Chapter 5

Phylogenetic Tree Construction

In this chapter we will consider the problem of inferring evolutionary trees, also known as phylogenetic trees. The analysis of such trees arises when we try to infer the evolutionary relationships among individuals of a single species (e.g. the 'mitochondrial Eve analysis, or the evolutionary tree for dogs) or among various species. To make the discussion a bit more concrete, let’s assume that we are considering female descent lines, as was done for the 'mitochondrial Eve analysis.

In principle one might have a complete history of all births, and construct a descent tree incorporating all of the birth events (see Figure 7). The usual situation will be different. First of all, if we are trying to infer an evolutionary tree from currently living individuals, the complete evolutionary history is unknown. Referring to Figure 7, we might have measurements from individuals $P, Q, R, \ldots$, and the goal is to reconstruct a descent tree as shown in the figure. The data being compared, called characters, might be the corresponding amino acids in a related protein sequence, or codons from corresponding DNA sites. The characters could also be other physical characteristics (hair vs. feathers, number of toes, time for a chemical reaction involving blood, etc.).

Commonly, the inferred 'parent’, say $H$ of offspring $P$ and $Q$, is intended to represent the last common ancestor before the character change that distinguishes $P$ from $Q$. For example, if the characters were corresponding amino acid sequences

$$P \approx abceg f, \quad Q \approx azcge f,$$

then the characters for $H$ might be $H \approx azcge f$, agreeing with the characters for $Q$, but different from $P$. In representing evolutionary relationships via a tree, it may be desirable to represent the time since $H$ split into types $P$ and $Q$ by the length of the edges $HP$ and $HQ$.

It is possible to anticipate a number of issues that will arise in trying to infer phylogenies. First of all, some trees will be more reasonable than others. If
we try to develop an evolutionary tree for mammals, and we hypothesize that humans and horses are more closely related than humans and any other mammal, it is likely to take a very complex tree to characterize the genetic changes that occurred since the time of the last common ancestor. The tree would likely be simpler if the hypothesized nearest relatives of humans were chimps. Thus it will be important to have a figure of merit for evolutionary trees. Once such a figure of merit is available, it will be possible, at least in principle, to enumerate all trees and find the best ones. Thus figures of merit, and tree enumerations, are likely to play a significant role. Consideration of these ideas will also raise important concerns about computational complexity, leading to more efficient, but possibly suboptimal methods for tree construction.

Figure 7: An evolutionary tree

5.1 Tree enumeration

The first goal is to find a way to generate all trees with a given number of tips or leaves. In the biological context these are likely to be the distinct existing organisms whose evolutionary history is being determined. There are a number of variations in the descriptions of trees, which may be rooted or unrooted,
bifurcating or multifurcating, and labelled or unlabelled.

A tree is *bifurcating* if nodes have either 3 neighboring nodes (interior nodes), or 1 neighboring node (tips or leaves). Trees which allow more than 3 neighbors for interior nodes are called *multifurcating*.

![Figure 8: Distinct labelled trees](image)

A tree will be called *labelled* if each tip is assigned a distinct number (or other index). For the purposes of distinguishing trees we will not label the interior nodes, but they may be indexed for other purposes. Figure 8 shows two distinct labelled trees which would be identical (as tree shapes) if the labelling were removed.

![Figure 9: Unrooted vs. rooted trees](image)

Often, tree branches are given a direction. A tree may have a distinguished node called a *root*. In an evolutionary tree the root would be the common ancestor of all the tip organisms. A rooted tree may be directed with branches or edges pointing from ancestors to immediate descendents, in which case the
root is unique vertex with no incoming edges, and the (remaining) leaves or tips have no outgoing edges (descendants). Figure 9 shows an unrooted tree, a rooted tree (not quite bifurcating) with the added node 4, and an alternate graphical representation of the rooted tree (which now has 5 nodes).

Suppose a labelled rooted bifurcating tree has \( n \) tips (not counting the root). By induction on \( n \), starting with \( n = 2 \), every tree with \( n \) tips has the same number of edges, and that number \( E(n) \) satisfies \( E(n) = E(n-1) + 2 \), a calculation we can make by looking at what happens at the node above two tips. Since \( E(2) = 3 \), \( E(n) = 2n - 1 \). (If we delete the root edge, as some texts do, then \( E(n) = 2n - 2 \).) Since the tree is rooted, the total number of nodes (or vertices) \( V(n) \) is one more than the number of edges, so \( V(n) = 2n \).

Counting trees and having techniques to generate all trees are basic computational problems. Let \( C(n) \) denote the number of distinct labelled rooted bifurcating trees. For \( n = 2 \) there is only one possible tree, so \( C(2) = 1 \). Determination of \( C(n) \) follows from consideration of the addition and deletion of tips from trees, as Figure 10 illustrates.

\[
\text{Figure 10: Counting rooted binary trees}
\]

Start with a labelled rooted bifurcating tree \( T_1 \) with \( n \) tips, numbered \( 1, \ldots, n \). We can form another tree \( T_2 \) with tips \( 1, \ldots, n, n+1 \) by selecting any edge in \( T_1 \), adding a node \( v \) in that edge (which splits the old edge into two edges), and then adding the edge \((v, n+1)\) to \( T_1 \). Node deletion reverses this process by removing tip \( n+1 \) and its parent node. By deleting tip \( n+1 \) and its parent, we see that every tree with \( n+1 \) tips can be generated by addition from a tree with \( n \) tips. Suppose addition of tip \( n+1 \) to trees \( T_0 \) and \( T_1 \) yields the same tree \( T_2 \). Then \( T_0 \) and \( T_1 \) must be the same, since they are both obtained from \( T_2 \) by deletion of node \( n+1 \) and its parent. Consequently, addition generates every tree with \( n+1 \) tips exactly once from the set of trees with \( n \) tips. Since
5.2. PARSIMONY

the number of distinct additions to tree $T_1$ with $n$ tips is the number of edges $E(n) = 2n - 1$, we conclude that

$$C(n + 1) = C(n)(2n - 1),$$

or

$$C(2) = 1, \quad C(3) = 1 \times 3, \quad C(4) = 1 \times 3 \times 5,$$

and

$$C(n) = 1 \times 3 \times \cdots \times (2n - 3) = \frac{(2n - 3)!}{2^{n-2}(n - 2)!},$$

which grows exponentially with $n$.

The conclusions are a mixture of good and bad news. On the positive side, there is a procedure for generating all the trees of interest. On the negative side, the number of trees grows so fast with $n$ (for example $C(30) > 10^{38}$) that exhaustive search will almost never be an algorithmic option.

5.2 Parsimony

The news is happier when it comes to the issue of comparing the merits of different evolutionary trees. Recall that our tree tips represent $n$ organisms which are compared on the basis of a set of $J$ characters. If the characters come from DNA or protein analysis, then the DNA or protein sequences are assumed to be (multiply) aligned. The most straightforward analysis will assume that evolutionary changes for distince characters are independent, so we may reduce our problem to the evaluation of single character changes. Assume that the selected character has $K$ possible values.

To distinguish good phylogenies from bad ones, one typically assumes that the best trees have the smallest number of character changes. In more complex models this idea is refined by using a $K \times K$ cost matrix $M$. Assume that there is 0 cost for no change to a character, so that $M(k, k) = 0$. Other entries will typically be positive.

There is a dynamic programing algorithm, the Sanko algorithm, for computing the minimal number of character changes needed to account for the variation of $n$ organisms in a fixed tree. For each node $j$ of the tree we will compute the number $S_j(i)$, which is the minimal cost of evolutionary events in the subtree rooted at $j$, given that the character has value $i$ at node $j$.

To initialize $S_j(i)$ at the tips, let $S_j(i) = 0$ if the state at tip $j$ is $i$. Otherwise assign the value $\infty$. Given that two nodes $a, b$ with a common parent $c$ have values assigned, simply define

$$S_c(i) = \min_t[M_{i,t} + S_a(t)] + \min_m[M_{i,m} + S_b(m)].$$
The score we want for the tree will be the minimum of the root values taken over the possible root states.

The expense for this computation is just $O(nK)$, since the values $S_b(i)$ are only computed for the $V(n)$ vertices $b$ and the $K$ states $i$. If needed we could save pointers to the minimizing states in the above formula, which would allow us to also compute the minimizing character changes for the tree.

### 5.3 Heuristic approaches

Given the complexity of enumerating all possible evolutionary trees for evaluation, and the absence of efficient algorithms which can always compute the same optimal result, a variety of reasonable, but not provably correct algorithms have been proposed for the construction of phylogenies. Two approaches will be considered here. Distance based methods condense the variation of character information, and offer quick construction of candidate trees. Random or local search methods provide a means to improve trees after an initial construction.

#### 5.3.1 Distance based approaches

Alternative methods of generating hypothesized evolutionary trees can be based on measures of distance between the organisms. If $J$ characters are being used to infer phylogeny, then one can always define a distance to be the number of characters in which organisms differ. More informative definitions of distance can be handled as well.

A particularly simple distance based algorithm is called UPGMA, for unweighted pair group method with arithmetic mean. The method constructs a tree by joining groups (subtrees). Each group $a$ has a size $n_a$. The groups are initially defined to be the tips, and these groups have size 1. Suppose that $a$ and $b$ are the closest groups, with sizes $n_a$ and $n_b$. Form the new group $ab$ with size $n_{ab}$. The distance is extended so that

$$d(ab, a) = d(ab, b) = d(a, b)/2,$$

and if $c$ is another group, then

$$d(ab, c) = \frac{n_a}{n_a + n_b}d(a, c) + \frac{n_b}{n_a + n_b}d(b, c).$$

The groups $a$ and $b$ are deleted, and the process is repeated until only one group remains. Note that this method is slightly different from the text’s version, which defines

$$d(ab, c) = \frac{1}{2}d(a, c) + \frac{1}{2}d(b, c).$$
5.3.2 Walking through the space of trees

Once some reasonable tree has been constructed, there are algorithms which search for improved trees. These may be based on methods which generate new trees from existing trees. The first method can be used for a systematic exploration of all 'neighbors' of a tree. The subsequent methods make larger modifications to the structure of a tree, and are more suitable for randomized exploration of the space of trees.

Nearest neighbor search

Consider an interior edge $e_0$ of a bifurcating tree. The edge $e_0$ has two vertices $v_1, v_2$, and incident on each of these vertices are two additional edges, giving a total of five edges $e_0, e_1, e_2, e_3, e_4$. Suppose that $e_1, e_2$ were incident on $v_1$, and $e_3, e_4$ were incident on $v_2$. The situation is illustrated in Figure 11, where $T_1, \ldots, T_4$ indicate the four subtrees remaining when edges $e_0, \ldots, e_4$ are deleted from the original tree.

![Figure 11: Tree with interior edges selected](image)

Tree modifications, called nearest neighbor interchanges can be made by re-connecting the subtrees $T_1, \ldots, T_4$. There are two new trees we can form. First, join subtrees $T_1$ and $T_2$ at $v_1$, while $T_3$ and $T_4$ join at $v_2$. The second new tree joins subtrees $T_1$ and $T_3$ at $v_1$, while $T_2$ and $T_4$ join at $v_2$. These tree are illustrated in Figure 12.

A typical option is to consider such variations for each of the possible interior edges. If any tree has a better score than the starting tree, start again at this new tree. Continue 'walking uphill' until you find a tree with no improved variants. Just as in calculus, the problem with this approach is that you can get stuck in a local minimum.
To avoid being trapped in local minimum it is often desirable to use a random search early in the process, usually followed by local searches.

![Variations of trees](image)

**Figure 12: Variations of trees**

**Large scale tree modifications**

One might also wish to move around in the space of trees with larger steps than provided by the previous method. These larger steps might be taken randomly to provide a good sampling of trees early in the search for the best tree or trees. Here are two methods which are similar to each other, and are generalizations of the addition and deletion technique that was used in tree counting.

It is helpful to keep in mind that a graph, in particular a tree, can be described as a set of vertices, together with a set of edges, where each edge is an (unordered) pair of vertices. Both techniques can be viewed as operations which fix the set of vertices, but modify the collection of edges.

The first method, called subtree pruning and reattachment (SPR), is illustrated in Figure 13. Pick an edge $e_1$ with vertices $v_1$ and $v_2$. Suppose that the other vertices adjacent to $v_1$ are $A$ and $B$. Cut the graph at vertex $v_1$, thereby dividing the tree into two subtrees. The subtree $T_1$ containing $A$ and $B$ is modified by deleting the vertex $v_1$ and the edges $(A, v_1)$ and $(B, v_1)$ A new edge $(A, B)$ is inserted. Call the modified tree $\tilde{T}_1$. The other subtree $T_2$, which includes the edge $(v_1, v_2)$, is not altered.

Similar to the tip addition discussed in our tree counting exercise, the tree $T_2$ is added to another edge $(C, D)$ of $\tilde{T}_1$. This addition is effected by replacing $(C, D)$ with the edges $(C, v_1)$ and $(D, v_1)$. 

The second method, called tree bifurcation and reconnection (TBR), is illustrated in Figure 14. Pick an edge \( e_1 \) with vertices \( v_1 \) and \( v_2 \). Suppose that the other vertices adjacent to \( v_1 \) are \( A_1 \) and \( B_1 \), while the other vertices adjacent to \( v_2 \) are \( A_2 \) and \( B_2 \). Cut the graph into subtrees \( T_1 \) and \( T_2 \) by eliminating the edges \( (A_j, v_j) \) and \( (B_j, v_j) \), while adding the edge \( (A_j, B_j) \) for \( j = 1, 2 \).
Now pick an edge $e_1 = (C_1, D_1)$ in $T_1$. Delete this edge, and add the edges $(C_1, v_1)$ and $(D_1, v_1)$. In similar fashion attach $v_2$ to $e_2$ in $T_2$, and join the subtrees with the edge $(v_1, v_2)$.

5.3.3 Random search

When searching for an optimal tree, one is faced with the problem that the tree used for beginning a search may be far from an optimal one. Local searches in the space of trees may get stuck at a local optimum which is not a global optimum. To reduce the chances of encountering this problem, one often begins the search process with a randomized search of the space of trees. This can be done in a variety of ways. For instance, if a distance-based method is used, the distances between tips can be randomly altered. If one is using the tree modification methods such as nearest neighbor, SPR, or TBR, one can select edges for detachment and reattachment in a random manner.

When using a randomized method, it is common to attempt to sample widely from the space of trees early in the process. A set of high scoring trees is selected. As the search progresses, new explorations are restricted to closer neighbors of the most promising trees. This can be accomplished by control of random variations, or by switching from a randomized search to a systematic local search.

5.4 Bibliographic notes

An online tutorial is [12]. An encyclopedic treatment is [7]. A more modest treatment (2 chapters) from the probabilistic viewpoint is [4].