the following definition of rSPR on an unrooted tree: given an unrooted tree U on a taxon set $X \ni \rho$, a single SPR move first cuts some edge of the tree except for that leading to ρ, resulting in two rooted trees R and S. Say $\rho \in \mathcal{L}(R)$. Suppress the degree two root node of R, and attach S to some edge of the resulting unrooted tree by inserting a degree two node onto the chosen edge, then connecting the root of S to that new node. This definition is the same as that of Bordewich and Semple (2005) when considering trees rooted at the edge leading to ρ.

As with any distance function defined implicitly in terms of a graph, the minimum number of rSPR moves required to transform one tree T into another S is a metric; we define $d_{\mathrm{rSPR}}(T, S)$ to be this number.

3.1. Tree length and the balanced minimum evolution criterion

As reviewed by Gascuel and Steel (2006), phylogenetics researchers now understand the optimality function of the neighbor-joining algorithm (Saitou and Nei, 1987). Let $\mathrm{p}(i, j)$ denote the path from i to j in the unrooted tree T, and define the *weight* of a path from leaf i to leaf j as

$$w(i,j) = \prod_{v \in \mathrm{p}(i,j)} \frac{1}{\deg(v) - 1}.$$

Then the "length" of an n taxon tree T with respect to an $n \times n$ distance matrix D is as follows (Semple and Steel, 2004):

$$\ell(T, D) = \sum_{i,j} w(i,j) D_{i,j}. \tag{1}$$

The name "tree length" comes from the fact that if D is a distance matrix derived from some assignment of branch lengths to the edges of T, then ℓ will be the total length of all of the edges. However, the name may be somewhat confusing initially, because ℓ need not be defined as the sum of the branch lengths of any specific tree.

The tree T that minimizes $\ell(T, D)$ for some distance matrix D is the BME tree for the distance matrix D. The BME criterion is consistent (Desper and Gascuel, 2004), and neighbor-joining is a consistent tree-building heuristic that greedily minimizes total tree length (Desper and Gascuel, 2005) As described in Problem 1, `constNJ` attempts to minimize

$$\sum_{i=1}^{k} \ell(T_i, D_i) \tag{2}$$

while enforcing pairwise constraints on the rSPR distance between pairs of trees. When $k = 1$, `constNJ` is simply neighbor joining, while for $k > 1$, `constNJ` is a strict generalization of NJ.

4. ROOTED SPR AND MAXIMUM AGREEMENT PARTITIONS

This section describes the primary technical content of this article. As described in the introduction, we would like to proceed via coalescences in a manner similar to neighbor-joining, while ensuring that the eventual rSPR distance between the trees is not too large. In order to assure adherence to the rSPR criterion, we develop the notion of maximum-agreement partition, which generalizes the notion of maximum agreement forest from Bordewich and Semple (2005). As shown in Theorem 19, maximum agreement partitions and the associated m value allow us to bound the rSPR distance between the two partially resolved trees "in advance."

4.1. Compatibility and coalescence

We will use the following definitions. A *split* on a taxon set X is a bipartition of X. Because the set X will be clear, we will often abuse notation by identifying $A \subseteq X$ with the partition $A|(X \backslash A)$. Furthermore, because we have a special element ρ, we can distinguish between the two sides of a split; the side not containing ρ will be called the *rsplit* (short for rooted split) of the split. It is clearly equivalent to describe a given partition in terms of a split or an rsplit, and we will use the two descriptions interchangeably.

Note that the neighbor-joining algorithm is typically thought of as proceeding by coalescing internal nodes of an unresolved phylogenetic tree (Fig. 4); however, for our purposes, it will sometimes be easier to consider the forest obtained by deleting the central node and the associated edges. The opposite construction will be called "starification."

Definition 3. *Given a forest F, define the* starification $\bigstar(F)$ *of F as the following unrooted tree. If F has one tree, then suppress the degree two root node of F. If F has two trees, then join their root nodes by an edge. If F has three or more trees, join all of the root nodes of trees of F to a single node. The new introduced node will be called the* star node.

We will identify any one, two, or three tree forest F with its starification.

Definition 4. *Given a tree T which is part of a forest F on a taxon set X, define the* edge splits $\Sigma_E(T)$ *to be* $\mathcal{L}(T)|[\mathcal{L}(T)]^c$ *along with the set of splits on X induced by the edges of T. We define* $\Sigma_E(F)$ *to be the union of the edge splits of T across all trees T in F.*

For example, the rsplits $\{3\}$ and $\{2, 3, 4\}$ are both edge rsplits of the forest $((1, \rho), 2);(3, 4)$.

Definition 5. *Given a forest F on a taxon set X, A is a* separating split *of F if A is the union of taxon sets for a collection of at least two trees of F. The set of separating splits of F will be denoted* $\Sigma_S(F)$.

Given a forest F, we will write $\Sigma(F)$ for $\Sigma_E(F) \cup \Sigma_S(F)$. This will be the set of splits used to make agreement partitions, as described below.

Definition 6. *Two rsplits A and B will be called* compatible *if either* $A \cap B = \emptyset$, $A \subseteq B$, or $B \subseteq A$.

Because A and B are the sides of the splits that do not contain ρ, this is the same as the usual criterion for split compatibility (Semple and Steel, 2003). Therefore, we have the following well-known theorem.

Theorem 7 (Buneman, 1971). *A collection of splits M on a taxon set X is pairwise compatible iff there exists an unrooted tree T on taxa X such that M is a subset of* $\Sigma_E(T)$. *There is a one-to-one correspondence between compatible sets of splits on X and minimally-resolved unrooted trees on X.*

Definition 8. *Two forests F and G on taxon set X are* compatible *if* $\Sigma_E(F)$ *and* $\Sigma_E(G)$ *are pairwise compatible.*

Definition 9. *The* join $T \wedge S$ *of two trees T and S on disjoint taxon sets is the tree obtained by joining the root nodes of T and S to a new root node. The* coalescence *of T and S in the forest F is the forest* $\{T \wedge S\} \cup (F \setminus \{T, S\})\}$.

Note that the operation of coalescence gives a partial order on the set of forests on a given taxon set. Namely, we write $F \succeq G$ if F is a coalescence of G. Clearly, trees are the maximal elements in this partial order.

Definition 10. *A tree S is a* subtree *of an unrooted tree U if S is one component of the disconnected graph obtained by cutting an edge of U. A tree S is a* subtree *of a rooted tree T if S is a component of the disconnected graph obtained by cutting an edge of T, and S does not include the root of T.*

We emphasize that the subtree definition is different than that of an *induced subtree*, which is as follows. The existence of induced subtrees is guaranteed by Theorem 7 or its rooted equivalent.

Definition 11. *Given a tree T and $Y \subseteq \mathcal{L}(T), T|_Y$ is the (rooted or unrooted) tree on taxa Y with rsplits $\{A \cap Y : A \in \Sigma_E(T)\}$.*

There is also an analogous definition for forests.

Definition 12. *Given a forest F and $Y \subseteq \mathcal{L}(F)$, $F|_Y$ is*

$$\{T|_Y : T \in F \text{ and } L(T) \cap Y \neq \emptyset\}.$$

Proposition 13. *If two forests F and G on a taxon set X are compatible, and F has more than one tree, then there exists $H \succ F$ such that H is compatible with G.*

Proof. If F has two or three trees, the proposition is trivial. Otherwise, let U be the tree with split set equal to the union of $\Sigma_E(F)$ and $\Sigma_E(G)$. If U is not resolved (i.e., if there exists an internal node of degree greater than three), then take an arbitrary resolution. As all of the trees T of F are resolved, each T sits as a subtree of U; let J be the union of the nodes of the $T \in F$ (considered as nodes of U). Let p denote the longest path in U which does not contact any of the nodes in J. Because F has at least four trees, p will be nontrivial. Pick one end of this path, which must be connected to a pair of trees S', S'' of F. Let $K = \mathcal{L}(S') \cup \mathcal{L}(S'')$. As the split $K|K^c$ is already a split of U, we know that it is compatible with $\Sigma_E(F)$ and $\Sigma_E(G)$, and thus that $\Sigma(S' \wedge S'')$ is compatible with $\Sigma_E(G)$. Let H be the coalescence of S' and S'' in F. ■

4.2. Maximum agreement partitions

In this section, we introduce the notion of maximum agreement partition (MAP), which generalizes the idea of maximum agreement forests. Maximum agreement forests were first introduced by Hein et al. (1996) and further refined by Bordewich and Semple (2005). In broad terms, given two forests F and G on a taxon set X, we will be interested in considering partitions P which are obtainable from F and G independently by "combining" edge splits and separating splits of those forests, in the same way that edge cuts are combined when making maximum agreement forests. The appropriate notion of "combining" splits is the infimum, which we now describe.

The set of partitions on a given finite set Y form a partial order, such that a partition $P_1 \leq P_2$ if P_1 is a refinement of P_2. In fact, the set of partitions is a *complete lattice*, meaning that any set of partitions on Y has a supremum and an infimum. For a collection of partitions M, we will use $\inf(M)$ to denote their infimum.

Thus, as described below, a necessary condition for P to be an agreement partition for two forests F and G is that P can be expressed as $\inf(M)$ and $\inf(N)$ for $M \subseteq \Sigma(F)$ and $N \subseteq \Sigma(G)$. It will now be useful to connect that definition to one in terms of convexity of characters (Semple and Steel, 2003).

Definition 14. *Given a partition P on some set K, define $P|_J$ for some $J \subseteq K$ to be the partition $\{Y \cap J : Y \in P\}$.*

The following is a slight generalization of the definition of convexity given by Semple and Steel (2003).

Definition 15. *A partition P on a taxon set X is convex on a forest F on X if there exists an $H \succeq F$ such that P induces a convex character on $\bigstar(H)$, i.e. if there exists a partition $\tilde{P}$ on vertices $\mathcal{V}(\bigstar(H))$ such that*

(i) $P = \tilde{P}|_X$.

(ii) Any $\tilde{Y} \in \tilde{P}$ separates $\bigstar(H)$ into connected components.

The following proposition relates the notions of "obtainable by a series of cuts along edge or separating splits" with the notion of character convexity.

Proposition 16. *A partition P of a taxon set X is convex on a forest F iff there exists* $M \subseteq \Sigma(F)$ such that $P = \inf(M)$.

Proof. Assume $M \subseteq \Sigma(F)$ such that $P = \inf(M)$. Note that $K = \inf\ (M \cap \Sigma_S(F))$ is a set of disjoint separating or root-edge splits for F; thus, we can perform coalescences, making H, such that the splits from sets in K are edge splits of H. Such an H will satisfy the criteria of the definition.

For the converse implication, cutting any edge (u, v) of $\bigstar(H)$ for any $H \succeq F$ gives a split in $\Sigma(F)$. We then define M as the set of such splits $s_{u,v}$ such that (u, v) is an edge, and u and v are in distinct sets of the partition $\tilde{P}$. By construction, $P = \inf(M)$. ■

The following definition generalizes the notion of agreement forest.

Definition 17. *We say that a partition P of taxon set X is an* agreement partition *for a pair of forests F, G on X if*

(i) for every pair of rsplits $A \in \Sigma_E(F), B \in \Sigma_E(G)$, *and* $Y \in P$, $A \cap Y$ *is compatible with* $B \cap Y$.

(ii) P is convex on F and G.

We say that P is a maximum agreement partition (MAP) if the number of sets of P is less than or equal to that of any other agreement partition. Let $m(F, G)$ *be the number of sets in the MAP minus one.*

Note that by Theorem 7, for two resolved unrooted trees U, V on a taxon set $X \ni \rho$, the size of the maximum agreement partition is the same as the size of the maximum agreement forest of the trees (rooted at ρ) in the sense of Bordewich and Semple (2005). Recall that the definitions of maximum agreement forest in Bordewich and Semple (2005) differs from that of Hein et al. (1996) and Allen and Steel (2001).

Proposition 18. *Assume F, G, and H are forests on a taxon set X such that* $H \succeq F$, *and P is an agreement partition for H and G. Then P is also an agreement partition for F and G.*

Proof. Part (i) of the definition is clear as $\Sigma_E(F) \subseteq \Sigma_E(H)$. Next we check (ii), i.e., that P is convex on F. Note that $H \succeq F$ implies $\Sigma(H) \subseteq \Sigma(F)$, as the "extra" edge splits of H will be separating splits of F. By Proposition 16, there exists an $M \subseteq \Sigma(H)$ such that $P = \inf(M)$; by the previous sentence, $M \subseteq \Sigma(F)$, and so by Proposition 16 again P is convex on F. ■

The following theorem is the main motivation for studying the maximum agreement partition. Thus, the proposition says that the size of the maximum agreement partition of the two forests F and G is the same as the rSPR distance in the best case.

Theorem 19. *The minimum of* $d_{\mathrm{rSPR}}(U, V)$ *across all unrooted trees* $U \succeq F$ *and* $V \succeq G$ *is equal to* $m(F, G)$.

The proof of this proposition will come after two lemmas.

Lemma 20. *Given a partition P convex on F and* $Y \in P$ *such that* $F|_Y$ *includes two distinct trees T and S, then there exist distinct trees* $\tilde{T}$, $\tilde{S} \in F$ *such that* $(\tilde{T} \wedge \tilde{S})|_Y = T \wedge S$. *Furthermore, for any* $Z \in P$ *not equal to Y and any* $R \in \{\tilde{T}, \tilde{S}\}$, *we have either* $Z \subset \mathcal{L}(R)$ *or* $Z \cap \mathcal{L}(R) = \emptyset$.

Proof. Let H and $\tilde{P}$ be as in Definition 15. Let $\tilde{Y} \in \tilde{P}$ be such that $\tilde{Y} \cap \mathcal{L}(F) = Y$. Let $\tilde{T}$ (resp. $\tilde{S} \in F$) be the tree such that $\tilde{T}|_Y = T$ (resp. $\tilde{S}|_Y = S$). Let $Q = \mathcal{L}(T) \cup \mathcal{L}(S)$.

We now show that $\tilde{T} \neq \tilde{S}$. The contrary would imply $\bigstar(H)|_Q = \tilde{T}|_Q$. Because $Q \subset Y$ and $\bigstar(H)|_Y$ is connected by definition, $\tilde{T}|_Q$ is connected so T and S would not be distinct. This is a contradiction. It follows that $\mathcal{L}(\tilde{T}) \cap \mathcal{L}(\tilde{S}) = \emptyset$ so $(\tilde{T} \wedge \tilde{S})|_Y = \tilde{T}|_Y \wedge \tilde{S}|_Y = T \wedge S$.

We now show the second statement of the lemma. Let $r(W)$ denote the root node of any tree $W \in F$. Note that $r(\tilde{T})$ and $r(\tilde{S})$ must be in $\tilde{Y}$ because $\bigstar(H)|_Q$ is connected and the path between any $a \in \mathcal{L}(T)$ and $b \in \mathcal{L}(S)$ passes through $r(\tilde{T})$ and $r(\tilde{S})$.

Now assume that for some $R \in \{\tilde{T}, \tilde{S}\}$ we have that some $Z \neq Y$ of P intersects $\mathcal{L}(R)$ but is not contained in it. Take $c \in Z \cap \mathcal{L}(R)$ and $d \in Z \cap [\mathcal{L}(R)]^c$. Let $\tilde{Z} \in P$ be such that $\tilde{Z} \cap L(F) = Z$. By the same argument as in the previous paragraph, $r(R)$ is in $\tilde{Z}$. This is a contradiction as $\tilde{Y}$ and $\tilde{Z}$ are disjoint. ■

Lemma 21. *Assume that F and G are forests on a taxon set X, and P is an agreement partition for F and G. Then there exist resolved trees $U \succeq F$ and $V \succeq G$ such that P is an agreement partition for U and V.*

Proof. It is enough to show that if one of the forests, say F, has at least four trees, then there exists an $H_0 \succ F$ such that P is an agreement partition for H_0 and G.

If for every $Y \in P$ we have that $F|_Y$ is a single tree, then we can make H_0 by taking an arbitrary coalescence of F; any such coalescence will be "broken" by P and thus will not introduce any splits violating (i) of Definition 17. Thus, we assume that $F|_Y$ has at least two trees. By Proposition 13, there exist nontrivial $T, S \in F|_Y$ such that the coalescence of T and S in $F|_Y$ is compatible with $G|_Y$.

By Lemma 20, there exist $\tilde{T}$ and $\tilde{S}$ in F such that $[\tilde{T} \wedge \tilde{S}]|_Y = T \wedge S$. Let H_0 be the coalescence of $\tilde{T}$ and $\tilde{S}$ in F; the second statement of Lemma 20 implies that the coalescence of $\tilde{T}$ and $\tilde{S}$ does not introduce any new edge splits when restricted any $Z \neq Y$ in P, and so H_0 satisfies the criterion (i) of a maximum agreement partition.

Also, P is convex on H_0, establishing criterion (ii). Indeed, by Proposition 16, let $M \subseteq \Sigma(F)$ be such that $P = \inf(M)$; we need to show that $M \subseteq \Sigma(H_0)$. The only difference between $\Sigma(F)$ and $\Sigma(H_0)$ is that $\Sigma(H_0)$ does not have separating partitions which separate $\tilde{T}$ and $\tilde{S}$, but M cannot contain such a partition because T and S both have taxa in the same partition of P. ■

Proof of Theorem 19. Lemma 21 shows that the minimum of $m(U, V)$ is less than or equal to $m(F, G)$. The other inequality follows from Proposition 18.

Now note that for a resolved tree on X rooted at ρ, the notions of maximum agreement forest and MAP coincide. Thus, by Theorem 2.1 of Bordewich and Semple (2005), $m(U, V)$ is equal to the rSPR distance between U and V for any resolved $U \succeq F$ and $V \succeq G$. ■

4.3. Calculating the maximum agreement partition

As introduced above, and described more clearly below, `constNJ` needs to find a great number of agreement partitions. Indeed, a sample `constNJ` run with three distance matrices, 27 taxa, with pairwise constraints of size two required 5867 calls to the subroutine finding the size of a MAP. Therefore, a speedy calculation of the MAP is essential.

In the present implementation of `constNJ`, the MAP is calculated via a simple extension of the algorithm by Bordewich and Semple (2005). As with the usual Bordewich-Semple algorithm, we contract isomorphic subtrees and replace chains of pendant subtrees with chains of three pendant edges. However, we consider separating rsplits as well as edge rsplits to find the agreement partition.

An alternative would be to consider an integer linear programming (ILP) approach to the MAP problem based on the work of Wu (2008), who has recently developed an ILP approach to finding a maximum agreement forest. Although Wu's ILP approach is many orders of magnitude faster than the Bordewich-Semple algorithm for finding the size of the maximum agreement forest in the "hard" case when two trees are quite different, our tests have shown that it is slower in the "easy" case. This difference is probably because there is overhead to creating the linear programming matrix, which does not scale strongly with respect to the difficulty of the problem, while the Bordewich-Semple algorithm is very fast for easy problems. It is possible that some of the ILP overhead could be amortized by clever re-use of portions of the matrix across coalescences, or a combination of Bordewich-Semple and Wu ideas, but we have not followed these directions.

5. THE `constNJ` ALGORITHM

Assume `constNJ` is given k distance matrices on a taxon set X. On the way to constructing our trees $T_1, \ldots, T_k$ on X, we will be constructing collections of forests $\mathbf{F} = F_1, \ldots, F_k$; we will call such a collection $\mathbf{F}$ an "instance." For example, each boxed pair of trees in Figure 4 is an instance (after deleting the central "star" nodes). The *agreement profile* for an instance $\mathbf{F}$ is the $k \times k$ matrix $\alpha(\mathbf{F})$, where $\alpha(\mathbf{F})_{ij}$ is $m(F_i, F_j)$. It describes the degree to which the forests agree. The *identical agreement profile* is the $k \times k$ zero matrix. Define the *instance tensor* to be a partially filled tensor of instances indexed by $\mathbb{N}^{k^2}$, where $\mathbf{F}$ is stored in the "slot" indexed by its agreement profile $\alpha(\mathbf{F})$.

Algorithm 22 (`constNJ`). *Given $n\times n$ distance matrices $D_1, \ldots, D_k$ and a $k\times k$ constraint matrix C,*

```
1. Let $\mathbf{F}^{(0)}$ be the trivial instance, i.e., $F_i^{(0)}$ is the trivial forest on n taxa for each $1 \le i \le k$. Let $\mathbf{H}^{(0)}$ be the instance
   tensor containing only $\mathbf{F}^{(0)}$.
2. Repeat the following until termination:
   a. Let $\mathbf{H}$ be the instance tensor from the previous step.
   b. Rank all possible coalescences of all of the instances of $\mathbf{H}$ by how much they will decrease total tree length.
   c. Make a "step" by walking down this ranked list in order as follows:
      i. Perform the next coalescence, say of an instance $\mathbf{F}$, and assume that the resulting instance $\mathbf{F}'$ has
         agreement profile X.
      ii. If some entry of X is greater than the corresponding element of C, discard $\mathbf{F}'$ and test the next coalescence.
      iii. If not, and $\mathbf{F}'$ is the first in this step to have agreement profile X, then save it. If, on the other hand,
         another instance has already been found in this step with agreement profile X, then discard $\mathbf{F}'$ as it must
         have a larger total tree length.
      iv. Stop walking down the list if X is the identical agreement profile.
   d. Terminate if each of the $F_i$ have three trees or fewer.
```

We now show that this algorithm is consistent.

Proof of Theorem 2. Algorithm 22 is consistent because of the consistency of neighbor-joining (Gascuel, 1997; Bryant, 2005) and because the coalescence which most decreases the total tree length must be a neighbor-joining step (Desper and Gascuel, 2005). We are given a sequence of distance matrices $D_1, \ldots, D_k$ and a symmetric $k\times k$ constraint matrix C. By hypothesis, these distance matrices come from a sequence of trees $T_1, \ldots, T_k$ such that the rSPR distance between T_i and T_j is bounded above by $C_{i,j}$. First, by the consistency of neighbor-joining, NJ applied to each distance matrix independently will recover the correct collection of trees. Say the sequence of neighbor-joining coalescences making T_i gives a series of forests $F_{1,i}, \ldots, F_{n-1,i}$, where $F_{n-1,i} = T_i$. Thus, by Theorem 19 (more specifically, Proposition 18) and our assumptions about the T_i,

$$m(F_{a,i}, F_{b,j}) \le C_{i,j} \tag{3}$$

for any $1 \le a$, $b \le k$ and $1 \le i, j \le n$. Thus, the constraints will always be satisfied as long as we follow the sequence of NJ steps for each tree.

Next we show by induction that given this data, at every step every `constNJ` forest will one of the $F_{r,j}$ for $1 \le r < n$ and $1 \le j \le n-1$. This is clearly true at initialization. By induction, assume the assertion is true at some `constNJ` step. Consider the coalescence which decreases total tree length as much as possible irrespective of constraints; say it occurs in $F_{k,i}$. As the coalescence decreases, the tree length of $F_{k,i}$ compared to other coalescences of $F_{k,i}$ is also a neighbor-joining step for $F_{k,i}$, making $F_{k+1,i}$. By the previous paragraph, we know that this coalescence will preserve the constraints, and thus is also a `constNJ` step (recall that each `constNJ` step decreases the total tree length as much as possible amongst coalescences which preserve the constraints). Thus, at the end we get $F_{n-1,i}$ for each i by induction. Because $F_{n-1,i} = T_i$, `constNJ` is a consistent algorithm. ■

5.1. Implementation

We have implemented `constNJ` in the fast functional/imperative language `ocaml` (Leroy et al., 2007). The implementation has a simple command line interface, which is documented in the accompanying manual. It is available for download at http://github.com/matsen/constnj/.

As described above, the primary input for `constNJ` is a series of distance matrices, with one for each alignment block. The program is designed to accept distance matrices from the DNADIST program of the PHYLIP package, although longer lines and taxon names are allowed. The first taxon is assumed to be the outgroup. The program assumes that the taxa in the distance matrices are ordered in a corresponding way. For instance, if one is using `constNJ` to investigate recombination, all of the taxa should be listed in the same order, so that the taxa in the no-recombination blocks correspond to one another. On the other hand, if one is using `constNJ` to investigate host-parasite relationships, the ith taxon in the parasite alignment should parasitize the ith taxon in the host alignment. If, for example, a given parasite is present in multiple hosts, this will require duplication of that parasite sequence in the alignment.

The second input for `constNJ` is a set of constraints for the resulting correlated set of trees. There are two options for specifying these constraints: first, via a file, or second, by enforcing "linear" constraints. For example, assume we supply three distance matrices: D_0, D_1, and D_2, and would like to construct trees T_0, T_1, and T_2. To specify constraints for these matrices, one writes one constraint per line, with first the indices of the distance matrices, then the number of rSPR moves allowed between those distance matrices. For example, a line saying 0 2 1 would mean that T_0 and T_2 are constrained to be one rSPR move apart. On the other hand, one may specify a linear constraint with a linear constraint parameter. If the linear constraint parameter is L, then trees T_i and T_j are constrained to be $L \cdot |i-j|$ rSPR moves apart. So if we apply a linear constraint with parameter 2 in our example, then both T_0 and T_1 and T_1 and T_2 are constrained to be at most 2 rSPR moves apart, while T_0 and T_2 are constrained to be at most 4 rSPR moves apart.

The output for `constNJ` is collection of correlated sets of trees, each of which get their own `.tre` file, along with a `.lengths` file, which describes the total tree length for each of these sets of trees. `constNJ` returns at most one correlated set of trees for each agreement profile within the constraints, which is labeled by the agreement profile. If the constraints are given in a file, then the agreement profile is written in the order given in the file. If linear constraints are given, the agreement profile is written as a vector representing an upper triangular matrix in the usual way. For example, the agreement profile for three trees with linear constraints is written ($d_{\text{rSPR}}(T_0, T_1)$, $d_{\text{rSPR}}(T_0, T_2)$, $d_{r\text{SPR}}(T_1, T_2)$), so the set of trees in the file example.2_1_1`.tre` has agreement profile (2,1,1). The `.lengths` file contains the information on tree lengths, as in Table 1. Namely, for each correlated set of trees returned by `constNJ`, it displays the total tree length for those trees.

5.2. *Speed*

A rigorous worst-case runtime analysis of `constNJ` would show that it can be incredibly slow. Indeed, the MAP is a generalization of the maximum agreement forest; thus, finding the size of the MAP is NP-hard by the corresponding theorem by Bordewich and Semple (2005). However, `constNJ` does not just need to solve one such problem; it needs to solve quite a number of them. At worst, `constNJ` would need to find as many MAPs as there are possible coalescences, just for a single step and a single instance; if an instance had forests with $\ell_1, \ldots, \ell_k$ trees, then there will be $\binom{\ell_1}{2} \times \cdots \times \binom{\ell_k}{2}$ possible coalescences, each of which in theory could require solving of a MAP problem. At any step, there can be as many instances as there are agreement profiles satisfying the constraint matrices, and a problem with n taxa and k distance matrices will require nk such steps. Such an analysis would not give a very clear understanding of the practical time requirements of running `constNJ`.

In practice, `constNJ` can be used effectively for a moderate number of taxa and a small number of closely constrained trees. The running time depends somewhat on the number of taxa, but quite a lot on the constraints and number of distance matrices. Indeed, the main bottleneck is the MAP calculation, and the running time of the MAP calculation depends very strongly on the constraints and the number of distance matrices.

However, what may be surprising is how much the running time depends on the quality of the data. This is vividly illustrated by the simulations, where in the case of two trees with two reticulation events and divergence of 0.1 mutation per site per tree, the sequences with 100 sites took on average 10.3 minutes to run, while the simulations with 6400 sites took on average 0.68 seconds each. This represents a difference of almost three orders of magnitude. On the same processor (Intel® Xeon® CPU at 2.33GHz) using real HIV data, an example with three 38-taxon distance matrices and pairwise constraints of three for each pair of distance matrices took 49 seconds, while an example with only two 40-taxon distance matrices with a single constraint of size three took almost 21 minutes. The quality of the data impacts "how far" `constNJ` has to go down the list of coalescences in order to find one with the desired agreement profile and how often it needs to calculate a new agreement partition.

We have made some coding choices to increase the speed. For example, there is a natural partial order on agreement profiles, which is just the element-wise numerical order. In considering which coalescences to perform, we only investigate those coalescences which could lead to an agreement profile that is smaller than those which have already been performed. In principle, one could do a more comprehensive search which might lead to more optimal sets of trees; we have not found a significant improvement following such a direction.

6. SIMULATIONS

In order to evaluate the performance of `constNJ`, we performed a number of simulations. The trees in the study were generated as follows. We choose the number of trees in the recombination network, say k, the size of the trees, say n, and a number of rSPR moves, say m. We start with a tree T_1 drawn from the Yule distribution of trees on n taxa. After choosing the desired expected number of substitutions on the tree (in simulations below, 0.1, 0.5, and 2), we divided this number by the number of non-root edges in the tree to get the expected number of substitutions per edge. We then drew the actual number of substitutions per edge from the exponential distribution with the corresponding mean to get the branch lengths of T_1.

We then generated T_{i+1} from T_i by applying k rSPR moves to T_i as follows. For each rSPR move, first select a non-root edge uniformly; call the subtree below the chosen edge S. Cut off S, then glue it back in on a uniformly selected edge of T_1 not contained in S. The location along the chosen edge to attach S is chosen uniformly. Then to simulate differential rates of evolution of different regions, take the average of the previous branch length and a branch length drawn from an exponential distribution as before.

Given such a series of trees $T_1, \ldots, T_k$, we generated a collection of distance matrices $D_1, \ldots, D_k$ by simulating sequences on the trees. We did so using the Jukes-Cantor model of sequence evolution with a single rate. Distances were then calculated using the Jukes-Cantor distance correction (Felsenstein, 2004a). In case the Jukes-Cantor correction gave an undefined value, we repeated the analysis with a new sequence. We chose the simple Jukes-Cantor model to focus attention on our method rather than the distance estimator.

For the first set of simulations, we wanted to understand how the topological accuracy of `constNJ` compares to that of concatenating alignment blocks or running them independently. For concatenation, we estimated a single distance for each pair of taxa by taking the Jukes-Cantor correction of the average number of substitutions in each alignment block; such a procedure simulates the process of concatenating equal-length alignment blocks. We then considered the resulting tree as the output of running NJ on the concatenated alignment for each alignment block. For independent construction, we simply ran NJ on each distance matrix independently. For `constNJ`, we constrained the rSPR distance between the trees to be less than or equal to the number of rSPR moves used to generate the trees. The trees used in the comparison were then the shortest (i.e., smallest total tree length) trees returned given those constraints.

To measure topological accuracy, we used the Robinson-Foulds distance (Robinson and Foulds, 1981), which is simply one half the size of the symmetric difference of the edge split sets. The RF distance for two collections of trees is defined to be the total RF distance among the corresponding pairs of trees. The results are shown in Figures 5–7. In these simulations, `constNJ` typically outperforms either alternate strategy. When sequences are short, the main source of error is insufficiently accurate distance estimations; concatenation increases the amount of useful sequence information for distance estimation, and so outperforms independent construction in that case. However, constNJ does almost as well. On the other hand, when

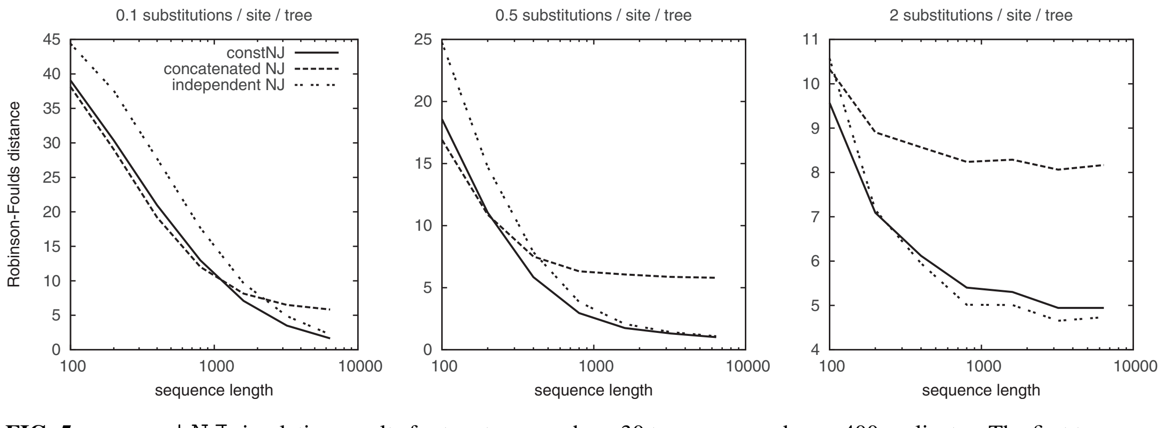

FIG. 5. `constNJ` simulation results for two trees, each on 30 taxa, averaged over 400 replicates. The first tree was drawn from the Yule distribution, and the second tree was made by applying a random rooted subtree-prune-regraft (rSPR) move to the first. "`constNJ`" is our algorithm, "concatenated NJ" is neighbor-joining run with a concatenated alignment, and "independent NJ" is neighbor-joining run independently on the alignments for the different trees as described in the text.

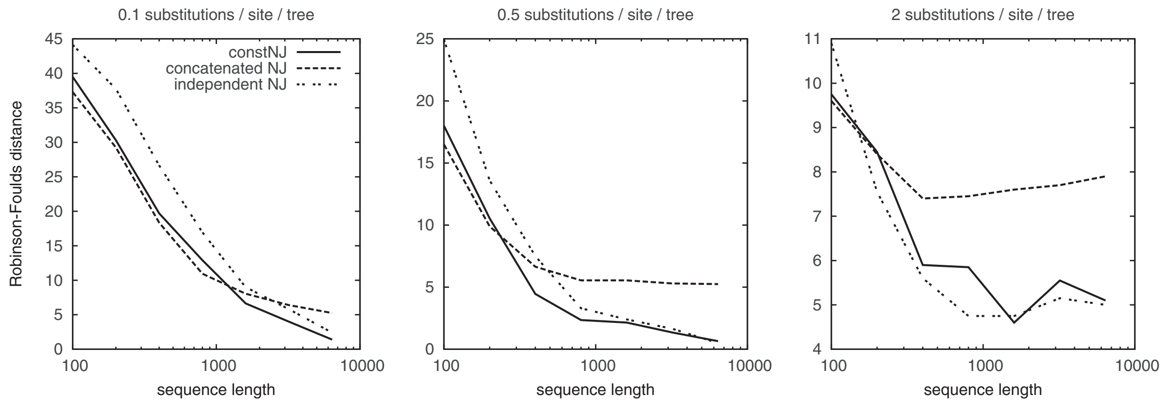

FIG. 6. constNJ simulation results for two trees, each on 30 taxa, averaged over 400 replicates. This time, two rooted subtree-prune-regraft (rSPR) moves were applied to the first tree to get the second.

sequences are long, independent estimation does well, as there is enough sequence information to reconstruct the tree for each block independently. In that case, constNJ also does well.

The reader may object that these graphs represent an unfair comparison, as they assume that the number of reticulation events is correctly bounded in advance. The next two simulations address this objection. The first set, with results shown in Figure 8, seems to indicate that by looking at the .lengths file one can do a reasonable job of deciding how many rSPR moves to allow, as was done in the example case of the introduction. The second set, with results shown in Figure 9, concerns what happens if one makes an incorrect decision.

Figure 8 explores one of the main themes of this article, which is the trade-off between phylogenetic optimality (in this case, total tree length) and congruence among individual trees. To make Figure 8, we generated pairs of trees as before, generating a Yule tree and then applying some number of rSPR moves to get the second tree, except that this time we threw out pairs of trees that did not have the correct rSPR distance between them (i.e., when a subtree was moved back to its original location). We drew branch lengths as above, then simulated 1000 sites with an expectation of 0.5 mutations per site per tree. The *x*-axis is the index of the .lengths file, i.e., the number of rSPR moves between the two reconstructed trees. The *y*-axis, "total tree length," shows the total length of trees with that number of rSPR moves between them. For instance, consider the point on the line labeled "three rSPR moves," which is at *x*-value 2. This says that if we simulate a pair of trees that are three rSPR moves apart as described above, then we expect the pair of trees output by constNJ with agreement profile two to have total length of about 1.038. Note that constNJ does not always return a tree for every agreement profile that is allowed under the constraints. In those cases, we simply took the total tree length from the largest non-empty agreement profile.

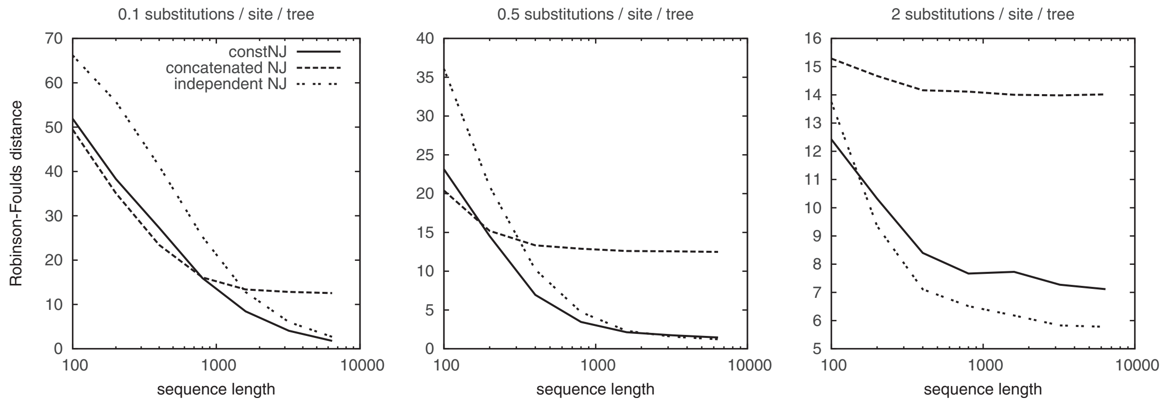

FIG. 7. constNJ simulation results for three trees, each on 30 taxa, averaged over 400 replicates. Here one random rooted subtree-prune-regraft (rSPR) move was done to change the first tree to the second tree, and the second tree to the third.

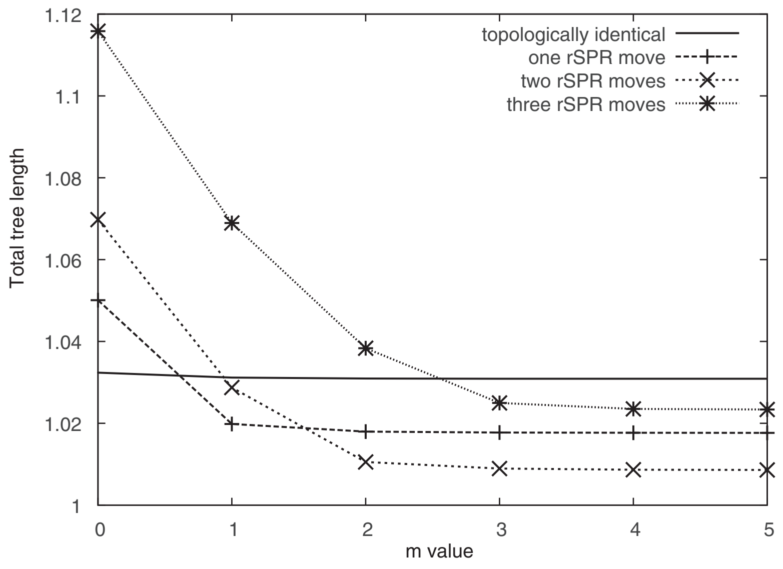

FIG. 8. Comparison of the total tree lengths for simulated trees differing by the described number of moves and then reconstructed using `constNJ;` average of 100 simulations. As can be seen, the most significant decreases in the total tree length happen when getting to the correct number of rooted subtree-prune-regraft (rSPR) moves, after which the plot levels off. For example, on the line "two rSPR moves," there are significant decreases in length when going to one and two rSPR moves, but not much decrease after that. Thus, at least in simulation, it appears possible to make a reasonable choice concerning the number of rSPR moves to allow between the two trees.

Figure 8 shows exactly what one might expect. Namely, if we generate pairs of identical trees, then not much improvement in terms of total tree length is gained by allowing the trees to differ. However, if the trees are one rSPR move apart, then there is a substantial drop when allowing one rSPR move, but not much more after that; this indicates that only one rSPR move is called for by the data. The situation is similar for the other numbers of rSPR moves. Thus, at least in simulation with good quality data, it appears that one

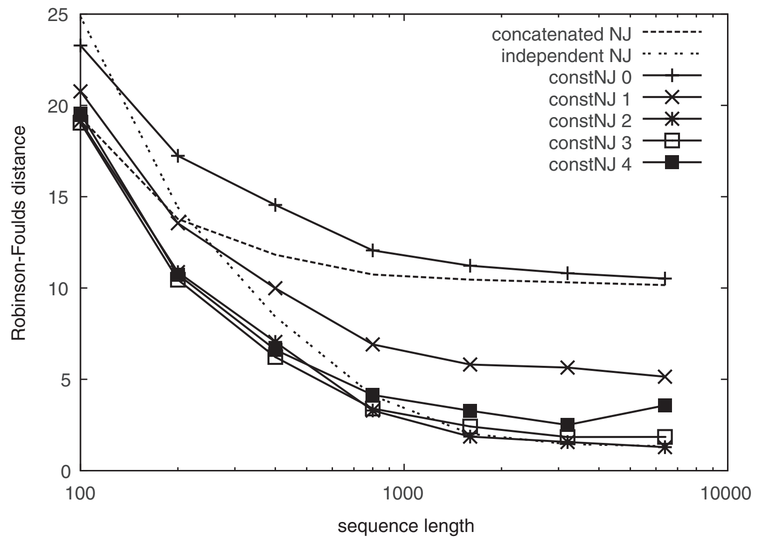

FIG. 9. Comparison of various specified constraints for `constNJ;` average of 100 simulations. Data was simulated on two trees, each on 30 taxa, such that two random rooted subtree-prune-regraft (rSPR) moves were done to change the first tree to the second tree. Then reconstruction was done with rSPR distance constraints of 0, 1, 2, 3, and 4. As would be expected, having a constraint of 0 (identical trees) has qualitative performance similar to that of concatenated neighbor-joining (NJ), the best performance is obtained by the correct constraint of 2 moves, and a constraint of 1 gives results between those for 0 and 2. The performance of 3 is similar to that for 2, while 4's performance degrades with accurate distances.

should be able to make a reasonable judgment as to the correct number of rSPR moves for the data set at hand, as was done in the introduction.

We also performed some simulations allowing an incorrect number of rSPR moves (Fig. 9). As shown, giving a too-small constraint interpolates between results from concatenated data and the correct specification, while too-large constraints give performance similar to the correct constraint. One might expect `constNJ` with too-large constraints to give results similar to independent NJ; we do not have a clear explanation why this is not the case.

7. CONCLUSION

In this article, we present `constNJ`, a consistent distance-based algorithm for a collection of trees with pairwise rSPR constraints, such as those constraints satisfied by collections of trees that fit into a reticulation network. `constNJ` is deterministic and a strict generalization of the neighbor-joining algorithm. In order to ensure that the resulting set of trees satisfy the specified constraints on rSPR distance, we develop the theory of maximum agreement partitions, culminating in Theorem 2. We simplify the problem considerably by assuming that the alignment blocks are known in advance; in doing so, we preserve the correlation between sites in the alignment with the same history. Rather than first finding trees and then attempting to put them into a recombination network, we investigate the balance between discordance between trees and optimality of the ensemble of trees. By using trees whose rooting is derived from outgroups, we find explicit evolutionary histories.

Many improvements are possible. First, we enforce pairwise bounds on the rSPR distance between trees, which is a relatively weak way to show that these trees fit into a network. A more explicit approach would be desirable. Second, an alternative direction for heuristic optimization might be to do a more complete search for the minimum length tree in a manner analogous to algorithms searching for the BME tree. Third, `constNJ` does not reconstruct branch lengths. It would be possible to do a distance-based branch length estimation in a manner similar to that for usual neighbor-joining, but the fact that we are choosing trees which may be sub-optimal according to the NJ criterion implies that negative branch lengths might be encountered. Such reconstruction is probably best done in the context of a complete likelihood-based approach.

Ten years ago, Kuhner et al. (2000) wrote "[w]hen recombination occurs, adjacent sites may have different, although correlated, genealogical histories. Reconstructing these genealogies with certainty is impossible." Although we do not claim certainty for this (or any forthcoming) algorithm attempting to reconstruct reticulate phylogenetic history, we think that there is cause for optimism and look forward to seeing future developments in this area.

ACKNOWLEDGMENTS

I would like to thank Lior Pachter for a number of helpful early discussions. Contact with the Mullins HIV lab at the University of Washington led indirectly to the present work. The Armbrust Oceanography lab at the University of Washington generously allowed me to use their computing cluster. I made use of the following useful software: Figtree (Rambaut, 2008), PHYML (Guindon and Gascuel, 2003), and PHYLIP (Felsenstein, 2004b).

DISCLOSURE STATEMENT

No competing financial interests exist.

REFERENCES

Allen, B., and Steel, M. 2001. Subtree transfer operations and their induced metrics on evolutionary trees. *Ann. Combinat.* 5, 1–15.

Ané, C., Larget, B., Baum, D., et al. 2007. Bayesian estimation of concordance among gene trees. *Mol. Biol. Evol.* 24, 412.

Bandelt, H., and Dress, A. 1992a. A canonical decomposition theory for metrics on a finite set. *Adv. Math.* 92, 47–105.

Bandelt, H., and Dress, A. 1992b. Split decomposition: a new and useful approach to phylogenetic analysis of distance data. *Mol. Phylogenet. Evol.* 1, 242–52.

Baroni, M., Semple, C., and Steel, M. 2005. A framework for representing reticulate evolution. *Ann. Combinat.* 8, 391–408.

Baroni, M., Semple, C., and Steel, M. 2006. Hybrids in real time. *Syst. Biol.* 55, 46–56.

Bloomquist, E., and Suchard, M., 2010. Unifying Vertical and Nonvertical Evolution: A Stochastic ARG-based Framework. *Syst. Biol*. 59, 27–41.

Bordewich, M., and Semple, C. 2005. On the computational complexity of the rooted subtree prune and regraft distance. *Ann. Combinat.* 8, 409–423.

Bordewich, M., and Semple, C. 2007a. Computing the hybridization number of two phylogenetic trees is fixed-parameter tractable. *IEEE TCBB* 4, 458–466.

Bordewich, M., and Semple, C. 2007b. Computing the minimum number of hybridization events for a consistent evolutionary history. *Discr. Appl. Math.* 155, 914–928.

Bryant, D. 2005. On the uniqueness of the selection criterion in neighbor-joining. *J. Classif.* 22, 3–15.

Bryant, D., and Moulton, V. 2004. Neighbor-net: an agglomerative method for the construction of phylogenetic networks. *Mol. Biol. Evol.* 21, 255–265.

de Oliveira Martins, L., Leal, E., and Kishino, H., 2008. Phylogenetic detection of recombination with a Bayesian prior on the distance between trees. *PLoS ONE* 3.

Desper, R., and Gascuel, O. 2004. Theoretical foundation of the balanced minimum evolution method of phylogenetic inference and its relationship to weighted least-squares tree fitting. *Mol. Biol. Evol.* 21, 587–598.

Desper, R., and Gascuel, O. 2005. The minimum evolution distance-based approach to phylogenetic inference, 1–32. In Gascuel, O., ed., *Mathematics of Evolution and Phylogeny*. Oxford University Press, Oxford, UK.

Edwards, S., Liu, L., and Pearl, D. 2007. High-resolution species trees without concatenation. *Proc. Natl. Acad. Sci. USA* 104, 5936.

Felsenstein, J. 2004a. *Inferring Phylogenies*. Sinauer Press, Sunderland, MA.

Felsenstein, J. 2004b. PHYLIP (Phylogeny Inference Package), version 3.6. Department of Genome Sciences, University of Washington, Seattle. Available at: http://evolution.genetics.washington.edu/phylip.html.

Gascuel, O. 1997. Concerning the NJ algorithm and its un-weighted version, UNJ, 149–70. In Mirkin, B., and McMorris, F.R., eds., *Mathematical Hierarchies and Biology*. American Mathematical Society, Providence.

Gascuel, O., and Steel, M. 2006. Neighbor-joining revealed. *Mol. Biol. Evol.* 23, 1997.

Griffiths, R., Jenkins, P., and Song, Y. 2008. Importance sampling and the two-locus model with subdivided population structure. *Adv. Appl. Prob.* 40, 473–500.

Guindon, S., and Gascuel, O. 2003. A simple, fast, and accurate algorithm to estimate large phylogenies by maximum likelihood. *Syst. Biol.* 52, 696–704.

Gusfield, D., Eddhu, S., and Langley, C. 2004. Optimal, efficient reconstruction of phylogenetic networks with constrained recombination. *J. Bioinform. Comput. Biol.* 2, 173–214.

Hein, J., Jiang, T., Wang, L., et al. 1996. On the complexity of comparing evolutionary trees. *Discr. Appl. Math.* 71, 153–169.

Hudson, R., and Kaplan, N. 1985. Statistical properties of the number of recombination events in the history of a sample of DNA sequences. *Genetics* 111, 147–164.

Hudson, R.R. 1983. Properties of a neutral allele model with intragenic recombination. *Theoret. Pop. Biol.* 23, 183–201.

Huggins, P., Owen, M., and Yoshida, R. 2008. First steps toward the geometry of cophylogeny. http://arxiv.org/abs/0809.1908.

Huson, D., and Kloepper, T. 2005. Computing recombination networks from binary sequences. *Bioinformatics* 21, 159–165.

Huson, D., Kloepper, T., Lockhart, P., et al. 2005. Reconstruction of reticulate networks from gene trees. *Proc. RECOMB 2005* 233–249.

Kuhner, M.K., Yamato, J., and Felsenstein, J. 2000. Maximum likelihood estimation of recombination rates from population data. *Genetics* 156, 1393–1401.

Leroy, X., Doligez, D., Garrigue, J., et al. 2007. The objective caml system, release 3.10. Available at: http://caml.inria.fr/. Accessed April 1, 2010.

Levy, D., and Pachter, L. 2008. The neighbor-net algorithm. Accepted to *Adv. Appl. Math.* Available at: http://arxiv.org/abs/math/0702515. Accessed April 1, 2010.

Linz, S., Semple, C., and Stadler, T. 2010. Analyzing and reconstructing reticulation networks under timing constraints. *J. Math. Biol.* Online first http://dx.doi.org/10.1007/s00285-009-0319-y.

Lole, K., Bollinger, R., Paranjape, R., et al. 1999. Full-length human immunodeficiency virus type 1 genomes from subtype C-infected seroconverters in India, with evidence of intersubtype recombination. *J. Virol.* 73, 152–160.

Lyngsø, R., Song, Y., and Hein, J. 2008. Accurate computation of likelihoods in the coalescent with recombination via parsimony. *Proc. RECOMB 2008*.

Maddison, W.P. 1997. Gene trees in species trees. *Syst. Biol.* 46, 523–536.

Minin, V., Dorman, K., Fang, F., et al. 2005. Dual multiple change-point model leads to more accurate recombination detection. *Bioinformatics* 21, 3034–3042.

Nakhleh, L., Warnow, T., Linder, C., et al. 2005. Reconstructing reticulate evolution in species-theory and practice. *J. Comput. Biol.* 12, 796–811.

OED (Oxford English Dictionary). 1989. "reticulated." Entry 50204774. Available at: dictionary.oed.com.

Rambaut, A. 2008. Figtree v.1.1.1. Available at: http://tree.bio.ed.ac.uk/software/figtree/. Accessed April 1, 2010.

Robinson, D., and Foulds, L. 1981. Comparison of phylogenetic trees. *Math. Biosci.* 53, 131–147.

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

Semple, C., and Steel, M. 2003. *Phylogenetics.* Oxford University Press, Oxford, UK.

Semple, C., and Steel, M. 2004. Cyclic permutations and evolutionary trees. *Adv. Appl. Math.* 32, 669–680.

Sneath, P. 1975. Cladistic representation of reticulate evolution. *Syst. Zool.* 24, 360–368.

Song, Y., and Hein, J. 2005. Constructing minimal ancestral recombination graphs. *J. Comput. Biol.* 12, 147–169.

Suchard, M.A., Weiss, R.E., Dorman, K.S., et al. 2003. Inferring spatial phylogenetic variation along nucleotide sequences: a multiple changepoint model. *J. Am. Statist. Assoc.* 98, 427–437.

Thomson, M., Delgado, E., Manjón, N., et al. 2001. HIV-1 genetic diversity in Galicia, Spain: BG intersubtype recombinant viruses circulating among injecting drug users. *AIDS* 15, 509.

Wang, L., Zhang, K., and Zhang, L. 2001. Perfect phylogenetic networks with recombination. *J. Comput. Biol.* 8, 69–78.

Westesson, O., and Holmes, I., 2009. Accurate Detection of Recombinant Breakpoints in Whole-Genome Alignments. *PLoS Comput. Biol.* 5.

Woolley, S., Posada, D., and Crandall, K. 2008. A comparison of phylogenetic network methods using computer simulation. *PLoS ONE* 3.

Wu, Y. 2008. A practical method for exact computation of subtree prune and regraft distance. *Bioinformatics* 25, 190–196.

Address correspondence to:
Dr. Frederick A. Matsen
Program in Computational Biology
Fred Hutchinson Cancer Research Center
1100 Fairview Ave. N., M1-B855
P.O. Box 19024
Seattle, WA 98109-1024
USA

E-mail: matsen@fhcrc.org