Lecture Notes:
The Mathematics of Phylogenetics

Elizabeth S. Allman,
John A. Rhodes

IAS/Park City Mathematics Institute
June-July, 2005

University of Alaska Fairbanks
Spring 2009, 2012, 2016

Contents

Introduction

The basic problem addressed in these notes is the following: Suppose sequences such as DNA are taken from a number of different organisms. How can we use them to understand evolutionary relationships? Schematically, an instance of the problem and a possible solution might be depicted as in Figure 1:

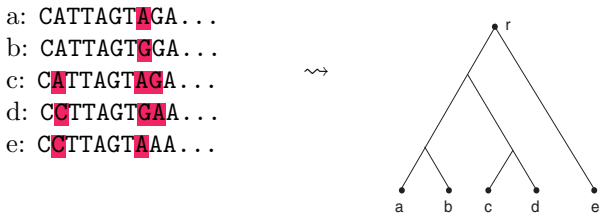

Figure 1: The basic problem of phylogenetics is to use sequences drawn from several different organisms to infer a tree depicting their evolutionary history.

Our goals are to develop some of the many things that the symbol "$\rightsquigarrow$" in this diagram might represent. Although the problem itself is a biological one, our focus will be primarily on how ideas and approaches from the mathematical sciences (mathematics, statistics, computer science) are used to attack it.

The audience we have in mind includes both biologists looking for a deeper understanding of the ideas that underlie the evolutionary analyses they may routinely perform with software, and mathematical scientists interested in an introduction to a biological application in which mathematical ideas continue to play a vital role. It is impossible to write a text that these two different groups will both find exactly to their liking — typical biologists have at most a few undergraduate math and statistics courses in their background, while mathematical scientists may have a similar (or even weaker) background in biology. Nonetheless, we believe that both groups can gain from interactions over this material, reaching a middle ground where each side better understands the other.

You will not find any direct help here on how to use software to perform

evolutionary analyses — software changes and improves rapidly enough that such a guide would become quickly outdated. You will also not find a thorough development of all the interesting mathematics that has arisen in phylogenetics (as much as the mathematician authors might enjoy writing that). Rather we hope to focus on central ideas, and better prepare readers to understand the primary literature on the theory of phylogenetics.

Before beginning, there are a few other works that should be mentioned, either for further or parallel study.

- Felsenstein's book [Fel04] is an encyclopedic overview of the field by one of its primary developers, informally presented, and offers many insightful perspectives and historical comments, as well as extensive references to original research publications.
- Yang [Yan06] gives a more formal presentation of statistical phylogenetics, also with extensive references. This book is probably too technical to serve as a first introduction to the field.
- Semple and Steel's text [SS03] provides a careful mathematical development of the combinatorial underpinnings of the field. It has become a standard reference for theoretical results of that sort.

While all of these books are excellent references, the first two lack exercises (which we feel are essential to developing understanding) and they often give an overview rather than attempting to present mathematical developments in detail. The third is an excellent textbook and reference, but its focus is rather different from these notes, and may be hard going for those approaching the area from a biological background alone. In fact, [Yan06] and [SS03] are almost complementary in their emphasis, with one discussing primarily models of sequence evolution and statistical inference, and the other giving a formal mathematical study of trees.

Here we attempt to take a middle road, which we hope will make access to both the primary literature and these more comprehensive books easier. As a consequence, we do not attempt to go as deeply into any topic as a reference for experts might.

Finally, we have borrowed some material (mostly exercises) from our own undergraduate mathematical modeling textbook [AR04]. This is being gradually removed or reworked, as editing continues on these notes.

As these notes continue to be refined with each use, your help in improving them is needed. Please let us know of any typographical errors, or more substantive mistakes that you find, as well as passages that you find confusing. The students who follow you will appreciate it.

Elizabeth Allman
e.allman@alaska.edu

John Rhodes
j.rhodes@alaska.edu

Chapter 1

Sequences and Molecular Evolution

As the DNA of organisms is copied, potentially to be passed on to descendants, mutations sometimes occur. Individuals in the next generation may thus carry slightly different sequences than those of their parent (or either parent, in the case of sexual reproduction). These changes, which can be viewed as essentially random, continually introduce the new genetic variation that is essential to evolution through natural selection.

Depending on the nature of the mutations in the DNA, offspring may be more, less, or equally viable than the parents. Some mutations are likely to be lethal, and will therefore not be passed on further. Others may offer great advantages to their bearers, and spread rapidly in subsequent generations. But many mutations will offer only a slight advantage or disadvantage, and the majority are believed to be selectively neutral, with no effect on fitness. These neutral mutations may still persist over generations, with the particular variants that are passed on being chosen by luck. As small mutations continue to accumulate over many generations, an ancestral sequence can be gradually transformed into a different one, though with many recognizable similarities to its predecessors.

If several species arise from a common ancestor, this process means we should expect them to have similar, but often not identical, DNA forming a particular gene. The similarities hint at the common ancestor, while the differences point to the evolutionary divergence of the descendants. While it is impossible to observe the true evolutionary history of a collection of organisms, their genomes contain traces of the history.

But how can we reconstruct evolutionary relationships between several modern species from their DNA? It's natural to expect that species that have more similar genetic sequences are probably more closely related, and that evolutionary relationships might be represented by drawing a tree. However, this doesn't give much of a guide as to how we get a tree from the sequences. While occa-

sionally simply 'staring' at the patterns will make relationships clear, if there are a large number of long sequences to compare it's hard to draw any conclusions. And 'staring' isn't exactly an objective process, and you may not even be aware of how you are reaching a conclusion. Even though reasonable people may agree, it's hard to view a finding produced in such a way as scientific.

Before we develop more elaborate mathematical ideas for how the problem of phylogenetic inference can be attacked, we need to briefly recount some biological background. While most readers will already be familiar with this, it is useful to fix terminology. It is also remarkable how little needs to be said: Since the idea of a DNA sequence is practically an abstract mathematical one already, very little biological background will be needed.

1.1 DNA structure

The DNA molecule has the form of a double helix, a twisted ladder-like structure. At each of the points where the ladder's rungs meet its upright poles one of four possible molecular subunits appears. These subunits, called *nucleotides* or *bases*, are adenine, guanine, cytosine, and thymine, and are denoted by the letters `A`, `G`, `C`, and `T`. Because of chemical similarity, adenine and guanine are called *purines*, while cytosine and thymine are called *pyrimidines*.

Each base has a specific complementary base with which it can form the rung of the ladder through a hydrogen bond. We always find either `A` paired with `T`, or `G` paired with `C`. Thus the bases arranged on one side of the ladder structure determine those on the other. For example if along one pole of the ladder we have a sequence of bases

`AGCGCGTATTAG`,

then the other would have the complementary sequence

`TCGCGCATAATC`.

Thus all the information is retained if we only record one of these sequences. Moreover, the DNA molecule has a directional sense (distinguished by what are called its 5' and 3' ends) so that we can make a distinction between a sequence like `ATCGAT` and the inverted sequence `TAGCTA`. The upshot of all this structure is that we will be able to think of DNA sequences mathematically simply as finite sequences composed from the 4-letter alphabet $\{\texttt{A}, \texttt{G}, \texttt{C}, \texttt{T}\}$.

Some sections of a DNA molecule form *genes* that encode instructions for the manufacturing of proteins (though the production of the protein is accomplished through the intermediate production of messenger RNA). In these genes, triplets of consecutive bases form *codons*, with each codon specifying a particular amino acid to be placed in the protein chain according to the *genetic code*. For example the codon `TGC` always means that the amino acid cysteine will occur at that location in the protein. Certain codons also signal the end of the

protein sequence. Since there are $4^3 = 64$ different codons, and only 20 amino acids and a 'stop' command, there is some redundancy in the genetic code. For instance, in many codons the third base has no affect on the particular amino acid the codon specifies.

Proteins themselves have a sequential structure, as the amino acids composing them are linked in a chain in the order specified by the gene. Thus a protein can be specified by a sequence of letters chosen from an alphabet with 20 letters, corresponding to the various amino acids. Although we tend to focus on DNA in these notes, a point to observe is there are several types of biological sequences, composed of bases, amino acids, or codons, that differ only in the number of characters in the alphabet — 4, 20, or 64 (or 61 if the stop codons are removed). One can also recode a DNA sequence to specify only purines (`R`) or pyrimidines (`Y`), using an alphabet with 2 characters. Any of these biological sequences should contain traces of evolutionary history, since they all reflect, in differing ways, the underlying mutations in DNA.

A stretch of DNA giving a gene may actually be composed of several alternating subsections of *exons* and *introns*. (Exons are the part of the gene that will ultimately be *expressed* while introns are *intruding* segments that are not.) After RNA is produced from both exons and introns, a splicing process removes those sections arising from introns, so the final RNA reflects only what comes from the exons. Protein sequences thus reflect only the exon parts of the gene sequence. This detail may be important, as mutation processes on introns and exons may be different, since only the exon is constrained by the need to produce functional gene products.

Though it was originally thought that genes always encoded for proteins, we now know that some genes encode the production of other types of RNA which are the 'final products' of the gene, with no protein being produced. Finally, not all DNA is organized into the coding sections referred to as genes. In humans DNA, for example, about 97% is believed to be non-coding. In the much smaller genome of a virus, however, which is packed into a small delivery package, only a small proportion of the genome is likely to be non-coding. Some of this is non-coding DNA is likely to be meaningless raw material which plays no current role, and is sometimes called *junk* DNA (though it may become meaningful in future generations through further mutation). Other parts of the DNA molecules serve regulatory purposes.The overall picture is quite complicated and still not fully understood.

1.2 Mutations

Before DNA, and the hereditary information it carries, is passed from parent to offspring, a copy must be made. For this to happen, the hydrogen bonds forming the rungs of the ladder in the molecule are broken, leaving two single strands. Then new double strands are formed on these, by assembling the appropriate complementary strands. The biochemical processes are elaborate, with

various safeguards to ensure that few mistakes are made in the final product. Nonetheless, changes of an apparently random nature sometimes occur.

The most common mutation that is introduced in the copying of sequences of DNA is a *base substitution*. This is simply the replacement of one base for another at a certain site in the sequence. For instance, if the sequence `AATCGC` in an ancestor becomes `AATGGC` in a descendant, then a base substitution `C`→`G` has occurred at the fourth site. A base substitution that replaces a purine with a purine, or a pyrimidine for a pyrimidine, is called a *transition*, while an interchange of these classes is called a *transversion*. Transitions are often observed to occur more frequently than transversions, perhaps because the chemical structure of the molecule changes less under a transition than a transversion.

Other DNA mutations that can be observed include the deletion of a base or consecutive bases, the insertion of a base or consecutive bases, and the inversion (reversal) of a section of the sequence. While not uncommon, these types of mutations tend to be seen less often than base substitutions. Since they usually have a dramatic effect on the protein for which a gene encodes, this is not too surprising. We'll ignore such possibilities, in order to make our modeling task both clearer and mathematically tractable. (There is much interest in dropping this restriction in phylogenetics, though it has proved very difficult to do so. Current methods that are based on modeling insertion and deletion processes can handle data sets with only a handful of organisms.)

Our view of molecular evolution, then, can be depicted by an example such as the following, in which we have an ancestral sequence S_0, its descendant S_1, and a descendant S_2 of S_1.

$$\begin{aligned} S_0 &: \texttt{ATGTCGCCTGATAATGCC} \\ S_1 &: \texttt{ATGCCGCTTGACAATGCC} \\ S_2 &: \texttt{ATGCCGCGTGATAATGCC} \end{aligned}$$

Notice site 4 underwent a net transition from S0 to S1, and then no further net mutation as S2 evolved. If we were only able to observe S0 and S2, we would still see evidence of one net transition in this site. However, while site 8 experienced at least two mutations, if we only saw the sequences S0 and S2 we could only be sure that one occurred. Site 12 shows a similar situation, where at least two mutations occurred, yet comparing S0 and S2 alone would show no evidence of either.

This illustrates that there may be *hidden mutations*, such as `C` → `T` → `G`, in which subsequent substitutions obscure earlier ones from our observation when we do not have access to sequences from all generations. A *back mutation* such as `T` → `C` → `T` is simply a more extreme case of this. The distinction between observed mutations and the actual mutations including hidden ones will be an important one when considering data.

In reality, we seldom have an ancestral DNA sequence, much less several from different times along a line of descent. Instead, we have sequences from several currently living descendants, but no direct information about any of

their ancestors. When we compare two sequences, and imagine the mutation process that produced them, the sequence of their *most recent common ancestor*, from which they both evolved, is unknown. This will produce additional complications as our analysis becomes more sophisticated.

1.3 Aligned Orthologous Sequences

Given a stretch in a DNA sequence from some organism, there are good search algorithms (such as BLAST) to find similar stretches in DNA sequences that have been found from other organisms. Thus if a gene has been identified for one organism, we can quickly locate likely candidate sequences for similar genes in related organisms. Perhaps after experimentally verifying that these are in fact genes and that they have a similar function, we can reasonably assume the sequences are *orthologous*, meaning they descended from a common ancestral sequence.

A complication can arise if an organism has undergone a *gene duplication* in which a gene that was formerly present in the genome appears with multiple copies in descendants. While the copies are still related, at least one of them is likely to be more free to vary, since the other may continue to fulfill its original role. These *paralogous* genes will retain similarities, though, and if they are passed on to several different organisms, it may be easy to mistakenly base an analysis on genes whose most recent common ancestor is the original duplicated one, rather than on those descended from a common copy after the duplication event. Even if one infers the correct evolutionary relationship between such sequences, it may be misleading as to the relationship between the organisms.

Either by trial and error, or using algorithms we will not discuss in these notes, we can *align* the sequences from the different organisms so that many of the bases match across most of the sequences. For some data sets, alignment is a trivial problem, since so much of the sequences match exactly. For other data sets, finding good alignments may be quite difficult. The essential problem is that deletions and insertions may have occurred, and we must decide where to place gaps in the various sequences to reflect this. While many software tools have been developed to do this, the computational burden of finding any sort of a 'best' alignment is so great that heuristic methods are essential. Faced with a large number of sequences, with much variation between them, even the best of current software may not produce an alignment that on human inspection appears very good. In the end, a mix of algorithms (most of which depend on first inferring at least an approximate tree) and *ad hoc* human adjustment is sometimes used to get better results. For those interested in learning more about sequence alignment, [Wat95] is a beginning source.

As the starting point for the techniques we discuss here, we take a collection of *aligned orthologous DNA sequences*. Our goal will be to produce a *phylogenetic tree* that describes their likely descent from a common ancestral sequence. Indeed, Figure 1 of the Introduction shows an example. Note that we will not

try to decide if the sequences did evolve along a tree from a common ancestor — that is an assumption we make. Our question is simply to determine the best candidate tree to describe the unknown history.

Chapter 2

Combinatorics of Trees I

2.1 Graphs and Trees

Before we discuss any of the methods of inference of phylogenetic trees, we introduce some background and terminology from graph theory. This simply formalizes the basic notions of the kinds of trees that biologists use to describe evolution, and gives us useful language.

While at first it might seem sufficient to simply draw a few examples of trees rather than define them formally, there are good reasons for being more precise. For instance, when we draw trees depicting the specific evolutionary relationships among a set of taxa, we may do this in many different ways. But while the drawings are different, they are meant to depict the same thing. Thus the idea of a tree is really an abstract one, and by giving it an abstract definition we can legitimately see that two trees are 'the same' even though the diagram we draw may have different forms. If we also look ahead to trying to write software to perform phylogenetic analyses, a computer must contain an internal representation of a tree, even though it has no visual abilities.

Definition. A *graph* G is pair $G = (V, E)$ where V is a set of *vertices* or *nodes*, and E is a set of *edges*. Each edge $e \in E$ is a two-element set $e = \{v_1, v_2\}$ of vertices $v_1, v_2 \in V$.

When $e = \{v_1, v_2\} \in E$, we say v_1 and v_2 are the *ends* of e, that e *joins* v_1 and v_2, that v_1 and v_2 are *incident* to e, and that v_1 and v_2 are *adjacent*. The *degree* or *valence* of a vertex is the number of edges to which it is incident.

Graphs are typically indicated by drawings such as that in Figure 2.1.

Recall that a set, by definition, cannot have a repeated element. This means that by requiring an edge have two vertices $v_1 \neq v_2$ we rule out the possibility of an edge looping from a vertex to itself. We also have ruled out the possibility of two or more edges having the same ends, since E also forms a set. Sometimes a graph is defined in such a way as to allow both of these possibilities, and our concept of graph is technically that of a *simple graph*. However, since we will

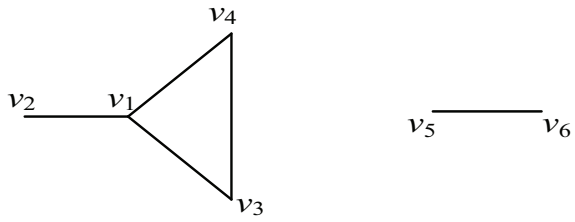

Figure 2.1: A depiction of a graph $G = (V, E)$ with $V = \{v_1, v_2, v_3, v_4, v_5, v_6\}$ and $E = \{\{v_1, v_2\}, \{v_1, v_3\}, \{v_1, v_4\}, \{v_3, v_4\}, \{v_5, v_6\}\}$. The vertices v_1, v_2, and v_3 have respective degrees 3, 1, and 2.

make no use of loops or multiple edges between the same vertices, we choose this more restrictive definition. All of our graphs will also be *finite*, meaning V, and hence E, is a finite set.

Graphs are widely used in mathematical biology to denote relationships between organism. For instance in ecology, a graph may summarize the direct interactions between species in an ecosystem, with nodes representing various species and edges between them their interactions (e.g., predation, or symbiosis). Gene interactions in a single organism can similarly be depicted by a graph, with nodes denoting genes and edges indicating when the product of one gene influences another. Graphs are sometimes referred to as *networks* in these contexts.

In phylogenetics, we often think of each vertex as representing a species, with the edges indicating lines of direct evolutionary relationships (ancestor-descendant pairs of species, along a lineage in which no other species under consideration splits off). Rather than species, however, we may be working with smaller groups of organisms such as subspecies or even individuals, or larger groups such as genera, etc. Thus we adopt the more neutral word *taxon* for whatever group is under consideration. (In scientific literature, these are sometimes referred to as *operational taxonomic units* or *OTUs*.)

If we are to use graphs to model evolution, with edges indicating lines of descent, then we of course need to rule out any taxon being an ancestor of itself. To be precise about this, we need a series of definitions.

Definition. In a graph, a *path of length* n from vertex v_0 to vertex v_n is a sequence of distinct vertices $v_0, v_1, v_2, \ldots, v_n$ such that each v_i is adjacent to v_{i+1}.

A graph is *connected* if there is a path between any two distinct vertices.

In Figure 2.1, there are two paths from v_2 to v_3, namely v_2, v_1, v_3 and v_2, v_1, v_4, v_3. However, v_2, v_1, v_2, v_1, v_3 is not a path since it repeats v_2 and v_1. Since there is no path from v_2 to v_5, this graph is not connected.

Definition. A *cycle* is a sequence of vertices $v_0, v_1, \ldots, v_n = v_0$ which are distinct (other than $v_n = v_0$) with $n \geq 3$ and v_i adjacent to v_{i+1}

In Figure 2.1, v_1, v_3, v_4, v_1 is a cycle. If we removed the edge $e = \{v_3, v_4\}$ however, the graph would not have a cycle. Note that the path v_1, v_3, v_1 is not a cycle, since the sequence of vertices is too short; this length condition rules out backtracking along a single edge being considered a cycle.

Finally, we can define

Definition. A *tree* $T = (V, E)$ is a connected graph with no cycles.

The graph in Figure 2.1 is not a tree, for two reasons; it is not connected, and it has a cycle. As an example of a tree, we have that depicted in Figure 2.2.

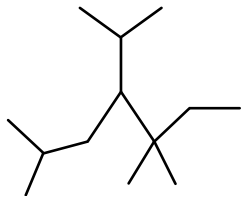

Figure 2.2: A tree.

If a vertex lies in two or more distinct edges of a tree, we say it is an *interior vertex*. If it lies in only one edge, then we call it a *terminal vertex* or a *leaf*. Similarly, an edge joining two interior vertices is an *interior edge*, while one incident to a leaf is called a *pendant edge*.

Though trees will be our primary focus, they are not the only types of graphs that are important for describing evolutionary relationships. For instance, if hybridization of two related ancestral species occurs, a graph with a cycle is needed to depict this, as the two will have both a common ancestor and a common descendant. Other types of lateral gene transfer similarly result in non-tree graphs. Much work has been done in recent years to use networks rather than trees to capture such biological phenomena, as well as for visualizing conflicting phylogenetic signals in data sets. Huson, *et al.* [HRS10] provide an excellent entry into these developments.

A basic property of trees that should be intuitively clear is the following:

Theorem 1. If v_0 and v_1 are any two vertices in a tree T, then there is a unique path from v_0 to v_1.

Sketch of proof. We leave a formalilization of the proof as an exercise, but summarize the argument: Given two different paths with the same endpoints, consider a route following first one and then the other backwards, so we return to our starting vertex. Of all such pairs of paths, choose one where this combined route is the shortest.

Now the combined route cannot be a cycle, since a tree has none. So the two paths must have a vertex other than the endpoints in common. But then we can use this to give a pair of paths producing a shorter combined route, which would be a contradiction. □

The length of the unique path between two vertices in a tree is called the *graph-theoretic distance* between the vertices. For example, in Figure 2.3, for the tree on the left the graph-theoretic distance from a to d is 3, while for the other two graphs it is 4.

The trees we have defined are sometimes called *unrooted trees* or *undirected trees*. To those who are familiar with the notion of a directed graph, it may seem odd that we emphasize trees with undirected edges for depictions of evolutionary histories. Indeed, time provides a direction that is of course central to evolutionary processes. However, we will see that many of the mathematical models and inference methods are more naturally associated with undirected trees, and so we make them our basic objects.

With that said, however, sometimes we pick a particular vertex ρ in a tree and distinguish it as the *root* of the tree. More often, we introduce a new vertex to be the root, by subdividing an edge, $\{u, v\}$, into two $\{u, \rho\}$ and $\{\rho, v\}$. Figure 2.3 shows two different rooted versions of the same unrooted tree,obtained by subdividing different edges. We use T to denote an unrooted tree, and T^ρ to denote a tree with a choice of a root ρ.

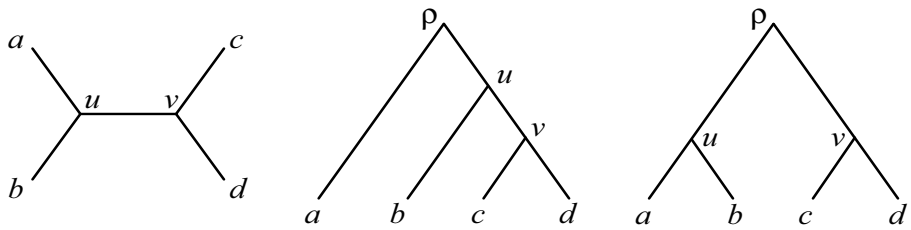

Figure 2.3: An unrooted tree and two rooted counterparts.

Biologically, a root represents the *most recent common ancestor* (MRCA) of all the taxa in the tree. Sometimes, however, roots are chosen for mathematical convenience rather than biological meaning, so one must be careful with such an interpretation.

Viewing ρ as a common ancestor of all other taxa represented in a tree, it is natural to give each edge of the tree an orientation away from the root. More precisely, a *directed edge* is not a set of two vertices, but rather an ordered pair $e = (v_1, v_2)$, where the order is specified by requiring that the path from ρ to v_1 is shorter than that from ρ to v_2. For instance, in passing from the leftmost tree to the middle tree in Figure 2.3, the undirected edge $\{u, v\}$ becomes the directed edge (u, v), since u is closer to ρ than is v.

For a directed edge $e = (v_1, v_2)$, we say v_1 is the *initial vertex* or *tail* of e and that v_2 is the *final vertex* or *head* of e. We also refer to v_1 as the *parent* of v_2, and v_2 as the *child* of v_1. More generally, for a rooted tree we may use the words *ancestor* and *descendant* in the obvious way to refer to nodes. For instance, in the middle tree of Figure 2.3, u is ancestral to d, but not to a.

An unrooted tree is said to be *binary* if each interior vertex has degree three. This terminology of course fits with the biological view of one lineage splitting into two, but this leads to the rather odd situation that a synonym for binary is *trivalent*. We call a rooted tree T^ρ binary if all interior vertices other than ρ are of degree three, while ρ is of degree two.

Although it's conceivable biologically that a tree other than a binary one might describe evolutionary history, it is common to ignore that possibility as unlikely. When non-binary trees arise in phylogenetic applications, they usually have vertices of degree greater than 3 (also called *multifurcations*) to indicate ambiguity arising from ignorance of the true binary tree. This is sometimes referred to as a *soft polytomy*. In contrast, a *hard polytomy* refers to a multifurcation representing positive knowledge of many lineages arising at once, such as a radiation of species. Since even radiations are likely to be modeled well by a succession of bifurcations with short times between them, in most applications, biologists generally prefer to find 'highly resolved' trees, with few multifurcations; a binary tree is the goal.

The trees used in phylogenetics have a final distinguishing feature — the leaves represent known taxa, which are typically currently extant and are the source of the data used to infer the tree. The internal vertices, in contrast, usually represent taxa that are no longer present, and from which we have no direct data. (Even if we have data from ancient organisms, we cannot assume they are direct ancestors of any extant ones; they are more likely to be on offshoots from the lineages leading to our modern samples.) This is formalized by labeling the leaves of the tree with the names of the known taxa, while leaving unlabeled the interior vertices. Thus for either rooted or unrooted trees we are interested primarily in the following objects.

Definition. Let X denote a finite set of taxa, or labels. Then a *phylogenetic X-tree* is a tree $T = (V, E)$ together with a one-to-one correspondence $\phi : X \to L$, where $L \subseteq V$ denotes the set of leaves of the tree. We call ϕ the *labeling map*. Such a tree is also called a *leaf-labeled tree*.

More informally, the labeling map simply assigns each of the taxa to a different leaf of the tree, so that every leaf receives a label.

Often we will blur the distinction between the leaves of a phylogenetic tree and the labels that are assigned to them. For instance, we will use T or T^ρ to denote a phylogenetic X-tree, often without explicitly mentioning either the set X or the labeling function. However, it is important to note that this labeling is crucial. Two different maps from X to the leaves of T usually produce two different phylogenetic trees. In other words, phylogenetic trees can be distinguished from one another either due to different 'shapes' of the underlying trees, or merely by a different labeling of the leaves.

2.2 Counting Binary Trees

Suppose we are interested in relating a collection X of n taxa by a phylogenetic tree. How many different trees might describe the relationship? It's helpful to know this since if we want to relate a group of taxa, the more possible trees there are, the harder the problem may be. Could we, for instance, simply list all the trees that might relate 10 taxa, and consider each one in turn? Or would this list be so long that such an approach is infeasible?

To refine the question, we first need to say what it means for two phylogenetic trees to be the same. Informally, we don't care about the names given to vertices, as long as the shape of the tree and how the labels are placed on leaves agree.

Definition. Two phylogenetic X-trees are *isomorphic* if there is a one-to-one correspondence between their vertices that respects adjacency, and their leaf labelings.

Let $b(n)$ denote the number of distinct (up to isomorphism) unrooted binary phylogenetic X-trees, where $X = \{1, 2, \ldots, n\}$. One quickly sees $b(2) = 1$, $b(3) = 1$, and $b(4) = 3$, as the diagrams in Figure 2.4 indicate.

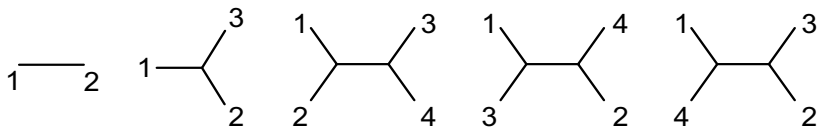

Figure 2.4: All unrooted binary phylogenetic trees for $X = \{1, 2, \ldots, n\}$, $n = 2, 3, 4$.

The key observation for counting larger trees now appears clearly when we consider $b(5)$: We can begin with any one of the 3 trees relating the first four taxa, and 'graft' another edge leading to a fifth taxon to any of the 5 edges in that four-taxon tree, so that $b(5) = 3 \cdot 5 = 15$. For $b(6)$, we can begin with any of these 15 trees and graft on another edge leading to a sixth taxon to any of the edges in that tree. To obtain a general formula, then, we also need to obtain a formula for the number of edges in these trees.

Theorem 2. An unrooted binary tree with $n \geq 2$ leaves has $2n - 2$ vertices and $2n - 3$ edges.

Proof. The statement is certainly true for $n = 2$, which is the base case for induction.

Suppose T has $n \geq 3$ leaves, and assume the statement is true for a tree with $n-1$ leaves. If v_1 is one of the leaves of T, then v_1 lies on a unique edge $\{v_1, v_2\}$, while v_2 lies additionally on two other edges $\{v_2, v_3\}$ and $\{v_2, v_4\}$. Removing these three edges and the vertices v_1, v_2 from T, and introducing a new edge $\{v_3, v_4\}$ gives a binary tree T' with $n - 1$ leaves. Since both the number of

vertices and the number of edges have been decreased by 2, for T the number of vertices must have been $(2(n-1)-2)+2 = 2n-2$, and the number of edges $(2(n-1)-3)+2 = 2n-3$. □

Theorem 3. If X has $n \geq 3$ elements, there are

$$b(n) = (2n-5)!! = 1 \cdot 3 \cdot 5 \cdots (2n-5)$$

distinct unrooted binary phylogenetic X-trees.

Proof. Again we use induction, with the base case of $n = 3$ clear.

Suppose then that T has n leaves, and let T' be the $(n-1)$-leaf tree constructed from T as in Theorem 2 by 'removing' the leaf v_1 and adjusting edges appropriately. Then with v_1 fixed, the map $T \mapsto (T', \{v_3, v_4\})$ is a bijection from n-leaf trees to pairs of $(n-1)$-leaf trees and edges. In this pair, we think of the edge $\{v_3, v_4\}$ as the one onto which we 'graft' a new edge to v_1 to recover T. Counting these pairs shows

$$b(n) = b(n-1) \cdot (2(n-1)-3) = b(n-1) \cdot (2n-5).$$

But since we inductively assume $b(n-1) = (2(n-1)-5)!! = (2n-7)!!$, we obtain the desired formula. □

This last theorem also yields a count for rooted binary trees, by simply noting that adjoining to any rooted binary tree T^ρ a new edge $\{\rho, v_{n+1}\}$, where v_{n+1} is a new leaf, gives a one-to-one correspondence between rooted binary trees with n leaves and unrooted binary trees with $n+1$ leaves.

Corollary 4. The number of rooted binary phylogenetic X-trees relating n taxa is

$$b(n+1) = (2n-3)!! = 1 \cdot 3 \cdot 5 \cdots (2n-3).$$

The formula for $b(n)$ shows it is a very large number even for relatively small values of n. (See exercises.) The implication of this for phylogenetic inference will quickly become clear: Any inference method that relies on considering every possible binary tree will be impossibly slow if the number of taxa is even moderately large.

2.3 Metric Trees

Sometimes it is useful to specify lengths of edges in a trees with non-negative numbers, as in Figure 2.5 below. Note that the lengths may either be specified by explicitly writing them next to the edges, or by simply drawing edges with the appropriate lengths, and providing a length scale.

Biologically, these lengths are usually interpreted as some measure of how much change occurred in sequences between the ends of the edge, with larger numbers denoting more change. Occasionally they represent elapsed time. However, it is generally incorrect to assume edge lengths are elapsed times, no matter how intuitively appealing that interpretation is.

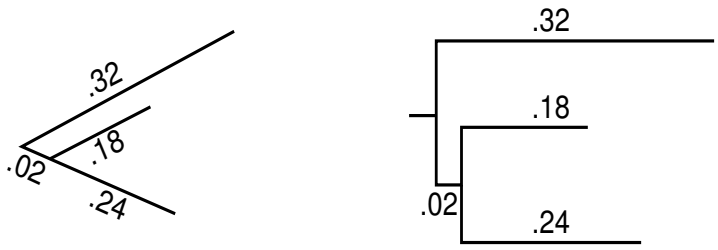

Figure 2.5: Two depictions of the same metric tree, drawn in different styles. In the second, all vertical line segments are considered to have no length, and the short horizontal segment merely indicates the root.

Definition. A *metric tree* (T, w) is a rooted or unrooted tree $T = (V, E)$ together with a function $w : E \to \mathbb{R}^{\geq 0}$ assigning non-negative numbers to edges. We call $w(e)$ the *length* or *weight* of the edge e.

When we wish to emphasize that edge lengths are *not* specified for a tree T, we will sometimes refer to T as a *topological tree*. We can obtain a topological tree from a metric one by simply ignoring the edge lengths. This means several metric trees may have the same underlying topological tree, but differ because edge lengths are not the same.

Notice the terminology conflict between the graph-theoretic notion of length of a path (the number of edges in the path) in a tree, and this new use of the word length. To avoid confusion, graph theorists tend to prefer the term 'weight,' but 'length' is more common in phylogenetics.

A metric tree leads to a way of measuring distances between its vertices. For any $v_1, v_2 \in V(T)$, define

$$d(v_1, v_2) = \sum_{\substack{e \text{ on the path} \\ \text{from } v_1 \text{ to } v_2}} w(e),$$

i.e., the sum of the edge lengths along the path between the two vertices.

Proposition 5. For a metric tree (T, w), the function $d : V \times V \to \mathbb{R}^{\geq 0}$ satisfies

(i) $d(v_1, v_2) \geq 0$ for any v_1, v_2 (non-negativity),

(ii) $d(v_1, v_2) = d(v_2, v_1)$ for any v_1, v_2 (symmetry),

(iii) $d(v_1, v_3) \leq d(v_1, v_2) + d(v_2, v_3)$ for any v_1, v_2, v_3 (triangle inequality).

If all edges of T have positive length, then, in addition,

$$d(v_1, v_2) = 0 \text{ if, and only if, } v_1 = v_2.$$

Proof. See the exercises. □

If all edges of T have positive length, then the proposition above shows d is a *metric* on $V(T)$ in the usual sense of the word in topology or analysis. In this case, we call d a *tree metric.*

If T has some edges of length zero, we can of course remove those edges, identifying the vertices at their two ends, to get a metric tree with all positive edge lengths that carries essentially the same information as our original metric tree. (However, if an edge leading to a leaf has length zero, then the resulting tree may no longer be a phylogenetic X-tree, as a label may now be on an internal vertex.) On this collapsed tree we then have that d is a tree metric. As it is often convenient to work with metric trees that have positive edge lengths, keeping this process in mind we will lose little by making such an assumption at times.

2.4 Ultrametric Trees and Molecular Clocks

If we wish to interpret edge lengths on a rooted metric tree as times, then it is essential that all leaves be equidistant from the root. This is simply because this distance would represent the total elapsed time from the MRCA of the taxa to the present, when the taxa were sampled.

Definition. A rooted metric tree is said to be *ultrametric* if all its leaves are equidistant from its root using the tree metric: For any two leaves a, b and the root ρ, $d(\rho, a) = d(\rho, b)$.

Ultrametric trees are often called *molecular clock* trees in phylogenetics, since if mutation occurs at a constant rate over all time and lineages, *i.e.* is clock-like, then many methods of inference from sequences will produce such trees in idealized circumstances. Edge lengths should be interpreted as measures of how much mutation has occurred, so that with clock-like mutation we simply scale elapsed time by a constant mutation rate to get lengths. But one should be careful — even if a tree is ultrametric, it need not have been produced under a molecular clock. For instance mutation rates could increase in time, uniformly over all lineages, and each would show the same total mutation from root to leaf.

A molecular clock assumption can be biologically reasonable in some circumstances, for instance, if all taxa are reasonably closely related and one suspects little variation in evolutionary processes during the evolutionary period under study. Other times it is less plausible, if more distantly related taxa are in the tree, and the circumstances under which they evolved may have changed considerably throughout the tree.

One attractive feature of a molecular clock hypothesis and ultrametric trees is that even if the location of the root is not known, there is a simple way to find it, as shown by the following.

Theorem 6. Suppose T^ρ is a rooted ultrametric tree with positive edge lengths. Let v_1, v_2 be any two leaves with $d(v_1, v_2)$ maximal among distances between

leaves. Then ρ is the unique vertex along the path from v_1 to v_2 with $d(\rho, v_1) = d(\rho, v_2) = d(v_1, v_2)/2$. Thus the placement of ρ can be determined from an unrooted metric version of T.

Proof. If T^ρ has more than one leaf, ρ is an internal vertex. Thus deleting ρ (and its incident edges) from T produces at least two connected components.

Suppose v_1 and v_2 are in different connected components of $T \setminus \{\rho\}$. Then the path from v_1 to v_2 must pass through ρ, and we see by Exercise 12 that $d(v_1, v_2) = d(v_1, \rho) + d(\rho, v_2)$. Since all leaves are equidistant from the root, $d(v_1, v_2) = 2d(v_1, \rho)$. Thus ρ lies the specified distance from v_1 and v_2. Since edge lengths are positive, there can be only one such vertex on the path with this property.

If v_1 and v_2 are in the same connected component of $T \setminus \{\rho\}$, then the path between them does not pass through ρ and so by Exercise 12 and the triangle inequality we have $d(v_1, v_2) < d(v_1, \rho) + d(\rho, v_2)$. But since all leaves are equidistant from the root, this shows $d(v_1, v_2) < 2d(v_1, \rho)$, and so the maximal distance between leaves cannot occur between v_1 and v_2. □

Sometimes when a root must be added to an unrooted metric tree, either as a rough approximation of the MRCA of the taxa or simply for convenience in drawing the tree, the root location is chosen as the midpoint of the path between the most distant pair of leaves. The above theorem justifies this as a valid way of locating the most recent common ancestor for a molecular clock tree, or a tree that appears to be 'close' to being one. For other trees, it is simply a heuristic method of locating a root, and cannot be reliably expected to give the true location of the MRCA.

2.5 Rooting Trees with Outgroups

In the absence of a molecular clock hypothesis, locating the root of a tree is generally not possible from mathematical considerations alone. Instead, a simple biological idea can often be used: We include an extra taxon in our study, beyond those we are primarily interested in relating. If this taxon is chosen so it is more distantly related to each of the taxa of primary interest than any of those are to each other, we call it an *outgroup*. For instance, if we are primarily interested in relating a number of species of duck, we might include some non-duck bird as an outgroup.

Provided we infer a good unrooted evolutionary tree for all our taxa including the outgroup, the vertex at which the outgroup is joined to the taxa of primary interest represents the MRCA of those, and hence serves as a root for the subtree relating only the taxa of primary interest. Thus we use prior biological knowledge of the relationship of the outgroup to the other taxa to solve our rooting problem. This is illustrated in Figure 2.6, where the taxon o designates the outgroup.

Though using an outgroup is the standard way of rooting phylogenetic trees when a molecular clock hypothesis is not reasonable, and the method usually

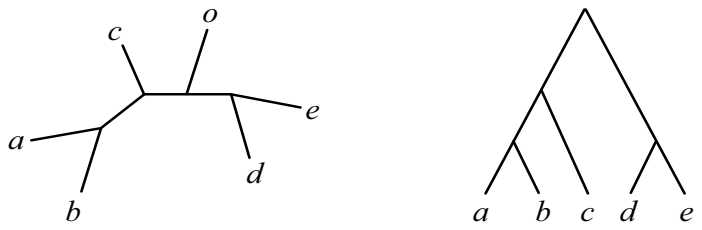

Figure 2.6: Using an outgroup o to infer a root

seems to work well, two features should be noted. First, this approach requires prior knowledge that the outgroup is distantly related to the other taxa. This may be obvious in some situations, such as that mentioned with ducks. But in others, particularly when one is attempting to infer very ancient relationships between very different taxa, it may not be clear what can serve as an outgroup. Second, even if it is clear how to pick an outgroup, there may be difficulties in inferring a tree that has it correctly placed. By definition, an outgroup is distantly related, and so whatever method is used to infer the tree may have more difficulties placing it correctly than the other taxa. If we lack knowledge of an outgroup, or doubt that we can place it on a tree correctly, then we may be forced to accept an unrooted tree as the best we can say about evolutionary relationships between the taxa of interest.

2.6 Newick Notation

There is a convenient and intuitive way of specifying a phylogenetic tree, which is simpler than the formal definition (as a collection of vertices and edges, with labelled leaves). It also has the advantage over a drawn figure of using standard typographic symbols. It is commonly used for input and/or output to computer programs, as well as in theoretical research papers.

Newick notation uses parenthetical grouping of taxon labels to specify the clustering pattern of a tree. This is best seen through examples. For instance the rooted tree on the right of Figure 2.6 is denoted $(((a, b), c), (d, e))$. However, it could also be designated $((d, e), (c, (b, a)))$, or in a number of other ways where we change the order of the items inside a pair of matched parentheses. This non-uniqueness of the Newick specification of a tree can make it hard to recognize by eye when when two large trees are the same.

If we instead wanted to designate an unrooted tree, we introduce a root arbitrarily, and use the rooted notation, simply specifying in words that we mean the tree should be unrooted. For instance the tree on the left of Figure 2.6 is the unrooted version of $(((a, b), c), ((d, e), o))$, or equivalently of $(((((d, e), o), c), b), a)$, or a host of other possibilities.

In the case of a metric tree, we append each edge length to the taxon or group below it. For example the trees in Figure 2.5 with taxa a, b, c listed from

top to bottom would be designated $(a{:}0.32, (b{:}0.18, c{:}0.24){:}0.02)$. Since removing the root effectively joins two edges, adding their length, the unrooted version of this metric tree could also be designated as $((a{:}0.34, b{:}0.18){:}0, c{:}0.24)$, or in other ways.

2.7 Exercises

1. Give sets of vertices and edges defining the first tree in Figure 2.3. Do the same for the second tree, but since it is rooted, give a set of directed edges (*i.e.* order the vertices of each edge from parent to child.) Which edge of the first tree was subdivided to create the root in the second?

2. Draw two different figures representing the tree $T = (V, E)$ with $V = \{a, b, c, d, e, f, g, h\}$ and $E = \{\{a, g\}, \{b, c\}, \{c, f\}, \{c, g\}, \{d, g\}, \{e, f\}, \{f, h\}\}$. Do this so neither drawing can be obtained from the other by just stretching or shrinking edges or angles, rotating the full figure, or by taking a mirror image of the full figure.

3. An unrooted tree has vertices $v_1, v_2, \ldots, v_9$ with edges

$$\{v_1, v_2\}, \{v_1, v_6\}, \{v_1, v_9\}, \{v_2, v_7\}, \{v_2, v_8\}, \{v_3, v_9\}, \{v_4, v_9\}, \{v_5, v_9\}.$$

   a. Without drawing the tree, determine the degree of each vertex.

   b. Use your answer to part (a) to determine the leaves of the tree.

   c. Use your answer to part (a) to determine whether the tree is binary.

   c. Draw the tree to check your work.

4. Consider the trees in Figure 2.7.

   a. Which of them are the same, as rooted metric trees?

   b. Which of them are the same, as unrooted metric trees, provided the root is deleted and its two incident edges joined into one?

   c. Which of them are the same, as rooted topological trees?

   d. Which of them are the same, as unrooted topological trees, provided the root is deleted and its two incident edges joined into one?

   e. Which trees are ultrametric?

5. Draw the three rooted binary topological trees that could describe the relationship between 3 taxa a, b, c. How do they naturally relate to the three unrooted 4-taxon trees in Figure 2.4?

6. Draw the 15 unrooted binary topological trees that could describe the relationship between 5 taxa a, b, c, d, e. Group them naturally according to the relationship they display for the first 4 taxa a, b, c, d.

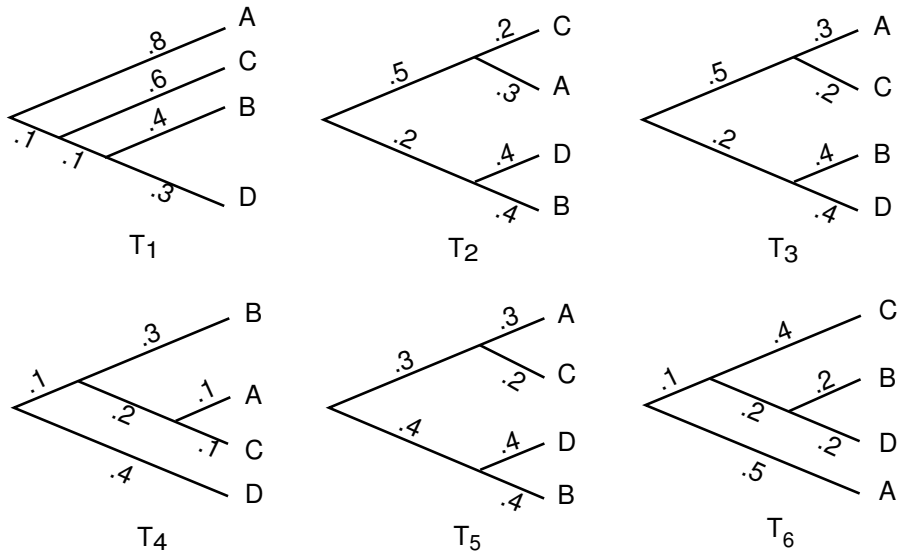

Figure 2.7: Trees for Problem 4

7. Make a table of values of the function $b(n)$, giving the number of unrooted binary X-trees for sets X of size up to 10.

8. Show that $b(n) = \dfrac{(2n-5)!}{2^{n-3}(n-3)!}$.

9. Mitochondrial DNA in humans is inherited solely from the mother. If it is used to construct a tree relating a large number of humans from different ethnic groups, then its root would represent the most recent ancestral female from which we all inherited our mitochondria. The clustering pattern of ethnic groups on such a tree might give insight into the physical location of this woman, who is sometimes called Mitochondrial Eve.

   In 1987 a work by Cann, Stoneking, and Wilson, claimed to locate Mitochondrial Eve in Africa, supporting the 'out of Africa' theory of human origins. A rooted tree was constructed that showed relationships between 147 individuals. Estimate how many topologically different trees would need to be looked at if every possibility was really examined. (You will need to use the statement in problem 8 and Stirling's formula: $n! \sim \sqrt{2\pi} n^{n+\frac{1}{2}} e^{-n}$. If you are not familiar with the asymptotic symbol '$\sim$' you can loosely interpret it as meaning 'is approximately.')

10. Give a complete proof of Theorem 1, that if T is a tree and $v_1, v_2 \in V(T)$ then there is a unique path from v_1 to v_2. (Note: You need to be careful with the fact that the term 'cycle' can only apply to sequences of at least 3 vertices.)

11. Prove Proposition 5.

12. Suppose T is a metric tree with all positive edge lengths. Prove that v_3 lies on the path from v_1 to v_2 if, and only if, $d(v_1, v_2) = d(v_1, v_3) + d(v_3, v_2)$. Show by example that this may be false if zero-length edges exist.

13. The phylogeny of four terminal taxa A, B, C, and D are related according to a certain metric tree with positive branch lengths. The tree metric distances between taxa are given in Table 2.1

	A	B	C	D
A		.5	.4	.8
B			.3	.5
C				.6

Table 2.1: Distances between taxa for Problem 13

a. Using any approach you wish, determine the correct unrooted metric tree relating the taxa, as well as all edge lengths. Explain how you rule out other topological trees.

b. Can you determine the root from these data? Explain why or why not.

c. Suppose you only had a table of numbers that were approximately those coming from a tree metric. Would whatever method you used in (a) still work? (This is the situation for real phylogenetic problems.)

14. The term 'ultrametric' originally was not applied to trees, but rather to a function $d : V \times V \to \mathbb{R}^{\geq 0}$ that satisfied not only the properties of a metric listed in Proposition 5, but also a strong form of the triangle inequality:

$$d(v_1, v_3) \leq \max(d(v_1, v_2), d(v_2, v_3)) \text{ for all } v_1, v_2, v_3.$$

a. Show this property implies the usual triangle inequality (iii) of Proposition 5.

b. Show that for a tree metric arising from an ultrametric tree, the strong triangle inequality holds when v_1, v_2, v_3 are leaves.

c. Show by a 3-leaf example that the strong triangle inequality on leaves does not hold for all tree metrics.

d. Show that if the strong triangle inequality holds, then for all choices of v_1, v_2, v_3 the two largest of the numbers $d(v_1, v_2)$, $d(v_1, v_3)$, and $d(v_2, v_3)$ are the same. (This is sometimes stated as: An ultrametric implies all triangles are isoceles.)

e. Show that if a tree metric from an unrooted 3-leaf tree satisfies the strong triangle inequality on the leaves, then there is a placement of a root for which the underlying tree is ultrametric. (This holds more generally for n-leaf tree metrics satisfying the strong triangle inequality on leaves; with proper placement of a root, they all arise from ultrametric trees.)

15. Suppose in a graph $G = (V, E)$ an edge $e = (\alpha, \beta)$ is subdivided by the introduction of a new node γ, to obtain a graph $G' = (V', E')$. What are V' and E'?

16. A rooted tree T is specified in Newick notation by $((a, b), c), d))$. List the 7 other Newick specifications for it.

17. How many different Newick specifications can be given for a rooted binary tree relating n taxa? Explain your formula.

18. How many different (rooted) Newick specifications can be given for an unrooted binary tree relating n taxa? Explain your formula.

Chapter 3

Parsimony

The first approach we'll develop for inferring phylogenetic trees is called the method of Maximum Parsimony (MP), or, more informally, parsimony. The essential idea is probably the oldest of those underlying any method, and is applied both to inferring phylogenies from sequence data and from data of other types, such as morphological data.

3.1 The Parsimony Criterion

Suppose we obtained the following aligned DNA sequences for four taxa:

```
                1
      123456789012345
S1 : AACTTGCGCATTATC
S2 : ATCTTGCGCATCATC
S3 : ATCTTGGGCATCATC
S4 : AACTTGGGCATTATC
```

There are 15 different rooted topological trees that might relate these, and we want to pick 'the best' in some sense.

Note the only variations in bases are at sites 2, 7, and 12. We might interpret site 2 as evidence for a tree such as $((S_1, S_4), (S_2, S_3))$ since a single base substitution occurring on a edge leading from the root could explain the data at that site. A tree such as $((S_1, S_2), (S_3, S_4))$, in contrast, would require at least 2 substitutions at that site. Site 7 data, on the other hand, could be explained by the smallest number of changes if $((S_1, S_2), (S_3, S_4))$ were the tree. Finally, site 12 supports the same tree as site 2. These sites conflict, but it's reasonable to decide in favor of a tree reflecting the majority of them, so $((S_1, S_4), (S_2, S_3))$ is a better choice of tree than $((S_1, S_2), (S_3, S_4))$. Of course there are 13 other rooted trees we should consider as well in order to pick the best overall for this

example, but we have at least developed the germ of a principled way of picking a tree: we look for a tree requiring the fewest mutations.

To generalize this viewpoint, imagine having n taxa which we wish to relate.

We collect a data sequence of *characters* which for each taxon might be in any of a finite number of *states*. For instance, if our data are aligned orthologous DNA sequences, then the characters correspond to the different sites in the sequences, and their possible states on each of the taxa are $\{\mathtt{A}, \mathtt{G}, \mathtt{C}, \mathtt{T}\}$. If our data are morphological, then we might have characters referring to wingedness, with states {straight, curved}, or some other body part with states such as {segmented, unsegmented, absent}. Other sources of characters might be genetic or biochemical features that vary among the taxa. Thus while our examples below will use DNA data, the method applies more generally.

The simplest form of the principal underlying parsimony is:

> *The best tree to infer from data is the one requiring the fewest changes in states of characters.*

Informally, then, a 'good' tree is one that could describe an evolutionary history with as little change as possible. Though there have been protracted and acrimonious debates about the philosophical justifications for this principal (for details see, for instance, [Fel04]), in some circumstances it is certainly a reasonable view. Under what circumstances it might be problematic will be considered later.

To mathematically formalize the principal we make a number of definitions:

Definition. A *character* with state set S on a set of taxa X is a function $\chi : X \to S$. If $s = |S|$, we say χ is an s-state character.

We will assume all sets are finite here. By passing to a numerical encoding of states, we could even assume $S = \{1, 2, \ldots, s\}$.

Our data for the parsimony problem for a set of taxa X are then a finite sequence of characters $\mathcal{C} = \{\chi_1, \chi_2, \ldots, \chi_k\}$ where χ_i is an s_i-state character on X for each i. For DNA data, the characters χ_i just refers to the ith site in the aligned sequences, k is the length of the sequences, and $s_i = 4$ for all i, with $S_i = S = \{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{T}\}$. For morphological data, the s_i and sets S_i may vary, and the ordering of the characters would be arbitrary.

Note that we refer to a *sequence* of characters since that terminology is mathematically valid and in accord with biological usage when, say, DNA sequences are used. However, we could more generally have referred to a set or *multiset* of characters since we will not in fact use the specific ordering. Recall that multisets are unordered, but allow an element to be repeated, unlike sets. For morphological data, there is generally no natural ordering to the characters, so the multiset terminology would be more natural.

Consider a fixed phylogenetic X-tree T, either rooted or unrooted, with leaves labeled by the taxa in X. For the remainder of this chapter we'll make no distinction between the leaves of T and their labels in X.

While our data are composed of characters χ on X, to judge the number of state changes T requires, we also need to consider characters $\tilde{\chi}$ on the full vertex set $V(T)$. In accordance with standard terminology for functions, we say a character $\tilde{\chi}$ on $V(T)$ is an *extension* of χ to T if $\tilde{\chi}(x) = \chi(x)$ for all $x \in X$. That is, an extension of a character assigns the same states to the leaves as the original character, but also assigns states to internal vertices. We use $Ext_T(\chi)$ to denote the set of all extensions of χ to T, and think of its elements as representing the possible evolutionary histories that are consistent with the observation of χ.

For each edge $e = \{v, w\} \in E(T)$ and character $\tilde{\chi}$ on $V(T)$, define

$$\delta(e, \tilde{\chi}) = \begin{cases} 1 & \text{if } \tilde{\chi}(v) \neq \tilde{\chi}(w) \\ 0 & \text{otherwise} \end{cases}.$$

Viewed as a function of e, with $\tilde{\chi}$ fixed, this is the indicator function of those edges where a state change occurs in the evolutionary history $\tilde{\chi}$.

Definition. The *state-change count* for $\tilde{\chi}$ on a phylogenetic X-tree T is the number of edges on which a state change occurs for $\tilde{\chi}$:

$$c(\tilde{\chi}, T) = \sum_{e \in E(T)} \delta(e, \tilde{\chi}).$$

The *parsimony score* of an X-tree T for a character χ on X is the minimal state-change count achieved by an extension of χ:

$$ps_\chi(T) = \min_{\tilde{\chi} \in Ext_T(\chi)} c(\tilde{\chi}, T).$$

We say $\tilde{\chi}$ is a *minimal extension* of χ for T if

$$c(\tilde{\chi}, T) = ps_\chi(T).$$

These definitions are for a single character. For a sequence of characters we need the following.

Definition. The *parsimony score* of a phylogenetic X-tree T for a sequence of characters $\{\chi_1, \ldots, \chi_k\}$ on a taxon set X is the sum of those for the individual characters:

$$ps_{\{\chi_i\}}(T) = \sum_{i=1}^{k} ps_{\chi_i}(T).$$

The set of *most parsimonious trees* for a sequence of characters $\{\chi_1, \ldots, \chi_k\}$ is the collection of trees achieving the minimal parsimony score:

$$\{T \mid ps_{\{\chi_i\}}(T) = \min_{T'} ps_{\{\chi_i\}}(T')\}.$$

Our goal now is to find a method that will take a sequence of characters on X and find the set of most parsimonious trees for it. There are several difficulties we face. First, we need to consider all possible trees that might relate our taxa. We've already discussed enumerating these in the last chapter, so, at least in principal, we could methodically consider one tree after another. However, for even midsize n, the number of n-taxon binary phylogenetic trees is enormous. We'll return to this issue later on.

Second, for each tree we consider, to compute the parsimony score we apparently must consider all possible extensions of the data characters. Even if a character has only 2 states, since the number of internal nodes of an n-taxon rooted binary phylogenetic tree is $n-1$, the number of extensions of that character is 2^{n-1}. For DNA data the numbers becomes 4^{n-1}. This exponential growth with$n$ means that considering each of these extensions in turn would also require an enormous amount of work, and become infeasible for a large number of taxa. Fortunately, there is a way around this computational issue, which we turn to in the next section.

3.2 The Fitch-Hartigan Algorithm

The problem of computing $ps_{\{\chi_i\}}(T)$ for a *fixed* tree T is sometimes called the *small parsimony problem*, while doing so for *all* trees T is the *large parsimony problem*. Although both problems at first appear to be computationally intensive, the small problem can in fact be solved efficiently. A simple algorithm was developed independently by Fitch and Hartigan, and proved to work as claimed by Hartigan in 1973. Although it can be easily generalized, we present it here only for binary trees. We illustrate it and motivate it by an example:

Suppose we look at a single character (in this example, a site in the aligned DNA sequences) for each of five taxa and observe states (bases) as shown:

$$\mathrm{S}_1 : \texttt{A}, \quad \mathrm{S}_2 : \texttt{T}, \quad \mathrm{S}_3 : \texttt{T}, \quad \mathrm{S}_4 : \texttt{G}, \quad \mathrm{S}_5 : \texttt{A}.$$

For the leaf-labeled tree T of Figure 3.1 we wish to compute $ps_\chi(T)$.

After writing the appropriate character state at each leaf, we simply trace backwards up the tree, from the leaves to the root, drawing reasonable conclusions as to what the state of the character might be at each interior vertex, assuming the fewest possible state changes occurred. For instance, at the parent of S_1 and S_2 we could have had either an `A` or a `T`, but obviously not a `C` or a `G`, if we are to minimize state changes, and at least 1 state change must have occurred on the edges below it. We thus write the set of possibilities $\{\texttt{A}, \texttt{T}\}$ at that vertex, and have a count of 1 so far. Now, given what appears at S_3, at the most recent common ancestor of S_1, S_2, and S_3 we should have a `T` (and in fact a `T` at the parent of S_1 and S_2); no additional change is necessary, beyond the 1 we already counted. We've now labeled two interior vertices, and still have a count of 1.

We continue to trace backward through the tree, writing a state or set of possible states at each vertex. If the children of the current vertex are marked

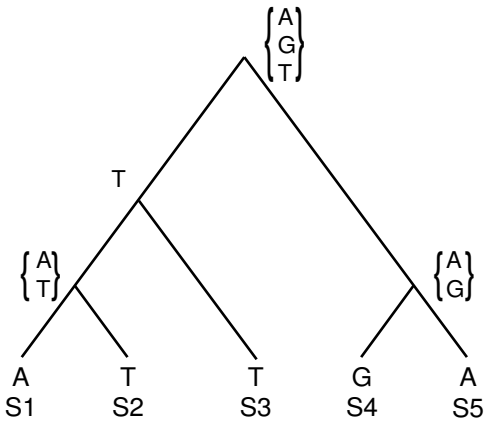

Figure 3.1: Computing the parsimony score of a tree using the Fitch-Hartigan algorithm for one character constructs sets of states at internal nodes, and yields a score of 3.

with two different states (or disjoint sets of states), we increase our state-change count by 1 and combine the two states (or the sets) to create a larger set of possible states at the parent vertex. If the two children's states agree (or the sets are not disjoint), then we label the parent vertex with that state (or the intersection of the sets). In this case no additional changes need to be counted. When all the vertices of the tree are labeled, the final value of the mutation count gives the minimum number of mutations (or state changes) needed if that tree described the evolution of the taxa. Thus the tree in Figure 3.1 would have a parsimony score of 3.

While this illustrates the algorithm, there are several points to be addressed. First, while it should seem reasonable, it is not clear that this method gives the minimum possible mutations needed for the tree. It does correctly calculate the parsimony score for the tree and character, but that must be proved. (You might also suspect the algorithm produces all minimal extensions of the character on the tree. However, it does *not*, as Exercise 9 shows. There can be assignments of states to the internal vertices that are not consistent with what this algorithm produces, yet which still achieve the same state-change count. This illustrates the importance of proving an algorithm works as one wishes; naive intuition can be misleading.)

Second, we performed the algorithm on a *rooted* tree. While we've been intentionally unclear as to whether we are considering rooted or unrooted trees with the parsimony criterion, in fact it doesn't matter, as we now show.

Theorem 7. If T^ρ is a rooted version of T, then for any character χ,

$$ps_\chi(T^\rho) = ps_\chi(T).$$

Proof. If ρ is a vertex of the tree T, then by the definition of the parsimony score this is clear.

So suppose ρ has been introduced along an edge of T, by replacing that edge with two edges meeting at ρ. Observe that any minimal extension of χ to T^ρ must have the same state at ρ as at least one of the adjacent vertices.

But then we see $ps_\chi(T^\rho) \geq ps_\chi(T)$, since when any minimal extension of χ to T^ρ is restricted to the vertices of T, its state-change count remains the same. But also $ps_\chi(T^\rho) \leq ps_\chi(T)$ since given any minimal extension of χ to T, we can further extend it to T^ρ without altering its state-change count. □

Thus the parsimony score does not depend on the root location, and once we show the Fitch-Hartigan algorithm gives the parsimony score for a rooted tree, it must give the same score regardless of root location. While the details of the algorithmic steps depend on the choice of root, the final score does not.

Before we give more proofs, though, let's return to our example. Now that we've evaluated the parsimony score of the tree in Figure 3.1, let's consider another tree, in Figure 3.2, that might relate the same 1-base sequences.

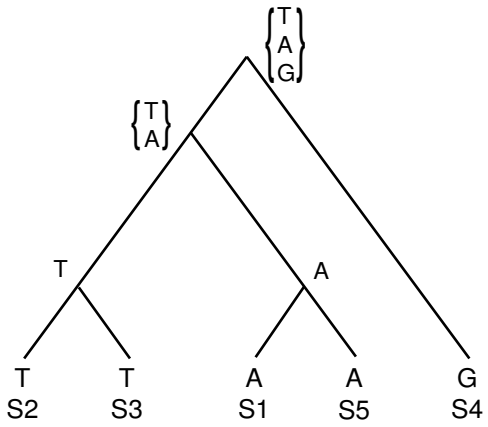

Figure 3.2: Using the same character as in Figure 3.1, the Fitch-Hartigan algorithm shows this tree is more parsimonious, with a score of 2.

Applying the method above to produce the labeling at the internal vertices, we find this tree has a parsimony score of 2; only two mutations are needed. Thus the tree in Figure 3.2 is more parsimonious than that of Figure 3.1.

To find the most parsimonious tree for these taxa we would need to consider all 15 possible topologies of unrooted trees with 5 taxa and compute the minimum number of mutations for each. Rather than methodically go through the 13 remaining trees, for this simple example we can just think about what trees are likely to have low parsimony scores. If the score is low, S1 and S5 are likely to be near one another, as are S2 and S3, but S4 might be anywhere. With this

observation, its easy to come up with several more trees that have parsimony score 2 but that are topologically distinct from that of Figure 3.2.

It's also easy to see that no tree will have a parsimony score of 1, since we need at least 2 mutations to have 3 different bases among the taxa. For this example there are in fact five trees that have a parsimony score of 2, which form the set of the most parsimonious.

Here's a more succinct presentation of the Fitch-Hartigan algorithm for computing the parsimony score $ps_\chi(T)$ of a binary tree T, for a single character χ on X with state space S.

Algorithm (Fitch-Hartigan).

1. If T is unrooted, arbitrarily introduce a root, to get a rooted binary tree T^ρ.
2. Assign to each vertex $v \in V(T^\rho)$ a pair (U, m) where $U \subseteq S$ and $m \in \mathbb{Z}^{\geq 0}$ as follows:
   (a) To each leaf $x \in X$, assign the pair $(\{\chi(x)\}, 0)$.
   (b) If the two children of v have been assigned pairs (U_1, m_1) and (U_2, m_2), then assign to v the pair
   $$(U, m) = \begin{cases} (U_1 \cup U_2,\ m_1 + m_2 + 1), & \text{if } U_1 \cap U_2 = \emptyset, \\ (U_1 \cap U_2,\ m_1 + m_2), & \text{otherwise.} \end{cases}$$
   Repeat until all vertices have been assigned pairs.
3. If the pair (U, m) has been assigned to ρ, then $ps_\chi(T) = m$.

With real data, we of course need to count the number of state changes required for a tree among *all* characters. Since the score for a sequence of characters is the sum of scores for each character, this can be done in the same manner as above, just treating each character in parallel. An example is given in Figure 3.3.

Proceeding up the tree beginning with the two taxa sequences ATC and ACC on the far left, we see we don't need mutations in either the first or third site, but do in the second. Thus the mutation count is now 1, and the ancestor vertex is labeled as shown. At the vertex where the edge from the third taxa joins, we find the first site needs a mutation, the second does not, and the third does. This increases the mutation count by 2, to give us 3 so far. Finally, at the root, we discover we need a mutation only in the second site, for a final parsimony score of 4.

While this is not hard to do by hand with only a few sites, as more sites are considered it quickly becomes a job best done by a computer.

We now prove the Fitch-Hartigan algorithm actually produces the correct parsimony score for a rooted tree. For simplicity, we consider only binary trees. The key idea is to prove something slightly stronger, as expressed in the last sentence of the following.

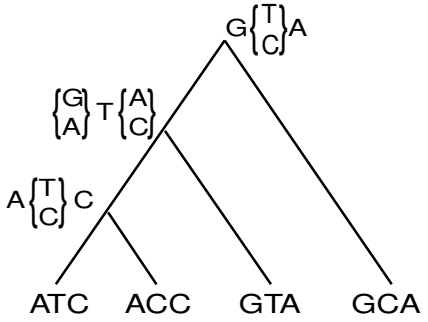

Figure 3.3: For multiple characters, the Fitch-Hartigan algorithm can be applied in parallel, yielding a score of 4 in this example.

Theorem 8. Let χ be a character on X, and T^ρ a rooted binary X-tree. Then the Fitch-Hartigan algorithm computes $ps_\chi(T)$. Furthermore, the set of states it assigns to the root ρ is exactly the set of states that occur at ρ in the minimal extensions $\tilde{\chi}$ of χ.

Proof. We proceed by induction on the size of the set X. The $|X| = 2$ case is clear.

For general $|X|$, let $v_1, v_2 \in V(T^\rho)$ be the children of ρ. Let T_i be the subtree, rooted at v_i, of descendants of v_i, and X_i denote the labels on the leaves of T_i, so that $X = X_1 \cup X_2$ is a disjoint union. Let $\chi_i = \chi|_{X_i}$. By the inductive hypothesis, the Fitch-Hartigan algorithm assigns to v_i the pair (U_i, m_i) where U_i is exactly the set of states that occur at v_i in minimal extensions of χ_i on T_i, and m_i is $ps_{\chi_i}(T_i)$.

Suppose $\tilde{\chi}$ is a minimal extension of χ to T, and let $\tilde{\chi}_i = \tilde{\chi}|_{V(T_i)}$. Then (see Exercise 7a) minimality ensures one of the following must hold:

(1) $\tilde{\chi}(\rho) = \tilde{\chi}(v_1) = \tilde{\chi}(v_2)$,

(2) $\tilde{\chi}(\rho) = \tilde{\chi}(v_1) \neq \tilde{\chi}(v_2)$, or

(3) $\tilde{\chi}(\rho) = \tilde{\chi}(v_2) \neq \tilde{\chi}(v_1)$.

Consider first case (2). Then $\tilde{\chi}_2$ must be a minimal extension of χ_2, for otherwise we could contradict the minimality of $\tilde{\chi}$ by defining a character on T that agreed with $\tilde{\chi}$ on $V(T_1) \cup \{\rho\}$ and agreed with a minimal extension of χ_2 to T_2 on $V(T_2)$.

We similarly see that $\tilde{\chi}_1$ must be a minimal extension of χ_1 by applying the same argument to the minimal extension $\tilde{\tilde{\chi}}$ of χ defined by

$$\tilde{\tilde{\chi}}(v) = \begin{cases} \tilde{\chi}(v) & \text{if } v \neq \rho, \\ \tilde{\chi}(v_2) & \text{if } v = \rho. \end{cases}$$

Thus in this case $\tilde{\chi}_1, \tilde{\chi}_2$ are both minimal extensions of χ_1, χ_2, respectively, while $\tilde{\chi}(v_1) \neq \tilde{\chi}(v_2)$. Using the inductive hypothesis, this shows $ps_\chi(T) = m_1 + m_2 + 1$. It also shows there do not exist any minimal extensions of χ_i on the T_i which agree on v_1 and v_2, since the existence of such would allow us to construct an extension of χ on T with a state-change count of $m_1 + m_2$, contradicting the minimality of $\tilde{\chi}$. Thus the U_i are disjoint, and $\tilde{\chi}(\rho) \in U_1 \cup U_2$. Finally, we note that for each choice of an element of $U_1 \cup U_2$ we can use minimal extensions of the χ_i on T_i to define a minimal extension of χ taking on the specified choice of state at ρ. This completes the proof in case (2). Case (3) is essentially the same.

In case (1), we cannot conclude that both $\tilde{\chi}_i$ are minimal extensions of the χ_i, but only that at least one must be (see Exercise 7b). Assume then that $\tilde{\chi}_1$ is minimal. We consider two subcases, according to whether $\tilde{\chi}_2$ is (a) minimal, or (b) not minimal.

In subcase (1a), we have that both $\tilde{\chi}_i$ are minimal, and $\tilde{\chi}(\rho) = \tilde{\chi}(v_1) = \tilde{\chi}(v_2)$. Thus $ps_\chi(T) = m_1 + m_2$, the U_i are not disjoint, and $\tilde{\chi}(\rho) \in U_1 \cap U_2$. To see that for every $s \in U_1 \cap U_2$ there is a minimal extension with $\tilde{\chi}(\rho) = s$, we choose any minimal extensions χ_1, χ_2 with $\chi_1(v_1) = \chi_2(v_2) = s$ and define $\tilde{\chi}$ to agree with them on the subtrees and have $\tilde{\chi}(\rho) = s$. Such a $\tilde{\chi}$ achieves the minimal score.

In subcase (1b), pick any minimal extension $\tilde{\tilde{\chi}}_2$ of χ_2. Define a new character $\tilde{\tilde{\chi}}$ on $V(T)$ to agree with $\tilde{\chi}$ on $V(T_1) \cup \{\rho\}$, and agree with $\tilde{\tilde{\chi}}_2$ on $V(T_2)$. Note that the state-change count for $\tilde{\tilde{\chi}}$ cannot be greater than that of $\tilde{\chi}$, since the number of state changes on edges in T_2 has been decreased, at the expense of only one additional state change on the edge $\{\rho, v_2\}$. Since it also cannot have a lower state-change count by the minimality of $\tilde{\chi}$, $\tilde{\tilde{\chi}}$ is a minimal extension of χ, with both $\tilde{\tilde{\chi}}_1$ and $\tilde{\tilde{\chi}}_2$ minimal on the subtrees. Thus $ps_\chi(T) = c(\tilde{\tilde{\chi}}, T) = m_1 + m_2 + 1$, and $\tilde{\chi}(\rho) = \tilde{\tilde{\chi}}(\rho) \in U_1 \cup U_2$. From $ps_\chi(T) = m_1 + m_2 + 1$, it is straightforward to see that the U_i must be disjoint. Finally, that a minimal extension of χ exists with any element in $U_1 \cup U_2$ as its value at ρ is as in case (2). □

3.3 Informative Characters

We can save some computational effort in finding parsimonious trees if we observe that some characters do not affect the parsimony scores of trees in ways that affect our judgment of which is optimal.

The obvious case is a character χ that assigns the same state to all taxa, *i.e.*, a constant character. This character extends to a constant character on $V(T)$ for any T, and hence $ps_\chi(T) = 0$ for all T. Thus we can simply discard the character from a sequence of characters, and not change parsimony scores.

A less obvious case is a character χ that assigns the same state to all taxa but a few, assigning different states to each of these few taxa. For example, suppose a character assigned the state `A` to one taxon, `G` to another, and `C` to all others. Then whatever tree is considered will certainly require at least 2

state changes, since 3 states are observed. On the other hand, regardless of the tree, the extension of the character that assigns a `C` to all internal nodes would achieve a state-change count of 2. Thus this character will have a parsimony score of 2 on every tree, and its inclusion in a data set of characters will only increase the parsimony score of every tree by the same amount. It therefore will have no effect on our choice of the best tree under the parsimony criterion.

To formalize this reasoning, suppose $|\chi^{-1}(i)| > 1$ for at most one state i. In this case, let j denote a state with $|\chi^{-1}(j)|$ maximal, and $l = |\chi(X)|$. Then for any T we can extend χ to T by assigning to each internal vertex the state j. This shows $ps_\chi(T) \leq l - 1$. Since $ps_\chi(T) \geq l - 1$ for any χ and T (see Exercise 8), this means $ps_\chi(T) = l - 1$ for all trees T. While the appearance of such a χ among a sequence of characters *does* affect the parsimony score of all trees, it simply inflates them all by the same amount, and so *does not* affect the selection of the most parsimonious trees.

This leads to the idea of an *parsimony-informative character*.

Definition. A character on X is *(parsimony-)informative* if it assigns to the taxa at least two different states at least twice each. That is, χ is informative if, and only if,

$$\left|\left\{i \in S \ : \ \left|\chi^{-1}(i)\right| > 1\right\}\right| > 1.$$

Before applying the Fitch-Hartigan parsimony algorithm, we can eliminate all non-informative characters from our data since they will not affect the choice of most parsimonious trees. For DNA sequence data, many characters are likely to be non-informative, since identical, or nearly-identical sites are needed to identify and align orthologous sequences. In the examples above, you'll notice only informative sites have been used.

There are several explicit warnings we should make concerning the notion of informative sites. First, the concept applies *only* when one is using the parsimony criterion for tree selection. Under other methods, deletion of these sites will lead to biased results — they contain useful information under other methods.

Second, even under parsimony the non-informative sites only have no information for selecting the *topological* tree. After finding the most parsimonious topological tree, sometimes lengths are assigned to edges as a measure of how many state changes had to occur on that edge. To do this, all minimal extensions of the characters are considered (see Section 3.6), and for each edge an average is taken of the counts associated to it by each extension. The resulting metric tree does depend on all characters except the ones that are identical for all taxa.

Additional 'pre-processing' of the data can reduce the computations a bit more. For instance, the particular states of a character at the leaves of a tree do not matter, as long as we understand which are different. For instance on a 5-taxon tree, a character that has states `A`,`C`,`C`,`G`,`A` at leaves 1-5 will produce the same parsimony score as a character with states `G`,`A`,`A`,`T`,`G` at those leaves, since both are instances of a pattern of the form $xyyzx$. By counting the number of

occurrences of each such pattern form in the data, parsimony scores can be found by only performing the Fitch-Hartigan algorithm on a single representative of each.

3.4 Complexity

If an algorithm is to be performed on a large data set, it is important to understand how much work this will require. Even if a computer is used (as of course is always the case), if the amount of work is too great, runtimes can be unacceptably slow for an algorithm to be useful. For instance, if our approach to the small parsimony problem was not the Fitch-Hartigan algorithm, but rather the more naive and cumbersome approach of listing all possible extensions of our characters, computing a score for each, and then picking the smallest, we could never expect to have software capable of analyzing data sets with a large number of taxa in an acceptable amount of time. We've already seen that the number of such extensions is exponential in the number of taxa, and thus impossibly large once there are many taxa.

To see that the Fitch-Hartigan algorithm is better than this, we have to quantify the amount of work in it. To do this, first consider a single character on an n-taxon set, with s possible states. A measure of the work to perform a basic step of the process (*i.e.*, to compare the sets of states at two child nodes, determine the set for the parent, and possibly increase the state-change count) is Cs, since we will have to consider each possible state in turn, and do a fixed amount of work C for each. Since a rooted binary n-leaf tree has $n-1$ internal vertices, we will perform this work $n-1$ times, for a total of $Cs(n-1)$. If there are k characters, then the work must be done for each, for a total of $Csk(n-1) < Cskn$. Thus the amount of work grows only linearly in the number of taxa, a vast improvement over the exponential growth of the naive approach.

Although we have not given a value for C, there is certainly such a constant (which might be measured in units of time, or computer operations). In standard jargon we would say the Fitch-Hartigan algorithm has runtime $\mathcal{O}(skn)$ (read "big-O of skn"), to indicate the runtime is less than some unspecified constant times the given product. In general, algorithms with polynomial runtimes are considered feasible, and those with exponential or worse are problematic. The linear runtime here is excellent. (If the unspecified constant were huge, that might also be a problem, but at least for the Fitch-Hartigan algorithm it should be clear that it is quite small.) Thus the algorithm gives a very efficient way of solving the small parsimony problem.

However, this is only for one tree T. For the large problem, we must compare parsimony scores for all unrooted binary trees, and there are $(2n-5)!!$ of them to consider. Searching through them all would be horribly slow. A natural question asks if this slowness is unavoidable.

The answer to this last question appears to be 'yes.' More technically, finding

a most parsimonious tree is an NP-hard problem[1], and so we do not expect any algorithm will be found that *guarantees* we have found all (or even one) optimal trees when n is large. (That no NP-hard problem can be solved in polynomial time has not yet been proved, but is widely believed. It is one of the most famous open mathematical questions.)

In practice, there are heuristic search methods that seem to work well and reasonably quickly. However, they do this by not considering all possible trees. In some circumstances, they can rule out a class of trees as not possibly containing the most parsimonious. But more often, after finding a reasonably good tree they spend most of their time focusing on only searching among trees that are "similar". As a result, only when the number of taxa is small can we be sure we have really found the most parsimonious trees. We will delay a discussion of methods of exploring similar trees in a heuristic search until a later chapter.

3.5 Weighted Parsimony

A natural generalization of the basic idea of parsimony is to introduce different penalties for different state changes in computing parsimony scores. For instance, in DNA sequence data it is often clear that transitions and transversions occur at different rates. If transversions are more rare, they may be less prone to occurring multiple times at the same site than transitions, and thus preserve more phylogenetic signal. We might then choose to give them a higher weight in calculating parsimony scores, to take advantage of this. We now develop an algorithm for computing the weighted parsimony scores that would result from such a scheme.

The Fitch-Hartigan algorithm for computing unweighted parsimony scores is an example of a *dynamic programming* algorithm, a rather common algorithmic approach to solving problems. Roughly this means the algorithm proceeds by finding optimal solutions to smaller instances of the problem (on subtrees) in order to use those solutions to get an optimal solution to the problem we care about.

The dynamic programming approach was used by Sankoff to develop an algorithm for computing weighted scores. It is very similar to the Fitch-Hartigan algorithm, but requires more bookkeeping as we work our way through the tree.

Suppose χ is an s-state character on X, and fix an $s \times s$ matrix $W = (w_{ij})$ of weights. Here w_{ij} is the cost we wish to impose for a state change from state i to state j in the descent from the root. While it is natural to assume $w_{ii} = 0$ and $w_{ij} \geq 0$ for $i \neq j$, it is not required that we do so.

For a rooted X-tree T^ρ, the weighted parsimony score is defined as follows:

[1]This is due to Foulds and Graham (1983). Ron Graham was sent to Fairbanks while in the U.S. AIr Force in the 1950s, then earned his undergraduate degree at UAF. He was later awarded an honorary doctorate from UAF.

Definition. For any extension $\tilde{\chi}$ of χ on T^ρ, the *cost* of $\tilde{\chi}$ is

$$c_W(\tilde{\chi}, T^\rho) = \sum_{e \in E(T)} w_{\tilde{\chi}(t_e)\tilde{\chi}(h_e)}.$$

Here $t_e, h_e \in V(T)$ denote the tail and head vertices of the edge e directed away from the root.

Definition. The *weighted parsimony score* of T^ρ is

$$ps_{\chi,W}(T^\rho) = \min_{\tilde{\chi} \in Ext_{T^\rho}(\chi)} c_W(\tilde{\chi}, T^\rho).$$

Rather then begin with an example, we'll formally state an algorithm for computing weighted parsimony scores first. Again, we only treat binary trees, since generalizing to arbitrary trees is straightforward. Suppose X, T^ρ, and χ are given.

Algorithm.

1. Assign to each leaf $x \in X$ of T^ρ an s-element vector $\mathbf{c}$, where

$$c_i = \begin{cases} 0 & \text{if } \chi(x) = i, \\ \infty & \text{otherwise .} \end{cases}$$

2. If the two children of $v \in V(T)$ have been assigned vectors $\mathbf{a}$ and $\mathbf{b}$, then assign to v the vector $\mathbf{c}$ with

$$c_i = \min_{j \in S}(w_{ij} + a_j) + \min_{k \in S}(w_{ik} + b_k).$$

   Repeat until all vertices have been assigned vectors.

3. If the vector $\mathbf{c}$ has been assigned to ρ, output $m = \min_i(c_i)$.

Figure 3.4 shows an example of the algorithm, where transitions have been given weight 1, transversions weight 2, and no substitution weight 0. The final score for the tree is thus found to be 3. (While the character in this example is uninformative for *un*weighted parsimony, that does not mean it must be uninformative if weights are used. See Exercise 19.)

We'll omit a formal proof that Sankoff's algorithm gives the correct weighted parsimony score, as it is similar to that for the Fitch-Hartigan algorithm.

Note also that allowing weights to be assigned to state changes may have made the parsimony score of a tree dependent on the root location. While this may be desirable for some purposes, it means we will have to search through all rooted trees rather than just unrooted ones. If we restrict the allowable choices of weights for state changes, however, we can preserve the independence of the parsimony score from the root location. We leave determining how this is done as Exercise 16.

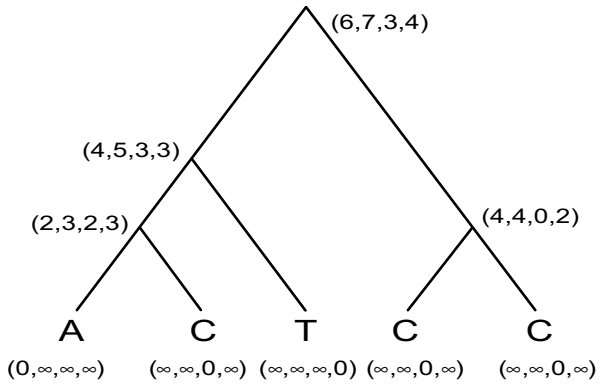

Figure 3.4: The Sankoff algorithm for computing weighted parsimony scores, requires that scoring vectors be computed at each internal node. In this example, transitions are weighted 1 and transversions weighted 2, with the base order A, G, C, T used for scoring vectors.

3.6 Recovering Minimal Extensions

In addition to the parsimony score of a tree, we may actually want to know what minimal extensions of χ achieve that score. Thinking of minimal extensions as likely evolutionary histories, we might want to know for each such history which state changes occurred on which edges of a tree. Across many characters, this would allow us to identify on which edges we had many state changes and which had few, thus indicating something about either the length of time those edges represent, or whether circumstances were highly favorable to state changes during those periods. Of course the various minimal extensions may indicate state changes along different edges, so at best we will be able to assign a range to the number of changes (or the total cost for weighted parsimony) along each edge, or the average over all minimal extensions.

Although we'll outline how to find all minimal extensions only in the setting of the weighted parsimony algorithm of Sankoff, there is also a more streamlined variation for the unweighted case that builds on the Fitch-Hartigan algorithm.

Suppose then that we have computed the weighted parsimony score of a character χ on a tree T^ρ by the algorithm above. If $\mathbf{c}$ is the vector assigned to ρ, then for each i with c_i minimal, there will be a minimal extension of χ taking on state i at ρ.

Considering only one such minimal c_i, let v_1 and v_2 be the children of ρ, with assigned vectors $\mathbf{a}, \mathbf{b}$. Then for each choice of state j at v_1 and state k at v_2 with

$$c_i = w_{ij} + a_j + w_{ik} + b_k,$$

there will be a minimal extension of χ taking on the states i, j, k at ρ, v_1, v_2, re-

spectively. We simply continue down the tree in this way, to eventually produce all minimal extensions.

3.7 Further Issues

There are a number of issues with parsimony that we won't discuss in detail, but that we leave for you to think about:

1. If several trees tie for most parsimonious (as they often do), what should we do? There are various approaches to defining *consensus trees* that attempt to summarize the features of several trees by a single tree. We will touch on this in Chapter 4

2. What about trees that are 'close' to most parsimonious? Are we really committed enough to the parsimony principle to feel that being only a few state changes off from most parsimonious means a tree should not be considered as biologically reasonable?

3. With more than a handful of taxa, there are many possible binary trees to consider that might relate them. If there are too many to search through to be sure of finding the most parsimonious, what heuristic search methods are likely to do well in practice?

4. If weighted parsimony is to be used, how should weights be assigned? Since assigning weights can affect which trees we view as 'best,' how can we objectively choose?

5. We can also choose to weight various characters differently in computing the parsimony score of a tree. This is different than assigning weights to different state changes for a particular character: For instance we might have morphological characters for body size and for wing shape of insects, and decide that wing shape changes are more significant indicators of evolutionary relationships than body size ones, so they should count twice as much in the parsimony score. More generally, it might be reasonable to assign higher weights to characters representing features of an organism that change less often, and lower ones to characters that vary more freely. But if this is done, how do we justify the precise choices of those weights? (Conversely, if this is not done, how do we justify giving all characters equal weights?)

6. A final issue, which has been the subject of much controversy among biologists, concerns the philosophical underpinning of parsimony. There are strong proponents of parsimony who believe it is the *only* valid way of inferring phylogenies. There are others who may prefer other methods but admit parsimony is a reasonable approach under some circumstances. In Chapter 11 we will see that some reasonable probabilistic models of mutation processes can sometimes lead to data that will cause parsimony to infer an incorrect tree.

We cannot yet explore this issue fully, but will give a hint as to part of the problem. If, say, we are discussing DNA mutation, then along an edge of a tree that represents a long period of time a single site may mutate several times, for instance `A` $\rightarrow$ `C` $\rightarrow$ `G`, or `A` $\rightarrow$ `C` $\rightarrow$ `A`. Thus while two state changes occurred, either one or none is observed if we only consider the ends of the edge. The parsimony principle, however, rejects the possibility of multiple changes as a preferred explanation.

As long as state changes are rare on all edges of a tree, parsimony is a quite reasonable approach to inferring a phylogeny. It is when state changes are less rare that potential problems arise. For characters describing morphology, or other larger-scale observations, we often expect few, if any, hidden mutations. For sequence data, the situation is less clear and depends on the data set being examined.

3.8 Exercises

1. a. Using the Fitch-Hartigan algorithm, compute the minimum number of base changes needed for the trees in Figure 3.5.

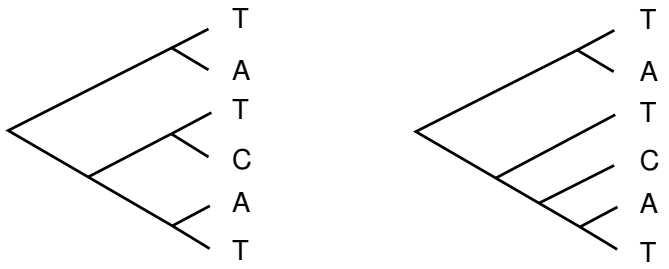

Figure 3.5: Trees for Problem 1

b. Give at least three trees that tie for most parsimonious for the one-base sequences used in part (a).

c. For trees tracing evolution at only one DNA site as in (a) and (b), why can we always find a tree requiring no more than 3 substitutions no matter how many taxa are present?

2. a. Find the parsimony score of the trees in Figure 3.6. Only informative sites in DNA sequences are shown.

b. Draw the third possible (unrooted) topological tree relating these sequences and find its parsimony score. Which of the three trees is most parsimonious?

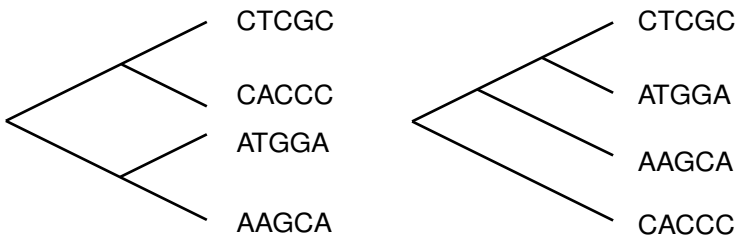

Figure 3.6: Trees for Problem 2

3. What changes are necessary to the Fitch-Hartigan algorithm if a tree has a polytomy? Give an example of how you would handle a vertex with 3 children.

4. Consider the following sequences from four taxa.

```
          1
  123456789012345
S1: AATCGCTGCTCGACC
S2: AAATGCTACTGGACC
S3: AAACGTTACTGGAGC
S4: AATCGTGGCTCGATC
```

   a. Which sites are informative?

   b. Use the informative sites to determine parsimony scores for the three possible unrooted trees relating the taxa. Which is the most parsimonious?

   c. If S_4 is known to be an outgroup, use your answer to (b) to give a rooted tree relating S_1, S_2, and S_3.

5. Though non-informative sites in DNA do not affect which tree is judged to be the most parsimonious, they do affect the parsimony score. Explain why, if P_{all} and P_{info} are the parsimony scores for a tree using all sites and just informative sites, then

$$P_{\text{all}} = P_{\text{info}} + n_1 + 2n_2 + 3n_3,$$

   where, for $i = 1, 2, 3$, by n_i we denote the number of sites with all taxa in agreement except for i taxa which are all different . (Notice that while P_{all} and P_{info} may be different for different topologies, $n_1 + 2n_2 + 3n_3$ does not depend on the topology.)

6. Show that if χ is an informative character on a set X of n taxa (so $n \geq 4$), then there exist two phylogenetic X-trees T_1, T_2 with $ps_\chi(T_1) \neq ps_\chi(T_2)$.

7. Complete the proof of Theorem 8 by doing the following:

(a) Suppose $\tilde{\chi}$ is a minimal extension of χ to T, and let $\tilde{\chi}_i = \tilde{\chi}|_{V(T_i)}$, with the T_i as defined in the proof. Explain why minimality ensures one of the following must hold: (1) $\tilde{\chi}(\rho) = \tilde{\chi}(v_1) = \tilde{\chi}(v_2)$, (2) $\tilde{\chi}(\rho) = \tilde{\chi}(v_1) \neq \tilde{\chi}(v_2)$, or (3) $\tilde{\chi}(\rho) = \tilde{\chi}(v_2) \neq \tilde{\chi}(v_1)$.

(b) Explain why in case (1) at least one of the $\tilde{\chi}_i$ is a minimal extension of χ_i. Give an example to show both need not be minimal. (You can do this with a 4-leaf tree.)

8. Suppose χ is any character on X, and let $l = |\chi(X)|$. Explain why the lower bound $ps_T(\chi) \geq l - 1$ holds for any phylogenetic X-tree T.

9. For the character and first tree in Figure 3.7, calculate the parsimony score, labeling the interior vertices according to the Fitch-Hartigan algorithm. Then show that the second tree requires exactly the same number of base changes, even though it is not consistent with the way you labeled the interior vertices on the first tree. (The moral of this problem is that naively interpreting the Fitch-Hartigan algorithm will not produce all minimal extensions of a character.)

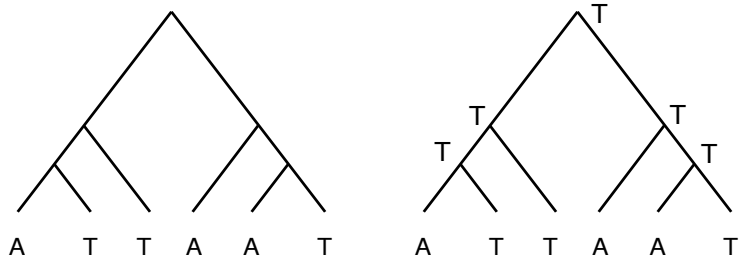

Figure 3.7: Trees for Problem 9

10. If characters are given for 3 terminal taxa, there can be no informative ones. Explain why this is the case, and why it doesn't matter.

11. The bases at a particular site in aligned DNA sequences from different taxa form a *pattern*. For instance, in comparing $n = 5$ sequences at a site, the pattern (`ATTGA`) means `A` appears at that site in the first taxon's sequence, `T` in the second's, `T` in the third's, `G` in the fourth's, and `A` in the fifth's.

a. Explain why in comparing sequences for n taxa, there are 4^n possible patterns that might appear.

b. Some patterns are not informative, such as the 4 patterns showing the same base in all sequences. Explain why there are $(4)(3)n$ non-informative patterns that have all sequences but one in agreement.

c. How many patterns are non-informative because 2 bases each appear once, and all the others agree?

d. How many patterns are non-informative because 3 bases each appear once, and all others agree?

e. Combine your answers to calculate the number of informative patterns for n taxa. For large n, are most patterns informative?

12. A computer program that computes parsimony scores might operate as follows: First compare sequences and count the number of sites f_i for each informative pattern p_i that appears (e.g., for 4 taxa, we might have $p_1 =$ `AAAA`, $p_2 =$ `AAGG`, $\dots$). Then, for a given tree T, calculate the parsimony score $ps_{p_i}(T)$ for each of these patterns. Finally, use this information to compute the parsimony score for the tree using the entire sequences. What formula is needed to do this final step? In other words, give the parsimony score of the tree in terms of the f_i and $ps_{p_i}(T)$.

13. Parsimony scores can be calculated even more efficiently by using the fact that several different patterns always give the same score. For instance, in relating four taxa, the patterns `ATTA` and `CAAC` will have the same score.

    a. Using this observation, for 4 taxa how many different informative patterns must be considered to know the parsimony score for all?

    b. Repeat part (a) for 5 taxa.

14. Use the Sankoff algorithm to computed the weighted parsimony score, with 1:2 weights for transitions:transversions, for the following aligned sequences on the tree which shows A and B most closely related.

    A: `TGA`
    B: `TAT`
    C: `TGT`
    D: `TAC`

15. For DNA data, what weight matrix makes weighted parsimony the same as unweighted parsimony? Using it, perform the Sankoff algorithm on the tree and character in Figure 3.1, and see that you recover the same score the Fitch-Hartigan algorithm produced, and the same states at the root.

16. What natural condition on the weight matrix ensures that weighted parsimony will always give the same score to any character extension regardless of the root location of the tree?

17. How would you modify the Sankoff algorithm for a tree with a polytomy?

18. Create an example of 4 sequences where unweighted parsimony leads to a different optimal tree than weighted parsimony with the transversion weight twice that of transitions.

19. When using weighted parsimony, one must be careful with the notion of an uninformative character. Consider a scheme in which transitions and transversions are weighted as 1 and 2, as in the example in Figure 3.4.

a. Explain why for the character in Figure 3.4 under this weighting every tree will produce a score of no more than 3. (Hint: assign all internal nodes the same state to achieve this score.)

b. Explain why for the character in Figure 3.4 under this weighting no tree can achieve a score lower than 3, and thus the character is uninformative under the weighted scheme.

c. Change the character so the taxon with state `T` instead has state `G`. Then find two trees on which this new character has different parsimony scores. This shows that the character is informative under the weighted scheme, though it would be uninformative under standard (unweighted) parsimony.

20. Explain why the weighted parsimony algorithm for one s-state character on a set of taxa X will involve $\mathcal{O}(s^2|X|)$ steps.

21. When applying weighted parsimony to sequence data such as for DNA, it is reasonable to require the weight matrix satisfy the condition

$$w_{ij} + w_{jk} \geq w_{ik}, \text{ for all } i, j, k.$$

Explain why, and how this relates to the possibility of multiple mutations along an edge.

22. How would you modify the Sankoff algorithm for weighted parsimony to deal with missing information on a character's value for some of the taxa?

Chapter 4

Combinatorics of Trees II

Now that we've developed one method of phylogenetic inference, parsimony, we turn back to combinatorial considerations of phylogenetic trees. Our focus will be on different ways we can view the information expressed by a tree, and how those ways lead to methods of summarizing a collection of trees on the same taxon set in a single *consensus* tree. This is often done when parsimony returns many trees with the same minimal parsimony score, to summarize their common features. However, it is improtant in many other situations as well.

We also briefly introduce a similar situation where one has a collection of trees on possibly *different* but overlapping sets of taxa. Then the goal is to combine the trees into a single *supertree* that shows the relationships that occur in all the given trees, or if that is not possible attempts to show many of the relationships.

4.1 Splits and Clades

Suppose T is an unrooted phylogenetic X-tree. Then if we delete any one edge e of T, we partition the taxa X into two subsets, according to the two connected components of the resulting graph. For instance, deleting the marked edge in the tree T of Figure 4.1 produces the partition $\{a, b, d\}, \{c, e, f\}$, which we call a *split*.

Definition. A *split* of X is any partition of X into two non-empty disjoint subsets. We write $X_0|X_1$ to denote the split with subsets X_0, X_1.

If T is a phylogenetic X-tree, and $e \in E(T)$, then the *split induced by* e is the partition of X obtained by removing the edge e from T and forming the subsets of X that appear on each of the two connected components of the resulting graph. We also say such a split is *displayed on* T.

In the split notation $X_0|X_1$, the order of the sets does not matter; $X_0|X_1$ is the same as $X_0|X_1$. Also, if $X_0|X_1$ is a split, then $X_1 = X \setminus X_0$ is the complement of X_0. Thus to specify a split, we really only need to give one of

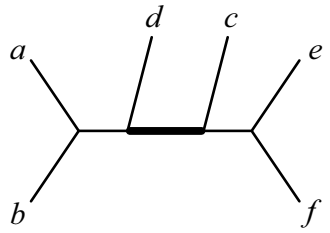

Figure 4.1: An edge inducing the split $\{a, b, d\}|\{c, e, f\}$.

the subsets. For instance, we might choose to give the smaller of the two, or the one that contains a particular taxon.

For a given set X of n taxa, there are $2^{n-1} - 1$ splits (see Exercise 2). However, a binary tree has only $2n - 3$ edges, and non-binary trees have even fewer. Thus at most $2n - 3$ splits will be displayed on any one tree.

A split induced by a pendent edge of a tree is particularly simple: In a tree relating n taxa, if a leaf is labelled by taxon S_1, then the split associated to that edge is $\{S_1\}|\{S_2, S_3, \ldots, S_n\}$. This split reflects nothing more about a tree than that S_1 labels a leaf. For that reason, a split $X_0|X_1$ in which one of the X_i has only a single element is said to be *trivial*. While trivial splits tell us essentially nothing interesting about a tree, the non-trivial splits induced by internal edges carry more information about the tree.

Suppose now that T is a phylogenetic X-tree and we consider the splits induced by two different edges e_1, e_2 of T. Removing both e_1 and e_2 from T produces three connected components, and thus a partition of X into three disjoint subsets X_1, X_2, X_3. With appropriate numbering of these sets, the split induced by e_1 is $X_1|X_2 \cup X_3$ and that induced by e_2 is $X_1 \cup X_2|X_3$. This shows splits induced from edges of a phylogenetic X-tree will have the following pairwise property.

Definition. Two splits $Y_0|Y_1$ and $Y_0'|Y_1'$ of a set Y are said to be *compatible* if for some i, j we have $Y_i \cap Y_j' = \emptyset$.

It's not hard to convince yourself that if you were given a pair of compatible splits, then you could find a tree displaying both of them. But what if we have a larger collection of splits? More broadly, to what extent does a collection of pairwise compatible X-splits correspond to a phylogenetic tree?

To answer this question, we need a slight generalization of a phylogenetic tree. While binary phylogenetic trees remain the biologists' ideal, sometimes the best tree we can produce has higher-degree vertices, some labels on internal vertices (which you might think of as leaves which have not yet been resolved from the interior of the tree, and multiple labels on some vertices (which you might also think of as a result of not yet resolving the branching of all taxa from one another). All that we will need is captured in the following definition.

Definition. An *X-tree* is a tree T together with a labeling map $\phi : X \to V(T)$ such that for each $v \in V(T)$ of degree 1 or 2 is labelled; *i.e.*, $\phi^{-1}(v) \neq \emptyset$ for all v of degree at most 2.

Obviously we can refer to splits of X induced by edges of an X-tree, just as for phylogenetic trees. Moreover, any two splits induced by edges in a single X-tree will be compatible.

We now establish an important theorem giving the relationship between compatible splits and trees.

Theorem 9. (Splits Equivalence Theorem) Let S be a collection of splits of X. Then there is an X-tree whose induced splits are precisely those of S if, and only if, the splits in S are pairwise compatible. Furthermore, this X-tree is unique, up to isomorphism.

Proof. That an X-tree induces pairwise compatible splits has been discussed, so suppose we have a set of pairwise compatible splits S and wish to construct such an X-tree. We proceed by induction on the size of S. The case $|S| = 1$ is straightforward, since a one-edge tree whose two vertices are labeled by the taxa in the two split sets has the desired property, and no other X-tree does.

Let $S = S' \cup \{X_0|X_1\}$ where $X_0|X_1 \notin S'$. Then by induction, there is an X-tree T' whose induced splits are precisely those of S'. Color red all those vertices of T' that are labeled by elements of X_0, and color blue all those labeled by elements of X_1, so every vertex is either uncolored, red, blue, or both red and blue.

Our first goal is to show there is a unique vertex v in T' whose removal results in connected components all of whose colored vertices have the same color. To do this, let T_1 be the minimal spanning tree of all blue vertices in T', and T_2 the minimal spanning tree of all red vertices in T'.

Observe that T_1 and T_2 cannot have an edge in common since if they did that edge of T' would induce a split not compatible with $X_0|X_1$: There would be both red and blue vertices in both components of the graph left by the edge's removal. However, if T_1 and T_2 are disjoint then they are joined by some path in T', but picking any edge on that path induces the split $X_0|X_1$, and since $X_0|X_1 \notin S'$ we have a contradiction. Thus T_1 and T_2 must have only vertices in common. In fact, they can have only one vertex in common, since having more vertices in common would imply they have edges in common, and that has been ruled out. Call this vertex v.

Note that when v is removed from T in each of the resulting connected components all colored vertices have the same color. Indeed, if there were a component with both red and blue vertices in it, then the red and blue minimal spanning trees would have in common the edge from v leading into this component, and we know a shared edge is impossible.

Furthermore v is the unique vertex with this property, since removing any other vertex will leave all of the minimal spanning tree of one color in a single component, and at least part (and hence some colored nodes) of the other in the same one.

Now we modify T' to get T by replacing v by two vertices v_0 and v_1 connected by an edge e, reconnecting the edges of T' incident to v so that red components join to v_0 and blue components to v_1. We modify the labeling map for T' to get a labeling map for T by reassigning all of v's labels from X_0 to v_0, and all of v's labels from X_1 to v_1. We now see that the new edge e of T induces the split $X_0|X_1$, while all other edges of T induce precisely the splits of S'.

That T is uniquely determined by S we leave as Exercise 7. □

Notice that the argument given in this proof can easily be formulated as an algorithm, called *Tree-Popping*, for constructing an X-tree from a collection of splits. To illustrate this, we show only one step which might be done in the middle of a longer process. Suppose the graph on the left of Figure 4.2 has already been constructed from some splits, each of which was compatible with the split $\{de\}|\{abcf\}$. Then we color the vertex labeled de red, and the other

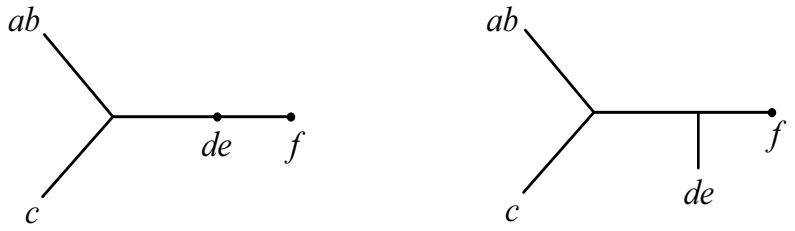

Figure 4.2: Incorporating the split $\{de\}|\{abcf\}$ into a tree following the proof of Theorem 9.

labeled vertices blue. If there were a red-blue vertex, that would have to be the node we seek, since it would lie on both minimal spanning trees. But since there is no red-blue vertex, to distinguish the vertex v that will be replaced, we must find the minimal spanning trees of the vertices of each color. For red, this just a single vertex, while for the blue, it is the entire tree. These intersect at the vertex labeled de. We thus remove this vertex, replacing it by an edge with all red vertices joined at one end, and all blue at the other. This gives the tree on the right of Figure 4.2.

It is instructive to work through a more complete example, such as in Exercise 9.

We also note that if the collection of splits includes all trivial ones (which, by Exercise 3, are always compatible with the other splits), then Tree Popping will produce a phylogenetic X-tree, where each of the taxa is located at a distinct leaf.

The notion of a split is useful for unrooted trees. Its analog for rooted trees is that of a *clade*:

Definition. If X is a set of taxa, then a *clade* is simply a non-empty subset of X.

If T^ρ is a rooted phylogenetic X- tree, and $v \in V(T)$, then the subset $U \subseteq X$ of all taxa descended from v on T^ρ is called the *clade induced by* v. We also say the clade U is *displayed* on T^ρ.

The *trivial* clades are those with only one element (which are induced by a vertex that is a leaf of the tree), and the full set X (which is induced by the root of the tree). Clades are sometimes called *clusters*, or if they are induced by a vertex on a rooted tree, *monophyletic groups*.

There is a simple relationship between clades on rooted trees and splits on unrooted trees, provided we are careful about how we relate the rooted and unrooted trees: Given an n-taxon rooted tree T^ρ, create an $(n+1)$-taxon unrooted tree $\tilde{T}$ by attaching an additional edge at the root, and labeling the new leaf this introduces with a new taxon name, say r. (If the tree was originally rooted by the use of an outgroup which was then deleted, this process simply reintroduces the outgroup.) Then each clade U displayed on T^ρ determines a split $U|V$ on $\tilde{T}$ with $V = X \cup \{r\} \setminus U$. Conversely, each split $U|V$ on $\tilde{T}$ determines a clade on T^ρ, by simply choosing which ever of the sets U, V does not contain r. We will take advantage of this fact in discussing consensus trees, since a method based on using splits from unrooted trees will automatically give a method using clades on rooted trees.

Notice that this correspondence between splits and clades requires that we change the number of taxa. We are *not* simply ignoring the root location in passing from a rooted tree to an unrooted one, but rather encoding it in a special way. This has implications for the notion of a consensus tree below: ignoring the root location and constructing a consensus tree using splits may give a different result than constructing a consensus tree using clades (encoded as splits on a larger taxon set) and then ignoring the root. (See Exercise 18.)

The notion of compatibility of splits carries over to clades, capturing the ability of two clades to be displayed on the same tree:

Definition. Two clades U_1, U_2 of X are said to be *compatible* if either $U_1 \subseteq U_2$, $U_2 \subseteq U_1$ or $U_1 \cap U_2 = \emptyset$.

4.2 Refinements and Consensus Trees

To illustrate the utility of the splits viewpoint on trees, we give two applications.

First, we can use Theorem 9 to determine when two X-trees can be 'combined' into a more refined X-tree. This is useful in a biological setting if we have several non-binary X-trees that we've inferred, perhaps from different data sets, that have managed to resolve different parts of some unknown binary tree. We want to combine the information they contain to give a tree that shows more resolution. For example, in Figure 4.3, we see two trees which have a common minimal refinement, in a sense that can be made precise through splits.

More formally, we say a tree T_2 is a *refinement* of T_1 if every split displayed by T_1 is also displayed by T_2.

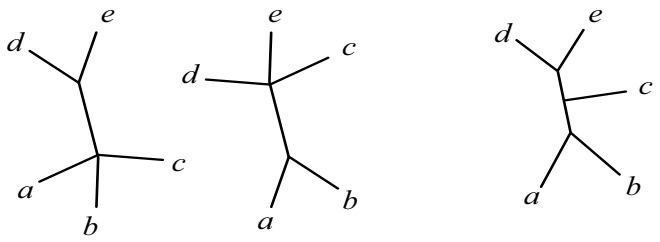

Figure 4.3: Two trees and their common refinement

By Theorem 9 we easily see the existence of a common refinement of two (or more) trees is equivalent to the compatibility of the splits of the two trees.

Corollary 10. Suppose $T_1, T_2, \ldots, T_n$ are X-trees. Then all induced edge splits from these trees are pairwise compatible if, and only if, there exists an X-tree T whose displayed splits are precisely those of the T_i. Furthermore, if this X-tree T exists, it is unique.

In the example of Figure 4.3, one could list the splits from the trees on the left (there is only one non-trivial split for each tree), check that they are pairwise compatible, and then apply the Tree Popping algorithm to produce their common refinement on the right. Of course this example is small, and the correct common refinement is clear without being so formal, but this gives a procedure that will produce common refinements in more complicated settings, with many larger trees.

A more common situation, though, is to have a collection of X-trees which have some incompatible splits, which therefore can not all be displayed on a single tree. Then we might want to combine them into a single *consensus tree* that only shows features common to all or most, and somehow ignores the incompatibilities. For instance, this is frequently done when parsimony has been used to infer trees, since many trees may tie for most parsimonious. Other situations where this may be desirable include when different genes have been used to infer highly resolved trees for the same set of taxa, but the trees are not in complete agreement.

There are several different versions of what should be called a consensus tree. We define these only for unrooted trees, using splits. For analogous versions using clades, one simply introduces an extra taxon to encode the root, as discussed earlier.

Definition. Suppose X-trees $T_1, T_2, \ldots, T_n$ are given. Then the *strict consensus tree* is the tree displaying precisely the splits that are displayed on every T_i. The *majority-rule consensus tree* is the tree displaying precisely the splits displayed on more than half of the T_i.

Note that the splits displayed on all T_i must be pairwise compatible since every pair is displayed on a single tree (in fact, on any of the T_i). Thus by

the Splits Equivalence Theorem, a strict consensus tree exists and is easily constructed by Tree Popping. Since the splits we use are a subset of those displayed on any of the T_i, the result will be a tree that has every T_i as a refinement. In practice, a strict consensus tree is often very poorly resolved; if there is a single T_i not displaying a particular split, then that split is not displayed by the strict consensus tree. The strict consensus approach effectively gives every tree veto power over the splits shown on the others.

The majority-rule consensus tree is defined to weaken the requirement of complete agreement. The 50% cutoff for a split 'winning' the vote is due to the following.

Theorem 11. Any two splits that are each displayed on more than 50% of the trees in a collection must be pairwise compatible.

Proof. If two splits each occur on more than half the trees, then, by the pigeonhole principle, they must both occur on at least one tree. But splits occurring on the same tree are compatible. □

Because of this theorem, we know the splits occurring with frequency above 50% across the trees are pairwise compatible, and thus by the Splits Equivalence Theorem they determine a tree. Thus the majority-rule tree actually exists, and can be found by Tree Popping.

Another possibility is the *loose consensus tree*. Of the splits displayed on any of the given trees, it displays exactly those that are compatible with all the trees. It will therefore display every split that the strict consensus tree does, but often some more, and thus be a refinement of it. It need not be a refinement of the majority-rule consensus tree, though, since a split displayed most trees that is incompatible with a single tree will not be displayed on the loose consensus tree, but will be on the majority-rule one. This is used less commonly than majority rule, perhaps because a single 'error' in one tree can effectively veto a correct split in all the others.

One could also consider consensus trees between the strict and majority rule levels, by choosing a parameter q between $1/2$ and 1. Taking all the splits displayed on more than the fraction q of the trees, we again obtain a pairwise compatible set, and thus determine a unique tree displaying precisely those splits.

It's also possible to consider cut-offs lower than $1/2$, but then we cannot be sure all such splits will be compatible. To get around this issue, one can rank the splits displayed on all the trees, from highest to lowest, by the number of trees on which they appear. We then accept the first split in the list. Proceeding down the list, we accept each split that is compatible with all those previously accepted. If we continue this process through the full list, and then construct a tree displaying all accepted splits, we have a *greedy consensus tree*. The motivation behind this is that, once we pass below the 50% frequency level, we are using compatibility with the more frequent splits to help us choose which of the less frequent splits we should display.

One issue with a greedy consensus tree is that we may have several splits with equal ranking, and then the list ordering is chosen arbitrarily. A different ordering can then lead to acceptance of a different collection of splits, and hence to a different greedy consensus tree. Software implementations should at least warn of this situation, though not all do so.

There are other approaches to consensus trees, that seek to resolve conflicts using different criteria. One of these (MRP) we discuss below in the context of supertrees, though it is straightforward to use it unchanged to construct a consensus tree. Another is the *Adams consensus tree*, which exists only for rooted trees. It shows the 'nesting' of groups of taxa within others that appears in the clade structure of the input trees. Its construction should be explained in a later version of these notes.

4.3 Quartets

A possible approach to determining a phylogenetic tree for a large number of taxa is to try to determine trees for smaller subsets of taxa, and then piece these together. For this, we'll use unrooted trees, since the trees relating these smaller sets might not all contain a common vertex.

If we focus on small subsets, then *quartets* of 4 taxa are the smallest ones we should consider, as an unrooted tree relating 3 taxa gives us no information on a tree topology.

Definition. A *quartet tree* is a unrooted binary tree with 4 labeled leaves. We denote the tree by $ab|cd$ if it induces the split $\{a,b\}|\{c,d\}$.

Now any phylogenetic X-tree T induces a collection $\mathcal{Q}(T)$ of quartet trees:

$$\mathcal{Q}(T) = \{ab|cd \ : \ \text{for some } X_0|X_1 \text{ displayed by } T,\ a,b \in X_0 \text{ and } c,d \in X_1\}.$$

If T is binary, then for every 4-element subset $\{a,b,c,d\}$ of X, exactly one of the quartets $ab|cd$, $ac|bd$, or $ad|bc$ is in $\mathcal{Q}(T)$, so $\mathcal{Q}(T)$ has $\binom{|X|}{4}$ elements. For non-binary $|X|$, of course, it has fewer.

If using quartets to deduce phylogenies of larger collections of taxa is to have a chance of working, we must have the following.

Theorem 12. The collection $\mathcal{Q}(T)$ determines T for a binary tree.

Proof. If $|X| \leq 4$ the result is clear, and so we proceed by induction.

If $X = \{a_1, \ldots, a_n\}$, let $\mathcal{Q}'$ be those quartets in $\mathcal{Q}(T)$ that only involve taxa in $X' = \{a_1, a_2, \ldots, a_{n-1}\}$. Then $\mathcal{Q}' = \mathcal{Q}(T')$ for a tree T' obtained from T in the obvious way, by deleting a_n and its incident edge, and conjoining the two other edges that edge meets, to get a binary tree. Thus $\mathcal{Q}'$ determines T' by the induction hypothesis.

To determine T, we need only determine the conjoined edge of T' to which the edge leading to a_n should attach. So considering each edge e of T' in turn, suppose e induces the split $X'_0|X'_1$ of X'. Assuming this split is non-trivial, so

each of the sets in the split has at least two elements, then if for all $a, b \in X_0'$ and $c, d \in X_1'$ both $ab|ca_n$ and $aa_n|cd$ are in $\mathcal{Q}(T)$, e is the edge we seek. If the split $X_0'|X_1'$ is trivial, then whether an edge leading to a_n shold be attached along e can be determined by a similar test, the formulation of which we leave as Exercise 13. □

This theorem can be extended to non-binary trees as well.

As you'll see in the exercises, a subset of $\mathcal{Q}(T)$ may be enough to identify T. In fact only $|X|-3$ well-chosen quartets are needed, though not every collection of quartets of that size is sufficient, and precisely what quartets are needed depends on the tree.

Of course the real difficulty with determining a phylogenetic tree from quartets is usually not the issue of having insufficiently many quartets. Rather, whatever inference method is used to infer quartets is likely to give some wrong results, and so produce a set of quartets that are incompatible. What is needed is a construction of a tree that somehow reconciles as much information as possible from inconsistent sources. Such *quartet methods* for tree construction have been explored in a number of research papers in recent years.

4.4 Supertrees

Assembling a tree from quartets is an example of a more general problem of constructing a *supertree*. We might have obtained a collection of different trees, with overlapping but different sets of taxa, which we wish to combine into a single tree. Doing this might enable us to see relationships between species which do not appear together on any of the individual trees.

Though there are a number of different approaches, here we describe only one, called Matrix Representation with Parsimony (MRP).

First, we need to extend the idea of parsimony to encompass characters with missing state information.

Suppose we only know the states some character assigns to *some* of the taxa in X. For the remaining taxa, we have no information on what the state is, though we believe there is some state. Then in a parsimony analysis we can treat these taxa for which we have no information as we treat internal vertices on the tree. We consider extensions of the character from the subset of taxa with known states to all internal vertices and taxa with unknown states. The standard definitions of parsimony scores can then be used to choose the most parsimonious tree(s).

If the Fitch-Hartigan algorithm is used to compute parsimony scores for trees, then only a simple modification needs to be made to deal with missing information. If the character has state space S, then to any taxon with missing information we assign the full set S. Taxa for which the state is known are assigned that state as usual. Then one works up the tree from the leaves to the root, assigning sets of states to each vertex and counting changes using the

same rules as when no information is missing. (A similar modification can be made for the Sankoff algorithm. See Exercise 22 of section 3.8.)

Now given a collection of trees $\{T_i\}$, where T_i is a phylogenetic X_i-tree and the taxon sets X_i may be different but overlapping, let $X = \cup X_i$. For each split displayed on each T_i, define a two-state character on X_i by assigning states in accord with the partition sets. For instance, if an edge e of T_i induces a split $X_{i,1}|X_{i,2}$ of X_i, we might define the character $\chi_{T_i,e}$ to assign state 1 to those taxa in $X_{i,1}$ and the state 0 to those in $X_{i,2}$. For any other taxa in X (*i.e.*, for those not on T_i), we treat the character value as missing. The MRP supertree is then the maximum parsimony tree for this character sequence encoding all the splits on all the T_i. This method thus seeks to find trees in which each split on each input tree is displayed (or come close to being displayed) when one ignores the taxa not on that input tree.

Of course as with any use of parsimony, the MRP tree may not be unique. If there are multiple trees that are most parsimonious, it is common to take a consensus tree of those.

The terminology 'Matrix Representation with Parsimony' may seem natural to a biologist, but strange to a mathematician. It comes from the use of the word 'matrix' to describe character information that has been placed in a table, with rows corresponding to taxa, columns to characters, and entries specifying states, and has little to do with a mathematicians notion of a matrix. In MRP, the trees we wish to combine are 'represented' by a matrix encoding characters corresponding to their splits.

4.5 Final Comments

All the concepts and methods discussed in this section have been combinatorial in nature. Even though the problem of reconciling different trees, or summarizing them in a single tree may arise naturally in a biological study, we have used no biological ideas or models in developing consensus and supertree methods. This should be kept firmly in mind when these methods are used. They may be reasonable in some circumstances, but not in others.

As an example, the gene tree/species tree problem of using many gene trees to infer a single species tree is one that can be approached with a biologically-motivated model. In that setting the performance of various consensus methods can be studied, and detailed results on the extent to which various consensus methods are reliable can be obtained. We'll return to this in a later chapter.

For now though, we have given no justification that in any particular biological setting it makes good sense to construct a consensus tree. Even though it may be common to report a consensus tree when parsimony leads to multiple most-parsimonious trees, one may question how this procedure actually fits with the parsimony criterion: a consensus tree of several most-parsimonious trees may in fact require *more* state changes, and thus not be most-parsimonious

itself. This does not mean that such a consensus tree should not be considered, but just that one should understand the limitations.

Finally, Bryant [Bry03] gives an excellent survey on consensus trees, which includes many more types than have been presented here. More material on supertrees can be found in [SS03].

4.6 Exercises

1. List all the splits displayed on the tree in Figure 4.1, including trivial ones.
2. Explain why a set of n taxa will have $2^{n-1} - 1$ splits.
3. Explain why a trivial split of X is compatible with any other split.
4. Show that if distinct splits $Y_0|Y_1$ and $Y_0'|Y_1'$ of a set Y are compatible then there is exactly one pair i, j with $Y_i \cap Y_j' = \emptyset$.
5. Show that two splits $Y_0|Y_1$ and $Y_0'|Y_1'$ of a set Y are compatible if and only if $Y_i \subseteq Y_j'$ for some i, j. Further show that $Y_i \subseteq Y_j'$ if and only if $Y_{1-i} \supseteq Y_{1-j}'$.
6. Problem Deleted.
7. Complete the proof of Theorem 9 by showing a tree T which displays exactly those splits in some collection S is in fact unique. (Hint: Use induction on the number of splits. Assume there are two such trees, and 'contract' the edge in each for one specific split.)
8. Elaborate on the given proof of Theorem 9 to explain why the T we construct has the X-tree property that all vertices of degree ≤ 2 are labeled.
9. For $X = \{a, b, c, d, e, f, g\}$ consider the pairwise compatible splits

$$a|bcdefg, eg|abcdf, b|acdefg, af|bcdeg, f|abcdeg.$$

   a. By Tree Popping, find an X-tree inducing precisely these splits.

   b. Use Tree Popping again, but with the splits in some other order, to again find the same tree.

10. List all the splits displayed on the two trees on the left of Figure 4.3, and then use Tree Popping to produce the tree on the right from them.
11. If T_1, T_2 are X-trees with T_2 a refinement of T_1, write $T_1 \leq T_2$. Show that '$\leq$' is a partial order on the set of X-trees. Then characterize X-trees that are maximal with respect to '$\leq$'.
12. List all 15 quartets displayed on the tree of Figure 4.1.
13. Complete the proof of Theorem 12, by addressing the issue in its final line.

14. A collection of quartets is said to be *compatible* if there is some tree T that displays them all.

    (a) Show that the quartets $ab|cd$ and $ac|be$ are compatible.

    (b) Explain why the quartets $ab|cd$, $ac|be$, and $bc|de$ are not compatible. (Note: every pair of these are compatible, so pairwise compatibility of quartets is not enough to show they can all be displayed on a single tree.)

15. For a 5-leaf binary phylogenetic X-tree T, $\mathcal{Q}(T)$ has 5 elements. Draw such a 5-leaf tree and give a set of only two of these quartets that still determine T. Give another set of two of these quartets that does not determine T.

16. Generalize the result of the last problem by showing that for any binary phylogenetic X-tree, there are $|X| - 3$ quartets that determine T.

17. Show that the definition of compatibility of clades given in the text is consistent with that for compatibility of splits, using the relationship of n-taxon rooted trees to $(n+1)$-taxon unrooted ones. (You may find it helpful to use the result in Exercise 4.)

18. Consider the three trees $(a,(b,(c,d)))$, $((a,b),(c,d)))$, and $(((a,b),c),d)))$ on taxa $X = \{a,b,c,d\}$.

    (a) Construct the strict consensus tree, treating these as rooted.

    (b) Construct the strict consensus tree, treating these as unrooted 4-taxon trees. Is your answer the same as the unrooted version of the tree from (a)?

    (c) Construct the majority-rule consensus tree, treating these as rooted.

19. Explain why the greedy consensus tree is a refinement of the loose consensus tree.

20. If an unrooted binary tree relates n taxa, how many quartets will it display? How many possible quartets (not necessarily compatible) are there for n taxa?

Chapter 5

Distance Methods

The next class of methods for inferring phylogenetic trees that we will discuss are *distance methods*.

Roughly put, a *distance* is some numerical measure of similarity of two taxa. A distance of 0 should indicate the two taxa are the same, and a larger number signifies the degree of their 'differentness.' Although a distance could be based on something other than a comparison of genetic sequences, in our examples that will be its origin. Given a collection of taxa, we somehow compute distances between each pair. We then attempt to find a metric tree so that distances between taxa on it matches our collection of distances. For a variety of reasons, it is seldom possible to do this exactly, so a key issue is how we deal with inexact fit.

We primarily focus on fast algorithmic approaches to using distances to build a single tree, though we will mention some other approaches that instead choose a 'best' tree according to some criterion.

5.1 Dissimilarity Measures

Before beginning, we need to clarify our terminology. In the introductory paragraph above, we actually used the word 'distance' in two different ways. First, it was the measure of differentness between taxa, and second it was a measurement along branches of a metric tree that described the relationships of the taxa. The first of these should be thought of as coming from data, and the second from some processing of the data that yields an inferred metric tree. When necessary to avoid confusion between these two distinct meanings, we'll call the first a *dissimilarity*, and the second a *tree metric* (as defined in Section 2.3). However it is very common for the word 'distance' to be used for both, so we will also use that term when it either seems unlikely to be confusing, or its use in some special terminology is standard practice.

Definition. A *dissimilarity* map for the set X of taxa is a function $\delta : X \times X \to \mathbb{R}$ such that $\delta(x, x) = 0$ and $\delta(x, y) = \delta(y, x)$.

In our applications, we'll always use dissimilarities with non-negative values: $\delta(x, y) \geq 0$.

If we already knew a metric tree relating the taxa in X, then the tree metric arising from it as defined in Chapter 2 would give a dissimilarity measure on X. Indeed, Proposition 5 shows it has the required properties, and additional ones as well.

But since our goal is to find such a tree, we instead look for a dissimilarity measure that can be calculated from data. The simplest such natural one is the *Hamming metric*, which simply counts the proportion of sites that are different in two sequences. If we are given a sequence of characters $\chi_1, \chi_2, \ldots, \chi_n$ on X, set

$$\delta(x, y) = \frac{1}{n} \sum_{i=1}^{n} \delta_{\chi_i(x), \chi_i(y)},$$

where

$$\delta_{a,b} = \begin{cases} 0 & \text{if } a = b \\ 1 & \text{if } a \neq b \end{cases}.$$

For instance, given sequences such as

$$x : \texttt{AACTAGATCCTGTATCGA}$$
$$y : \texttt{ACCTAGGTCCTGTTTCGC}$$

we count 4 differences among 18 sites, so $\delta(x, y) = 4/18$. In biological literature, the Hamming metric is more usually called the *p-distance* (because its value is often denoted by p) or the *uncorrected distance*. While it is a metric by the mathematical definition of that term, it usually is not a tree metric as defined in Section 2.3.

Of course one can easily imagine variations on the Hamming metric, for instance that might assign different weightings to transitions or transversions. We won't pursue such possibilities, though, since in a subsequent chapter we'll construct much more sophisticated dissimilarity maps using a probabilistic model of molecular evolution. We introduce this particular dissimilarity map here only to provide a concrete example to keep in mind as we explore how a dissimilarity can lead to trees.

Dissimilarities can be defined in very different ways than by comparing sites in sequences. For instance, before sequencing was affordable, DNA hybridization was sometimes used to measure dissimilarity. DNA from two taxa was heated, mixed, and cooled, so that the two strands of the helices would separate, and then combine with similar strands from the other taxon, to form less stable molecules due to mismatches in base pairings. The melting temperature (on an appropriate scale) of the hybridized DNA could then be used as a measure of dissimilarity. One could also imagine using measures of dissimilarity that have nothing to do with DNA directly, but capture other features of the taxa.

We'd like to think of dissimilarity values as representing at least approximate distances between taxa along some metric tree, even if we don't think they will

exactly match any tree metric. That is, we hope there is some metric tree (T, w), with positive edge lengths and tree metric d, so that the dissimilarity map δ has values close to the restriction of the tree metric d when applied to pairs of taxa. Of course there's nothing in our definition of a dissimilarity map to ensure this, or even to suggest that the dissimilarity values are even close to such distances measured along a metric tree. For now, we simply hope for the best, and forge ahead.

Suppose then that we have 4 taxa, and using some dissimilarity map we've computed a table of dissimilarities such as:

	S1	S2	S3	S4
S1		.45	.27	.53
S2			.40	.50
S3				.62

Table 5.1: Dissimilarities between taxa

How might we find a metric tree (T, w) for which these data are at least approximately the same as distances computed by the tree metric? More informally, if we know how far apart we want every pair of leaves to be, how can we come up with a tree topology and edge lengths to roughly ensure that?

If there were only 2 taxa, and so one dissimilarity value, this would be simple; draw a one edge unrooted tree and make the edge length match the dissimilarity value.

With 3 taxa, the situation is not much harder: draw the only possible binary unrooted tree relating the taxa. Then with the edge lengths called x, y, z, the 3 leaf-to-leaf distances on the tree are $x+y$, $y+z$, and $x+z$. If we set these equal to the three dissimilarity values, we have 3 linear equations in 3 unknowns, and can easily solve for x, y, and z.

However, if there are 4 taxa, this approach encounters two problems: First, we have 3 different unrooted trees to consider, and since we don't know which one is going to be best, we have to try them all. Second, there are 5 edge lengths to be determined on each of these trees. Since there are 6 dissimilarity values, if we attempt to solve equations to find the edge lengths, we obtain 6 linear equations in 5 unknowns. Such a system typically has no solution, indicating that there is no tree metric that will exactly match the dissimilarity values.

With larger sets of taxa, the problem only grows worse. There are more and more trees to consider, and we find we always have more equations than unknowns. Although computing a least-squares approximate solution would be a possible way of dealing with the overdetermined systems of equations, having to consider all trees will still be as difficult as it was with parsimony.

5.2 An Algorithmic Construction: UPGMA

Instead of trying to consider all possible trees, an alternative is to use a dissimilarity measure to *build* a tree. That is, by considering the dissimilarity values, we guess or infer certain features of the tree, trying to then combine these features until we reach a single tree. In this process we think of the dissimilarities as approximations of an unknown tree metric. We use the numbers in the dissimilarity table to assign plausible edge lengths on the metric tree we build.

When pressed to follow this outline, most students come up with some variant of an approach that is called the *average distance method*, or, more formally, the *unweighted pair-group method with arithmetic means (UPGMA)*. Rather than present the algorithm formally, we develop it through the example data in Table 5.1 above.

The first natural step is to assume that the two taxa which are shown as closest by the dissimilarity map are probably closest in the tree. With the data table above, we pick the two closest taxa, S1 and S3, and join them to a common ancestral vertex by edges. Drawing Figure 5.1, since S1 and S3 should be .27 apart, we decide to split this, making each edge $.27/2 = .135$ long.

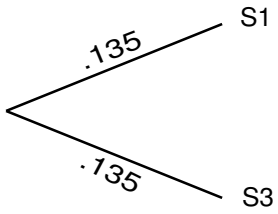

Figure 5.1: UPGMA, step 1

Since S1 and S3 have been joined, we now collapse them into a single combined group S1-S3. To say how far this group is from another taxon, we simply average the distances from S1 and S3 to that taxon. For example, the distance between S1-S3 and S2 is $(.45 + .40)/2 = .425$, and the distance between S1-S3 and S4 is $(.53 + .62)/2 = .575$. Our dissimilarity table thus collapses to Table 5.2.

	S1-S3	S2	S4
S1-S3		.425	.575
S2			.50

Table 5.2: Distances between groups; UPGMA, step 1

Now we simply repeat the process, using the distances in the collapsed table. Since the closest taxa and/or groups in the new table are S1-S3 and S2, which are .425 apart, we draw Figure 5.2.

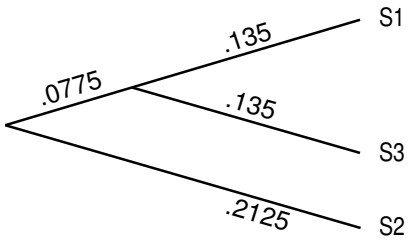

Figure 5.2: UPGMA; step 2

Note the edge to S2 must have length .425/2 = .2125. However the other new edge must have length (.425/2) − .135 = .0775, since we already have the edges of length .135 to account for some of the distance between S2 and the other taxa.

Again combining taxa, we form a group S1-S2-S3, and compute its distance from S4 by averaging *the original distances from S4 to each of S1, S2, and S3.* This gives us (.53+.5+.62)/3 = .55. Note that this is *not* the same as averaging the distance from S4 to S1-S3 and to S2. That average would downweight the contribution from S1 and S3, and thus not treat all our dissimilarities in the same way.

Since a new collapsed distance table would have only the one entry 0.55, there's no need to give it. We draw Figure 5.3, estimating that S4 is .55/2 = .275 from the root. The final edge has length .0625, since that places the other taxa .275 from the root as well.

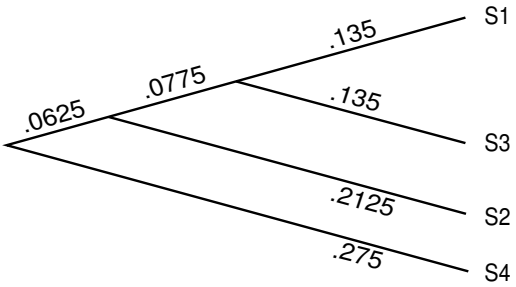

Figure 5.3: UPGMA; step 3

As suspected, the tree in Figure 5.3 constructed from the dissimilarity data does not exactly fit it. The tree metric from S3 to S4, for instance, gives .55, while according to the dissimilarity it should be .62. Nonetheless, the tree metric distances are at least fairly close to the dissimilarity distances.

If we had more taxa to relate, we'd have to do more steps to perform in the UPGMA algorithm, but there would be no new ideas involved. At each step, we join the two closest taxa or groups together, always placing them equidistant

from a common ancestor. We then collapse the joined taxa or groups into a single group, using averaging to compute a distance from that new group to the taxa and groups still to be joined. The one point to be particularly careful about is that when the distances between two groups are computed, we average *all the original distances* from members of one group to another — if one group has n members and another has m members, we have to average nm distances. Each step of the algorithm reduces the size of the distance table by one group/taxon, so that after enough steps, all of the taxa are joined into a single tree.

A few comments on the algorithm are in order here.

First, UPGMA requires little work to give us a tree — at least in comparison to using a parsimony approach where we search through all possible trees, performing an algorithm on each. UPGMA requires searching only for the smallest entry in a succession of tables, and some rather simple algebraic computations. This can be viewed as a strength, as the algorithm is very fast[1], and we get a single answer. But it also can be viewed as a weakness, since it is not completely clear why we should consider the output tree 'good.' We have not explicitly formulated a criterion for judging trees and then found the tree that is optimal. Instead we've designed a process that uses many small steps, each of which may be reasonable on its own, to simply give us a tree.

Second, UPGMA implicitly assumes that we are seeking a rooted ultrametric tree. In this example, when we placed S1 and S3 at the ends of equal length branches, we assumed that the amount of mutation each underwent from their common ancestor was equal. UPGMA *always* places *all* the taxa at the same distance from the root. While this feature of UPGMA might be desirable if we believe a molecular clock underlies our data, in other situations it could be problematic.

5.3 Unequal Branch Lengths

It is not always desirable to impose a molecular clock hypothesis, as use of UPGMA requires. One way of dealing with this arose in a suggested algorithm of Fitch and Margoliash, which builds on the basic approach of UPGMA, but attempts to drop the molecular clock assumption through an additional step. Though the Fitch-Margoliash algorithm is probably never used in current data analysis, understanding it is useful for developing ideas.

Before giving the algorithm, we make a few mathematical observations. First, if we attempt to put 3 taxa on an unrooted tree, then there is only one topology that needs to be considered. Furthermore, for 3 taxa we can assign lengths to the edges to *exactly* fit any given dissimilarities (provided we possibly accept negative lengths). To see this consider the tree in Figure 5.4. If we have

[1]For n taxa, UPGMA takes $\mathcal{O}(n^3)$ steps, though clever programming can reduce this somewhat.

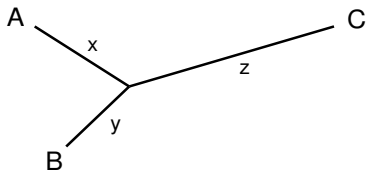

Figure 5.4: The unrooted 3-taxon tree

some dissimilarity data δ_{AB}, δ_{AC}, and δ_{BC}, then

$$\begin{array}{rcrcrcl} x & + & y & & & = & \delta_{AB}, \\ x & + & & & z & = & \delta_{AC}, \\ & & y & + & z & = & \delta_{BC}. \end{array} \tag{5.1}$$

Solving these equations leads to

$$\begin{aligned} x &= (\delta_{AB} + \delta_{AC} - \delta_{BC})/2, \\ y &= (\delta_{AB} + \delta_{BC} - \delta_{AC})/2, \\ z &= (\delta_{AC} + \delta_{BC} - \delta_{AB})/2. \end{aligned} \tag{5.2}$$

We'll refer to the formulas in equations (5.2) as the *three-point formulas* for fitting taxa to a tree. Unfortunately, with more than 3 taxa, exactly fitting dissimilarities to a tree is usually not possible (see Exercise 8). However, the Fitch-Margoliash algorithm uses the 3-taxon case to handle more taxa.

Now we explain the operation of the algorithm with an example. We'll use the dissimilarity data in Table 5.3.

	S1	S2	S3	S4	S5
S1		.31	1.01	.75	1.03
S2			1.00	.69	.90
S3				.61	.42
S4					.37

Table 5.3: Dissimilarities between taxa

We begin by choosing the closest pair of taxa to join, just as we did with UPGMA. Looking at our distance table, S1 and S2 are the first pair to join. In order to join them *without* placing them at an equal distance from a common ancestor, we temporarily reduce to the 3 taxa case by combining *all other* taxa into a group. For our data, we thus introduce the group S3-S4-S5. We find the distance from each of S1 and S2 to the group by averaging their distances to each group member. The distance from S1 to S3-S4-S5 is thus $d(\text{S1}, \text{S3-S4-S5}) = (1.01 + .75 + 1.03)/3 = .93$, while the distance from S2 to

	S1	S2	S3-S4-S5
S1		.31	.93
S2			.863

Table 5.4: Distances between groups; FM algorithm, step 1a

S3-S4-S5 is $d(\text{S2}, \text{S3-S4-S5}) = (1.00 + .69 + .90)/3 = .863$. This gives us Table 5.4.

With only three taxa in this table, we can exactly fit the data to the tree using the three-point formulas to get Figure 5.5. The key point here is that the

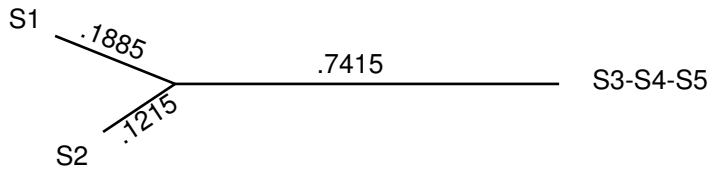

Figure 5.5: FM Algorithm; step 1

three-point formulas, unlike UPGMA, can produce unequal distances of taxa from a common ancestor.

We now keep only the edges ending at S1 and S2 in Figure 5.5, and return to our original data. Remember, the group S3-S4-S5 was only needed temporarily so we could use the three-point formulas; we didn't intend to join those taxa together yet. Since we have joined S1 and S2, however, we combine them into a group for the rest of the algorithm, just as we would have done with UPGMA. This gives us Table 5.5.

	S1-S2	S3	S4	S5
S1-S2		1.005	.72	.965
S3			.61	.42
S4				.37

Table 5.5: Distances between groups; FM algorithm, step 1b

We again look for the closest pair (now S4 and S5), and join them in a similar manner. We combine everything but S4 and S5 into a single temporary group S1-S2-S3 and compute $d(\text{S4}, \text{S1-S2-S3}) = (.75 + .69 + .61)/3 = .683$ and $d(\text{S5}, \text{S1-S2-S3}) = (1.03 + .90 + .42)/3 = .783$. This gives us Table 5.6. Applying the three-point formulas to Table 5.6 produces Figure 5.6.

We keep the edges joining S4 and S5 in Figure 5.6, discarding the edge leading to the temporary group S1-S2-S3. Thus we now have two joined groups, S1-S2 and S4-S5. To compute a new table containing these two groups we've

	S1-S2-S3	S4	S5
S1-S2-S3		.683	.783
S4			.37

Table 5.6: Distances between groups; FM algorithm, step 2a

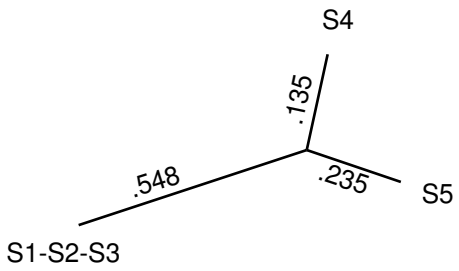

Figure 5.6: FM Algorithm; step 2

found, we average $d(\text{S1-S2}, \text{S4-S5}) = (.75 + 1.03 + .69 + .90)/4 = .8425$ and $d(\text{S3}, \text{S4-S5}) = (.61 + .42)/2 = .515$. We've already computed $d(\text{S1-S2}, \text{S3})$ so we produce Table 5.7. At this point we can fit a tree exactly to the table by a

	S1-S2	S3	S4-S5
S1-S2		1.005	.8425
S3			.515

Table 5.7: Distances between groups; FM algorithm, step 2b

final application of the three-point formulas, yielding Figure 5.7.

Now we replace the groups in this last diagram with the branching patterns we've already found for them. This gives Figure 5.8.

Our final step is to fill in the remaining lengths a and b, using the lengths in Figure 5.7. Since S1 and S2 are on average $(.1885 + .1215)/2 = .155$ from the vertex joining them and S4 and S5 are on average $(.135 + .235)/2 = .185$ from the vertex joining them, we compute $a = .66625 - .155 = .51125$ and $b = .17625 - .185 = -.00875$ to assign lengths to the remaining sides.

Notice that one edge has turned out to have negative length. Since that can't really be meaningful, many practitioners would choose to simply reassign the length as 0. If this happens, however, we should at least check that the negative length was close to 0. If not, we should doubt that our data really are described well by the tree we produced.

A dissappointing feature of the Fitch-Margoliash algorithm is that from any dissimilarity data it always produces the same unrooted topological tree as UPGMA. The reason for this is that each time we decide which taxa or groups

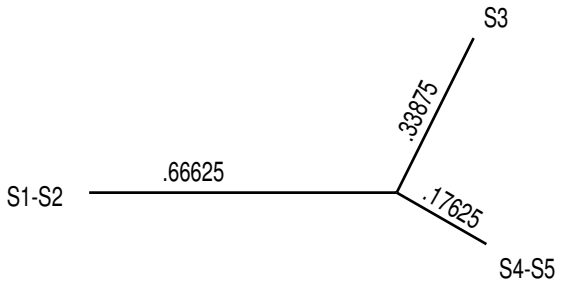

Figure 5.7: FM Algorithm; step 3

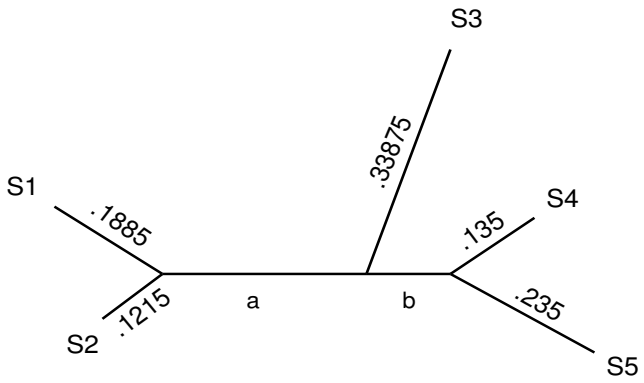

Figure 5.8: FM Algorithm; completion

to join, both methods consider exactly the same collapsed data table and both choose the pair corresponding to the smallest entry in the table. It is therefore only the metric features of the resulting trees that will differ.

To be fair, Fitch and Margoliash actually proposed their algorithm not as an end in itself, but a heuristic method for producing a tree likely to have a certain optimality property (see Exercise 10). We are viewing it here as a step toward the Neighbor Joining algorithm which will be introduced shortly. Familiarity with UPGMA and the Fitch-Margoliash algorithm will aid us in understanding that more elaborate method.

5.4 The Four-point Condition

If we are interested in potentially non-ultrametric trees, as is often the case because of the implausibility of a molecular clock hypothesis, there is a fundamental flaw in using UPGMA to construct trees. Moreover, this flaw is not corrected by the use of the three-point formulas alone.

To better understand the problem, consider the metric quartet tree in Figure

5.9, which we imagine representing the true relationships between the taxa. Here x and y represent specific lengths, with x much smaller than y. Then perfect dissimilarity data (*i.e.*, dissimilarities determined by the tree metric) would give us the distances in Table 5.8.

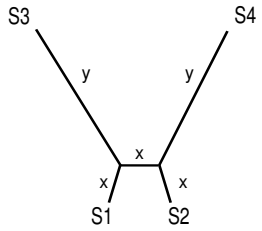

Figure 5.9: The true relationships of taxa S1, S2, S3, S4

	S1	S2	S3	S4
S1		$3x$	$x+y$	$2x+y$
S2			$2x+y$	$x+y$
S3				$x+2y$

Table 5.8: Distances between taxa in Figure 5.9

But if y is much bigger than x, (in fact, $y > 2x$ is good enough) then the closest taxa by distance are S1 and S2. Now the first step of UPGMA will be to look for the smallest dissimilarity in the table, and join those two taxa. This means we'll choose $S1$ and $S2$ as the most closely related, and relate them by a tree that already has a topological mistake. No matter what we do to compute edge lengths, or how the subsequent steps proceed, we will not recover the original tree topology.

The essential problem here is a conflict between closeness of taxa as measured by the tree metric and closeness in the graph theoretic sense as measured by the number of edges on the path connecting them. These are very different notions. It is only reasonable to expect the first to be obvious from data, while the second is the one more relevant to determining the topology of the tree. This is the issue we need to explore theoretically, so that in the next section we can give a practical algorithm to address the problem.

Definition. Two leaves of a tree that are graph-theoretic distance 2 apart are said to form a *cherry*, or are said to be *neighbors*.

Focusing on quartet trees for simplicity, the problem with UPGMA is that it can incorrectly identify a cherry. Once this first cherry is chosen, the full topological quartet tree is determined. So to improve on UPGMA, the key insight is to that we must find a better way to pick a cherry.

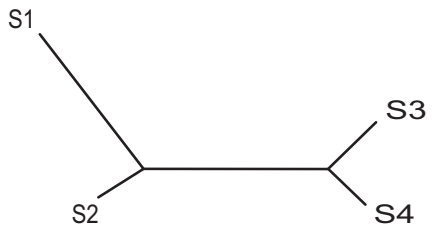

Figure 5.10: A quartet tree with cherries S1, S2 and S3, S4

Consider the quartet tree in Figure 5.10, viewed as a metric tree with positive edge lengths. Letting $d_{ij} = d(\mathrm{S}i, \mathrm{S}j)$ denote the tree metric between leaves, we see that

$$d_{12} + d_{34} < d_{13} + d_{24},$$

since the quantity on the left includes only the lengths of the four edges leading from the leaves of the tree, while the quantity on the right includes all of those and, in addition, twice the central edge length. Notice also that

$$d_{13} + d_{24} = d_{14} + d_{23}$$

by similar reasoning.

On the other hand, if we consider a different quartet tree, $((\mathrm{S1}, \mathrm{S3}), (\mathrm{S2}, \mathrm{S4}))$, we find

$$d_{13} + d_{24} < d_{12} + d_{34} = d_{14} + d_{23}.$$

Notice the inequalities and equalities in this statement are all incompatible with those for the first tree. For instance $d_{13} + d_{24} < d_{12} + d_{34}$ for $((\mathrm{S1}, \mathrm{S3}), (\mathrm{S2}, \mathrm{S4}))$, while $d_{13} + d_{24} > d_{12} + d_{34}$ for $((\mathrm{S1}, \mathrm{S2}), (\mathrm{S3}, \mathrm{S4}))$. These simple observations lead to a way of using metric information to identify cherries. We summarize this as a theorem.

Theorem 13. A metric quartet tree relating taxa a, b, c, d which has positive edge lengths has cherries a, b and c, d if, and only if, any (and hence all) of the three inequalities/equalities in the tree metric that follow hold:

$$d(a, b) + d(c, d) < d(a, c) + d(b, d) = d(a, d) + d(b, c).$$

It should already be clear this observation will be useful, since identifying quartet trees enables us to identify larger tree topologies: For a positive-edge-length binary phylogenetic X-tree T, we get induced positive-edge-length trees for every 4-element subset of X, and hence we can identify the appropriate quartet tree for each such subset. Then, by results in Chapter 4, these quartet trees determine the topology of the tree T.

However, we can do even better in this vein, giving a condition on a dissimilarity map that enables us to relate it to a tree metric.

Definition. A dissimilarity map δ on X satisfies the *four-point condition* if for every choice of four taxa $x, y, z, w \in X$ (including non-distinct ones),

$$\delta(x,y) + \delta(z,w) \leq \max\{\delta(x,z) + \delta(y,w), \delta(x,w) + \delta(y,z)\}. \tag{5.3}$$

Note the non-strict inequality in this definition allows us to formulate the following for trees that are not necessarily binary.

Theorem 14. Given any metric tree with positive edge lengths, its tree metric is a dissimilarity map that satisfies the four-point condition.

Proof. We leave the proof as Exercise 17. □

The converse to this theorem is more remarkable.

Theorem 15. Suppose $|X| \geq 3$, and δ is a dissimilarity on X with $\delta(x,y) \neq 0$ whenever $x \neq y$. Then if δ satisfies the four-point condition, there is a unique metric X-tree with positive edge lengths whose tree metric agrees with δ.

Proof. The case of $|X| = 3$ follows by taking $z = w$ in the 4-point condition as stated above, and applying the three-point formulas with a little care to show edge lengths are positive (see Exercise 18). The case $|X| = 4$ is Exercise 19.

For larger $|X| = N$, we proceed by induction. However, we first need the notion of a *generalized cherry* on an X-tree. For any taxon labeling an internal vertex of the tree, temporarily attach a new edge at that vertex and move the label to its other end, so all labels are now on unique leaves of a phylogenetic X-tree. Then we refer to any cherry on this modified tree as a generalized cherry on the original tree.

Now suppose $X = \{\mathrm{S1}, \mathrm{S2}, \ldots, \mathrm{S}N\}$. Using the inductive hypothesis, let T' be the unique metric X'-tree with positive edge lengths relating $X' = \{\mathrm{S1}, \mathrm{S2}, \ldots, \mathrm{S}(N-1)\}$ whose tree metric restricts to δ on X'. Now choose some generalized cherry on T', and, assume the taxa in the cherry are S1, S2 by renaming them if necessary.

For all $j \neq 1, 2, N$ consider the 4-leaf metric trees on the taxa $\{\mathrm{S1}, \mathrm{S2}, \mathrm{S}j, \mathrm{S}N\}$ whose tree metrics agree with δ. (These trees exists by the 4-taxon case previously shown.) We claim that at least one of the pairs S1, SN, or S2, SN, or S1, S2 forms a generalized cherry in all of these 4-leaf trees. To see why this is so, suppose S1, SN form a generalized cherry in the tree for $\{\mathrm{S1}, \mathrm{S2}, \mathrm{S}k, \mathrm{S}N\}$ for some k. On this 4-leaf tree if the vertex where paths to S1, S2 and SN meet is at a metric distance a from S1, and the vertex where paths from S1, S2, Sk meet is at a distance b from S1, then $a \leq b$. But since S1 and S2 form a generalized cherry in T', the vertex where paths from S1, S2, Sk meet on T' is the same as that where paths from S1, S2, and Sj meet for all $j < N$. Now a can be computed from the distances between S1, S2 and SN, and b from the distances between S1, S2, Sj for any $2 < j < N$, always giving the same value regardless of j, by the inductive hypothesis. But then the inequality $a \leq b$ shows that S1 and SN must form a generalized cherry in the trees for all $\{\mathrm{S1}, \mathrm{S2}, \mathrm{S}j, \mathrm{S}N\}$. Similarly if S2, S$N$ form a generalized cherry for one of the 4-leaf trees, they

form a generalized cherry for all. If there are no 4-leaf trees where either S1, SN or S2, SN form a generalized cherry, then S1, S2 must form a generalized cherry in all.

Using the claim of the last paragraph, by interchanging the names of S1, S2, SN if necessary, we may assume S1, SN always form a generalized cherry for all the 4-leaf trees relating S1, S2, Sj, SN. Again let T' be the tree for {S1, S2, ..., S($N-1$)}. (So now S1 and S2 may not form a generalized cherry in T' due to interchanging taxa names.) We leave to the reader the final step of giving the unique way to 'attach' SN to T' consistent with all dissimilarity values (Exercise 20). □

5.5 The Neighbor Joining Algorithm

In practice, UPGMA with its ultrametric assumption is only used on biological data in very special circumstances. Much preferred is a more elaborate algorithm, called *Neighbor Joining*, which is built on the four-point condition.

However, it is important that Neighbor Joining not require that the four-point condition be *exactly* met for the dissimilarity data to which it is applied, since dissimilarities computed from data should be expected at best to be only roughly consistent with a metric tree. We therefore want to perform various averaging processes as we go along in order to smooth out some of the errors in fit.

To motivate the algorithm, we imagine a binary positive-edge-length tree in which taxa S1 and S2 form a cherry joined at vertex v, with v somehow joined to the remaining taxa S3, S4,..., SN, as in Figure 5.11.

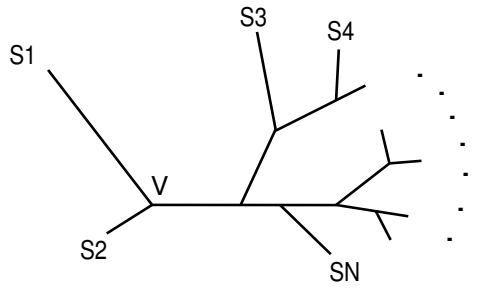

Figure 5.11: Tree with S1 and S2 forming a cherry

If our dissimilarity data agreed exactly with a metric for this tree then for every $i, j = 3, 4, \ldots, N$, we'd find from the four-point condition that

$$d_{12} + d_{ij} < d_{1i} + d_{2j}. \tag{5.4}$$

For fixed i, there are $N - 3$ possible choices of j with $3 \leq j \leq N$ and $j \neq i$. If

we sum the inequalities (5.4) for these j we get

$$(N-3)d_{12} + \sum_{\substack{j=3 \\ j\neq i}}^{N} d_{ij} < (N-3)d_{1i} + \sum_{\substack{j=3 \\ j\neq i}}^{N} d_{2j}. \tag{5.5}$$

To simplify this, define the total dissimilarity between taxon Si and all other taxa as

$$R_i = \sum_{j=1}^{N} d_{ij}.$$

Then adding $d_{i1} + d_{i2} + d_{12}$ to each side of inequality (5.5) allows us to write it in the simpler form

$$(N-2)d_{12} + R_i < (N-2)d_{1i} + R_2.$$

Subtracting $R_1 + R_2 + R_i$ from each side of this then gives a more symmetric statement,

$$(N-2)d_{12} - R_1 - R_2 < (N-2)d_{1i} - R_1 - R_i.$$

If we apply the same argument to Sn and Sm, rather than S1 and S2, we are led to define

$$M_{nm} = (N-2)d_{nm} - R_n - R_m. \tag{5.6}$$

Then if Sn and Sm form a cherry, we'll have that

$$M_{nm} < M_{nk}$$

for all $k \neq m$.

This gives us the criterion used for Neighbor Joining: From the dissimilarity data, compute a new table of values for M_{ij} using equation (5.6). Then choose to join the pair Si, Sj of taxa with the smallest value of M_{ij}.

The argument above shows that if Si and Sj form a cherry in a metric tree producing the dissimilarity data, then the value M_{ij} will be the smallest of the values in the ith row and jth column of the table for M. However, this is not enough to justify the Neighbor Joining criterion. An additional argument is still needed to show the smallest entry in the *entire table* for M truly identifies a cherry. Though this claim is plausible, we outline a proof of it in Exercise 24.

We now describe the full Neighbor Joining algorithm.

Algorithm.

1. Given dissimilarity data for N taxa, compute a new table of values of M using equation (5.6). Choose the smallest value in the table for M to determine which taxa to join. (This value may be, and usually is, negative, so 'smallest' means the negative number with the greatest absolute value.)

2. If Si and Sj are to be joined at a new vertex v, temporarily collapse all other taxa into a single group G, and determine the lengths of the edges from

Si and Sj to v by using using the three-point formulas for Si, Sj, and G as in the algorithm of Fitch and Margoliash.

3. Determine distances/dissimilarities from each of the taxa Sk in G to v by applying the three-point formulas to the distance data for the three taxa Si, Sj and Sk. Now include v in the table of dissimilarity data, and drop Si and Sj.

4. The distance table now includes $N-1$ taxa. If there are only 3 taxa, use the three-point formulas to finish. Otherwise go back to step 1.

Exercise 24 completes the proof of the following theorem, by showing that for dissimilarity data consistent with a tree metric, the Neighbor Joining algorithm really does join taxa to correctly recover the metric tree.

Theorem 16. Suppose a dissimilarity map on X is the restriction of a tree metric for a binary metric phylogenetic X-tree T with all positive edge lengths. Then the Neighbor Joining algorithm will reconstruct T and its edge lengths.

As you can see already, Neighbor Joining is not pleasant to do by hand. Even though each step is relatively straightforward, it's easy to get lost in the process with so much arithmetic to do. In the exercises you'll find an example partially worked that you should complete to be sure you understand the steps. After that, we suggest you use a computer program to avoid mistakes (or, even better, write your own program).

The accuracy of various tree construction methods – the ones outlined so far in these notes and many others – has been tested primarily through simulating DNA mutation according to certain specified models of mutation along phylogenetic trees and then applying the methods to see how often they recover the tree that was the basis for the simulation. These tests have lead researchers to be considerably more confident of the results given by Neighbor Joining than by UPGMA. While UPGMA may be reliable, or even preferred, under some special circumstances, Neighbor Joining works well on a broader range of data. Since it appears that a molecular clock hypothesis is often violated by real data, Neighbor Joining is by far the most commonly used distance method for tree construction in phylogenetics.

5.6 Additional Comments

An important point to remember is that so far we have simply hoped the dissimilarity measure we began with was reasonably close to a tree metric, and so could be lead to an appropriate metric tree. But we have given no argument that the Hamming distance, or any other dissimilarity measure, should be approximately 'tree-like.' In fact, under plausible models for the evolution of DNA sequences the Hamming distance need not be close to tree-like, unless the total amount of mutation between taxa is small. We'll overcome this problem in the next chapter, by developing such models and then using them to introduce corrections.

Although Neighbor Joining and UPGMA lack an explicit criterion for how they determine the 'best' tree to fit data, there are distance methods which are based on such criteria. For instance, the Minimum Evolution approach considers each possible topological tree in turn, use least-squares minimization to determine edge lengths that best fit the dissimilarity data, and then chooses from these metric trees the one with the minimum total of edge lengths. To follow this scheme, however, one is forced to consider all possible topological trees, and this prevents the method from being comparable in speed to the algorithmic approaches discussed here. A variant of this, Balanced Minimum Evolution, introduces weights in the least-squares minimization in a specific way. Neighbor Joining has been shown to be a greedy algorithm to approximate the BME tree, which avoids searching over all trees.

A valid criticism of all distance methods is that they do not use the full information in the data, since they are based on only pairwise comparisons of the taxa. By not comparing all taxa at once, some potential information is lost. It's not too hard to see that it is impossible to reconstruct sequence data, or even a rough description of it, from the collection of pairwise Hamming distances between them. Thus we have lost something by boiling the sequences down so crudely into a few numbers. Indeed, this is probably the main reason distance methods should be viewed as a less than optimal approach, to be used primarily for quick initial explorations of data, or on very large data sets when other methods are too slow to be feasible. Often software for more elaborate inference methods will begin by constructing an initial 'pretty-good' tree by a distance method, before searching for a better tree that is similar to it. However, for a very large number of taxa it may not be possible to effectively search among similar trees in an acceptable amount of time.

Finally, theoretical work on distance methods is continuing. There is a variant of Neighbor Joining, called Bio-NJ, that gains improved performance by taking into account that the dissimilarity between taxa a, b will be statistically correlated with that between c, d if the path from a to b shares some edges with that from c to d. Other recent theoretical work has indicated that if this correlation were more fully exploited, then distance methods could be (in a precise technical sense) similar in effectiveness to more elaborate inference through a maximum likelihood framework.

5.7 Exercises

1. For the tree in Figure 5.3 constructed by UPGMA, compute a table of distances between taxa along the tree. How does this compare to the original dissimilarities of Table 5.1?

2. Suppose four sequences S1, S2, S3, and S4 of DNA are separated by dissimilarities as in Table 5.9. Construct a rooted tree showing the relationships

between S1, S2, S3, and S4 by UPGMA.

	S1	S2	S3	S4
S1		1.2	.9	1.7
S2			1.1	1.9
S3				1.6

Table 5.9: Dissimilarity data for Problems 2 and 5

3. Perform UPGMA on the data in Table 5.3 that was used in the text in the example of the FM algorithm. Does UPGMA produce the same tree as the FM algorithm topologically? metrically?

4. The fact that dissimilarity data relating three taxa can be exactly fit by appropriate edge lengths on the single unrooted topological 3-taxon tree is used in the Neighbor Joining algorithm.

   a. Derive the three-point formulas of Equation 5.1.

   b. If the dissimilarities are $\delta_{AB} = .634$, $\delta_{AC} = 1.327$, and $\delta_{BC} = .851$, what are the lengths x, y, and z?

5. Use the FM algorithm to construct an unrooted tree for the data in Table 5.9 that were also used in Problem 2. How different is the result?

6. A desirable feature of a dissimilarity map on sequences is that it be *additive*, in the sense that if S0 is an ancestor of S1, which is in turn an ancestor of S2, then

$$d(\text{S0}, \text{S2}) = d(\text{S0}, \text{S1}) + d(\text{S1}, \text{S2}).$$

   a. Explain why an additive dissimilarity map is desirable if we are trying to use dissimilarities to construct metric trees.

   b. Give an example of sequences to illustrate that the Hamming dissimilarity might not be additive.

   c. If mutations are rare, why might the Hamming dissimilarity be approximately additive?

7. While any dissimilarity values between 3 taxa can be fit to a metric tree (possibly with negative edge lengths), that's not the case if we want to fit an ultrametric tree.

   a. If the three dissimilarities are 0.3, 0.4, and 0.5, find an unrooted tree that fits them, and explain why no choice of a root location can make this tree ultrametric.

   b. If the three dissimilarities are 0.2, 0.3, 0.3 find an unrooted tree that fits them, and locate a root to make it ultrametric.

c. If the three dissimilarities are 0.3, 0.3, and 0.4, find an unrooted tree that fits them, and explain why no choice of a root location can make this tree ultrametric.

8. While distance data for 3 taxa can be exactly fit to an unrooted tree, if there are 4 (or more) taxa, this is usually not possible.

   a. For the tree $((a,b),(c,d))$, denoting distances between taxa with notation like d_{ab}, write down equations for each of the 6 such distances in terms of the 5 edge lengths. Explain why if you use dissimilarity values in place of the distances these equations are not likely to have an exact solution.

   b. Give a concrete example of 6 dissimilarity values so that the equations in part (a) cannot be solved exactly. Give another example of values where the equations can be solved.

9. Suppose you have a dissimilarity values for n taxa, and wish to see if it could exactly fit a tree metric on a particular binary tree T. How many equations and how many unknowns would be in the resulting system of equations you would need to solve? Show for $n \geq 4$ there are more equations than unknowns in this system, and thus it is unlikely to have a solution.

10. A number of different measures of goodness of fit between dissimilarity data and metric trees have been proposed. Let δ_{ij} denote the dissimilarity between taxa i and j, and e_{ij} denote the tree metric distance from i to j. A few of the these measures are:

$$s_{FM} = \left(\sum_{i,j} \left(\frac{\delta_{ij} - e_{ij}}{\delta_{ij}} \right)^2 \right)^{\frac{1}{2}},$$

$$s_F = \sum_{i,j} |\delta_{ij} - e_{ij}|,$$

$$s_{TNT} = \left(\sum_{i,j} (\delta_{ij} - e_{ij})^2 \right)^{\frac{1}{2}}.$$

   In all these measures, the sums include terms for each distinct pair of taxa, i and j.

   a. Compute these measures for the tree constructed in the text using the FM algorithm, as well as the tree constructed from the same data using UPGMA in Problem 3. According to each of these measures, which of the two trees is a better fit to the data?

   b. Explain why these formulas are reasonable ones to use to measure goodness of fit. Explain how the differences between the formulas make them more or less sensitive to different types of errors.

Note: Fitch and Margoliash proposed choosing the optimal metric tree to fit data as the one that minimized s_{FM}. The FM algorithm was introduced in an attempt to get an approximately optimal tree.

11. Suppose the unrooted metric tree in Figure 5.12 correctly describes the evolution of taxa A, B, C, and D.

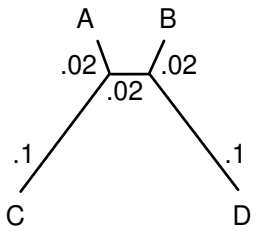

Figure 5.12: Tree for problem 11

a. Explain why, regardless of the location of the root, a molecular clock could not have operated.

b. Give a dissimilarity table by calculating tree metric distances between each pair of the four taxa. Perform UPGMA on that data.

c. UPGMA did not reconstruct the correct tree. Where did it go wrong? What was it about this metric tree that led it astray?

d. Explain why the FM algorithm will also not reconstruct the correct tree.

12. Show that every unrooted binary phylogenetic tree with at least three taxa has at least two cherries. (A unrooted binary tree with n leaves that has only two cherries is sometimes called a *caterpillar*. Despite the unfortunate mixing of metaphors, why is this terminology reasonable?)

13. For the quartet tree $((a, d), (b, c))$ that was not explicitly treated in Section 5.4, write the inequalities and equalities that hold expressing the four-point condition.

14. Show that if a dissimilarity map on X satisfies the four-point condition, then it satisfies the triangle inequality on X. (Rather than deducing this from Theorem 15, show it directly by taking several of the taxa to be the same in the four-point condition.)

15. If a dissimilarity map arises from a tree metric on X, then it is a metric on X. Show that the converse is false by giving an example of a metric on a set X that is not a tree metric.

16. Show that the four-point condition is equivalent to the following statement: For every choice of $x, y, z, w \in X$, of the three quantities

$$\delta(x, y) + \delta(z, w),\ \delta(x, z) + \delta(y, w),\ \delta(x, w) + \delta(y, z)$$

the two (or three) largest are equal.

17. Prove Theorem 14, by first observing that it's enough to prove it for 4-leaf trees. For 4-leaf trees, be sure you consider both binary and non-binary trees, and cases where x, y, z, w are all distinct and when they are not.

18. Prove Theorem 15 for the case $|X| = 3$. (Be sure you show all edge lengths are non-negative by using the four-point condition.)

19. Prove Theorem 15 for the case $|X| = 4$.

20. Give the final step of the proof of Theorem 15.

21. Before working through an example of Neighbor Joining, it's helpful to derive formulas for Steps 2 and 3 of the algorithm. Suppose we've chosen to join Si and Sj in Step 1.

    a. Show that for Step 2, the distances of Si and Sj to the internal vertex v can be computed by

$$d(\mathrm{S}i, v) = \frac{\delta(\mathrm{S}i, \mathrm{S}j)}{2} + \frac{R_i - R_j}{2(N-2)},$$
$$d(\mathrm{S}j, v) = \frac{\delta(\mathrm{S}i, \mathrm{S}j)}{2} + \frac{R_j - R_i}{2(N-2)}.$$

    Then show the second of these formulas can be replaced by

$$d(\mathrm{S}j, v) = \delta(\mathrm{S}i, \mathrm{S}j) - d(\mathrm{S}i, v).$$

    b. Show that for Step 3, the distances of Sk to v, for $k \neq i, j$ can be computed by

$$d(\mathrm{S}k, v) = \frac{\delta(\mathrm{S}i, \mathrm{S}k) + \delta(\mathrm{S}j, \mathrm{S}k) - \delta(\mathrm{S}i, \mathrm{S}j)}{2}.$$

22. Consider the distance data of Table 5.10. Use the Neighbor Joining algo-

	S1	S2	S3	S4
S1		.83	.28	.41
S2			.72	.97
S3				.48

Table 5.10: Taxon distances for Problem 22

    rithm to construct a tree as follows:

    a. Compute R_1, R_2, R_3 and R_4 and then a table of values for M for the taxa S1, S2, S3, and S4. To get you started

$$R_1 = .83 + .28 + .41 = 1.52 \text{ and } R_2 = .83 + .72 + .97 = 2.52$$

so

$$M(\text{S1}, \text{S2}) = (4-2).83 - 1.52 - 2.52 = -2.38.$$

b. If you did part (a) correctly, you should have a tie for the smallest value of M. One of these smallest values is $M(\text{S1}, \text{S4}) = -2.56$, so let's join S1 and S4 first.

For the new vertex v where S1 and S4 join, compute $d(\text{S1}, v)$ and $d(\text{S4}, v)$ by the formulas in part (a) of the previous problem.

c. Compute $d(\text{S2}, v)$ and $d(\text{S3}, v)$ by the formulas in part (b) of the previous problem.

Put your answers into the new distance Table 5.11.

	v	S2	S3
v		—	—
S2			.72

Table 5.11: Group distances for Problem 22

d. Since there are only 3 taxa left, use the three-point formulas to fit v, S2, and S3 to a tree.

e. Draw your final tree by attaching S1 and S4 to v with the distances given in part (b).

23. Consider the distance data in Table 5.12, which is exactly fit by the tree of

	S1	S2	S3	S4
S1		.3	.4	.5
S2			.5	.4
S3				.7

Table 5.12: Taxon distances for Problem 23

Figure 5.9, with $x = .1$ and $y = .3$.

a. Use UPGMA to reconstruct a tree from these data. Is it correct?

b. Use Neighbor Joining to reconstruct a tree from these data. Is it correct?

24. Complete the proof of Theorem 16 by showing that the criterion used by Neighbor Joining to pick cherries from dissimilarity data arising from a positive edge length binary metric tree will pick only true cherries. Do this by following the outline below of the proof of Studier and Keppler.

a. Show $M_{mn} - M_{ij} = \sum_{k \neq i,j,m,n} ((d_{ik} + d_{jk} - d_{ij}) - (d_{mk} + d_{nk} - d_{mn}))$.

Now suppose M_{ij} is minimal but that i and j do not form a cherry.

b. Explain why neither i nor j are in any cherry.

Pick some cherry $\mathrm{S}m, \mathrm{S}n$, and consider the 4-leaf subtree Q of T joining $\mathrm{S}i, \mathrm{S}j, \mathrm{S}m, \mathrm{S}n$ inside T. Denote the internal vertices of it with degree 3 by u (joined to i, j) and v (joined to m, n). For each additional taxon $\mathrm{S}k$, show the following:

c. If the path in T from $\mathrm{S}k$ to Q joins Q at a vertex x along the path from i to j, then $(d_{ik} + d_{jk} - d_{ij}) - (d_{mk} + d_{nk} - d_{mn}) = -2d_{ux} - 2d_{uv}$.

d. If the path in T from $\mathrm{S}k$ to Q joins Q at a vertex x along the path from u to v, then $(d_{ik} + d_{jk} - d_{ij}) - (d_{mk} + d_{nk} - d_{mn}) = -4d_{vx} + 2d_{uv}$.

e. Conclude from the fact that left hand side of the equality in (a) is non-negative that at least as many of the Sk are described by (d) as by (c); and thus there are strictly more leaves that are joined to the path from i to j at u than at any other vertex.

f. Explain why since i and j are not in any cherries there must be a cherry r, s which is joined to the path from i to j at some vertex other than u.

g. Applying the argument of parts (c), (d), (e) to r, s instead of m, n, arrive at a contradiction.

Chapter 6

Probabilistic Models of DNA Mutation

The methods developed so far have not required any detailed mathematical description of the mutation processes DNA undergoes from generation to generation. In fact, the only real consideration we've given to the mutation process is to point out that if we believe mutations are sufficiently rare events, then both parsimony and use of the uncorrected p-distance are justifiable approaches to inferring a tree. Under such circumstances mutations are unlikely to occur multiple times at the same site, so possible hidden mutations would be negligible.

However, if mutations are not so rare, either because of a high mutation rate or because of a long elapsed time, then along any edge of a tree a particular site might experience more than one change. In the idealized situation of observing both the ancestral and descendant sequence, at most one mutation is observable at a site, though several might have occurred. Under these circumstances both the principal of parsimony and the Hamming dissimilarity map would give inappropriate views as to how much mutation has occurred — they both underestimate the true amount of change.

To do better, we need explicit mathematical models describing how base substitutions occur. Since mutations appear to be random events, we formulate these probabilistically. In subsequent chapters these models will be used first to develop improved dissimilarity maps for distance methods, and then as a basis for the primary statistical approaches to phylogenetic inference, Maximum Likelihood and Bayesian analysis.

6.1 A first example

To begin, we present a simple example in order to illustrate the basic modeling approach. We'll keep this discussion as informal as possible, and then be more careful with our terminology later on. For those who have seen Markov models

before, either in a probability or linear algebra course, the framework will be familiar.

Our model will first describe *one site* in a DNA sequence, and how it changes from an ancestral sequence to a descendant sequence along a *single edge* of a tree. To further simplify, we focus not on the precise base at that site, but only on whether a purine `R` or a pyrimidine `Y` appears.

To be concrete about the situation we wish to describe, suppose we somehow had access to data of an ancestral sequence S0 and a descendant sequence S1:

S0 : `RRYRYRYYRYYYRYRYYRRYY`

S1 : `RYYRYYYYRYYYRYRYYRRYR`

To describe an arbitrary site in an ancestral sequence, we simply specify the probabilities that site might be occupied by either an `R` or `Y`. For instance the two numbers

$$(\mathcal{P}(S0 = \texttt{R}), \mathcal{P}(S0 = \texttt{Y})) = (p_R, p_Y) = (.5, .5)$$

would indicates an equal chance of each, while $(p_R, p_Y) = (.6, .4)$ would indicate a greater likelihood of a purine. Since our site must have either a purine or a pyrimidine, note the two probabilities must add to 1. If $(p_R, p_Y) = (.5, .5)$, then we can think of the ancestral base as being determined by the flip of a fair coin. If $(p_R, p_Y) = (.6, .4)$, then we can still think of the ancestral base as being determined by a coin flip, but the coin must be biased so that its '`R`' side lands up in 60% of a large number of tosses.

Although our model is describing only one site in the sequence, we view the data sequences as being many different trials of the same probabilistic process. Thus (p_R, p_Y), the probabilities that a site in an idealized ancestral sequence of infinite length is occupied by an `R` or `Y`, can be estimated by the frequencies at which these occur in the observed sequence. For example, the 21-base sequence S0 above has 9 `R`s and 12 `Y`s, which leads us to estimate $p_R = 9/21$ and $p_Y = 12/21$.

To justify this estimate, we are making what is often called an *i.i.d. assumption*, that each site behaves *independently* with an *identical distribution.* If the sites are independent and identically distributed, we may use the data for them all to infer things about the common process they all undergo.

With the ancestral sequence described, we now focus on probabilities of base substitutions as S0 evolves into S1. There are 4 possibilities here, `R` $\to$ `R`, `R` $\to$ `Y`, `Y` $\to$ `R` and `Y` $\to$ `Y`, which in our data occurred in a total of 7, 2, 1, and 11 sites, respectively. It is most convenient to summarize this through conditional probabilities. For instance, the conditional probability that an ancestral `R` remains an `R` is denoted by

$$\mathcal{P}(\text{S1} = \texttt{R} \mid \text{S0} = \texttt{R}),$$

which we read as 'the probability that we have an `R` in S1 *given that* we had an `R` in S0.' For our data, using the i.i.d. assumption, we estimate

$$\mathcal{P}(\text{S1} = \texttt{R} \mid \text{S0} = \texttt{R}) = 7/9,$$

since of the 9 ancestral Rs, 7 remained Rs. Similarly we estimate

$$\begin{aligned}\mathcal{P}(\text{S1} = \texttt{Y} \mid \text{S0} = \texttt{R}) &= 2/9,\\ \mathcal{P}(\text{S1} = \texttt{R} \mid \text{S0} = \texttt{Y}) &= 1/12,\\ \mathcal{P}(\text{S1} = \texttt{Y} \mid \text{S0} = \texttt{Y}) &= 11/12.\end{aligned}$$

Notice

$$\begin{aligned}\mathcal{P}(\text{S1} = \texttt{R} \mid \text{S0} = \texttt{R}) + \mathcal{P}(\text{S1} = \texttt{Y} \mid \text{S0} = \texttt{R}) &= 1,\\ \mathcal{P}(\text{S1} = \texttt{R} \mid \text{S0} = \texttt{Y}) + \mathcal{P}(\text{S1} = \texttt{Y} \mid \text{S0} = \texttt{Y}) &= 1,\end{aligned}$$

as the total conditional probability of either an R or Y appearing in S1, assuming that the base in S0 is given, must be 1.

Now our model is summarized by six probabilities, which we organize into a row vector and a matrix:

$$\mathbf{p}_0 = \begin{pmatrix} p_R & p_Y \end{pmatrix} = \begin{pmatrix} 9/21 & 12/21 \end{pmatrix},$$

$$\begin{aligned} M &= \begin{pmatrix} \mathcal{P}(\text{S1} = \texttt{R} \mid \text{S0} = \texttt{R}) & \mathcal{P}(\text{S1} = \texttt{Y} \mid \text{S0} = \texttt{R}) \\ \mathcal{P}(\text{S1} = \texttt{R} \mid \text{S0} = \texttt{Y}) & \mathcal{P}(\text{S1} = \texttt{Y} \mid \text{S0} = \texttt{Y}) \end{pmatrix} \\ &= \begin{pmatrix} p_{RR} & p_{RY} \\ p_{YR} & p_{YY} \end{pmatrix} = \begin{pmatrix} 7/9 & 2/9 \\ 1/12 & 11/12 \end{pmatrix}. \end{aligned}$$

(Note the switch in order here for the state indications on the matrix entries, where $\mathcal{P}(\text{S1} = \texttt{R} \mid \text{S0} = \texttt{Y}) = p_{YR}$, for instance.) The rows of the matrix refer to ancestral states, while the columns refer to descendant ones. As a result, the entries across any row add to 1, though the column sums have no special value.

One point of this matrix notation is that the product $\mathbf{p}_0 M$ has meaningful entries:

$$\mathbf{p}_0 M = \begin{pmatrix} p_R & p_Y \end{pmatrix} \begin{pmatrix} p_{RR} & p_{RY} \\ p_{YR} & p_{YY} \end{pmatrix} = \begin{pmatrix} p_R p_{RR} + p_Y p_{YR} & p_R p_{RY} + p_Y p_{YY} \end{pmatrix}$$

where, for instance, the left entry is

$$\begin{aligned} p_R p_{RR} + p_Y p_{YR} &= \mathcal{P}(\text{S1} = R \mid \text{S0} = R)\mathcal{P}(\text{S0} = R) && (6.1)\\ &\quad + \mathcal{P}(\text{S1} = \texttt{R} \mid \text{S0} = \texttt{Y})\mathcal{P}(\text{S0} = \texttt{Y}) && (6.2)\\ &= \mathcal{P}(\text{S1} = \texttt{R}). && (6.3)\end{aligned}$$

Although this last equality follows from various formal multiplication and sum rules of probabilities, it can also be understood intuitively: The term on the right side of (6.1) gives the probability that we have an ancestral R that then remains an R. The term in (6.2) gives the probability that we have an ancestral Y that then changes to an R. Since these are the only ways we could have an R in the descendant site, by adding these, we are computing the probability that the descendant has an R, as claimed in equation (6.3).

Similarly, the right entry of the product $\mathbf{p}_0 M$ is $\mathcal{P}(\text{S1} = \texttt{Y})$. Thus if $\mathbf{p}_1$ denotes the probability distribution of Rs and Ys for S1, we have shown

$$\mathbf{p}_1 = \mathbf{p}_0 M.$$

The matrix M is therefore not just a table to encapsulate the various probabilities of changes in the substitution process between the ancestral and descendant sequences; the multiplication of a vector by M actually 'does' the process.

What, then, might happen if the sequence S1 continues to evolve? Assuming circumstances are similar to those during the evolution of S0 to S1, and the elapsed time is similar, it's reasonable to hypothesize that we would obtain a sequence S2 whose R/Y composition would be described by

$$\mathbf{p}_2 = \mathbf{p}_1 M = \mathbf{p}_0 M^2.$$

Thinking of M as describing one time-step, we now have a *model* of sequence evolution using discrete time, with a descendant sequence after n time steps being described by

$$\mathbf{p}_n = \mathbf{p}_0 M^n.$$

The *parameters* of our model are a vector $\mathbf{p}_0$ of non-negative numbers that sum to 1, which we call the *root distribution vector*, and a matrix M of non-negative numbers whose rows sum to one, which we call a *Markov matrix* (or *stochastic matrix*). The root distribution describes the models starting conditions (the ancestral sequence) while the Markov matrix describes the substitution process in a single time step.

A continuous-time version

In the presentation above, we initially thought of M as describing evolution along a full edge of a tree. Then we shifted to imagining it as describing evolution for just one time-step, and that an edge might be many time steps long. While both of these views are useful in some settings, they essentially use a discrete notion of time.

While mutations do occur at discrete times, corresponding to generations of the evolving organism, this is not always the simplest way to view things. Since the span of a generation is usually quite small relative to evolutionary time scales, it is more common to describe the evolutionary process using a continuous notion of time.

In this formulation, we imagine that there are certain *rates* at which the various types of substitutions occur, and we organize them into a matrix such as

$$Q = \begin{pmatrix} q_{RR} & q_{RY} \\ q_{YR} & q_{YY} \end{pmatrix}.$$

Here, for instance q_{RY} denotes the instantaneous rate at which Rs are replaced by Ys, and would be measured in units like (substitutions at a site)/(unit of time). Moreover $q_{RY} \geq 0$, or more likely $q_{RY} > 0$.

At first it seems nonsensical to say entry q_{RR} should be the rate at which Rs are replaced by Rs. But if some Rs are being replaced by Ys, then some Rs will cease to be Rs. Thus $q_{RR} \leq 0$ gives this rate of loss of Rs. Moreover, we must have $q_{RR} + q_{RY} = 0$ since these two rates must balance.

Considering the second row of Q similarly, we see that the entries in each row of Q must add to give 0 (unlike the 1 of the discrete-time Markov matrix). The off-diagonal entries of Q giving rate of changes from one state to another must be non-negative, so the diagonal entries must be non-positive.

Letting $\mathbf{p}_t = (p_R(t)\ p_Y(t))$ denote the distribution vector of purines and pyrimidines at time t, with $t = 0$ for the ancestral sequence, we have a system of differential equations:

$$\begin{aligned}\frac{d}{dt}p_R(t) &= p_R(t)q_{RR} + p_Y(t)q_{YR}\\ \frac{d}{dt}p_Y(t) &= p_R(t)q_{RY} + p_Y(t)q_{YY}.\end{aligned}$$

Expressing this system in matrix form, by letting $\mathbf{p}_t = (p_R(t), p_Y(t))$, yields

$$\frac{d}{dt}\mathbf{p}_t = \mathbf{p}_t Q. \tag{6.4}$$

(While systems of differential equations such as this are commonly covered in ordinary differential equations courses, they are usually presented using column vectors, with the matrix appearing to the left of the column vector. Our notation is more standard for probabilistic models, and is equivalent, through a matrix transpose.)

Before we sketch the solution of this system of differential equations, for motivation recall a similar differential equation that involves no vectors or matrices:

$$p'(t) = rp(t),\ p(0) = p_0.$$

This equation is often used to model a population, whose initial size is p_0, and which grows at a rate proportional to its size, with r being the constant of proportionality. The solution of this is

$$p(t) = p_0 e^{rt}.$$

Moreover, the exponential function appearing in this solution can be understood by a Taylor series:

$$e^x = 1 + x + \frac{1}{2}x^2 + \frac{1}{3!}x^3 + \frac{1}{4!}x^4 + \ldots.$$

The system of differential equations (6.4) is solved similarly, using the initial values given by $\mathbf{p}_0$, with solution

$$\mathbf{p}_t = \mathbf{p}_0 e^{Qt}.$$

This formula involves the exponential of a matrix $A = Qt$, which can be defined by the usual Taylor series formula, where all terms are reinterpreted as matrices. For any square matrix A

$$e^A = I + A + \frac{1}{2}A^2 + \frac{1}{3!}A^3 + \frac{1}{4!}A^4 + \dots. \tag{6.5}$$

Note that this is *not* the same as applying the usual exponential function to the entries of A individually; the powers that appear in the Taylor series result in interaction between the various entries of A.

While the Taylor series for the matrix exponential can provide a good conceptual understanding, to compute matrix exponentials easily requires more theory from linear algebra. Provided A can be diagonalized, then writing $A = S\Lambda S^{-1}$ with Λ the diagonal matrix of eigenvalues of A and S the matrix whose columns are the corresponding (right) eigenvectors, we also have (Exercise 16)

$$e^A = Se^{\Lambda}S^{-1}, \tag{6.6}$$

where e^{Λ} is a diagonal matrix with diagonal entries the exponentials of the entries of Λ.

Returning to our model, since $\mathbf{p}_t = \mathbf{p}_0 e^{Qt}$, it is natural to define

$$M(t) = e^{Qt}$$

as the Markov matrix which encodes the substitutions of bases that occur when an amount of time t passes. The entries of $M(t)$ are thus the conditional probabilities of the various substitutions being observed as we compare sequences from time 0 to those from time t, so that $M(t)$ is a single matrix describing the mutation along an edge representing a time of length t.

Our formulation of the model through the rate matrix Q has made the assumption that over the full time interval state changes have been occurring at exactly the same rates. Although we can be a little less strict than this by imagining our clock runs slow or fast at different times, we are still assuming the rates at which Rs becomes Ys and the rate at which Ys become Rs are in fixed proportion to each other. Thus the state substitution process is viewed as uniform, except perhaps for clock speed changes

Note the important distinctions between a Markov matrix M and a rate matrix Q. The entries of M are probabilities, and hence lie between 0 and 1, its rows sum to 1, and it describes a substitution process either along an entire edge, or over a discrete time step. The entries of Q are not probabilities, but rather rates describing the instantaneous substitution process, and hence do not need to be between 0 and 1. The off-diagonal entries, however, cannot be negative, and the ones on the diagonal cannot be positive. The rows of Q sum to 0 . Applying the matrix exponential function to the product of a rate matrix and an elapsed time gives a Markov matrix.

6.2 Markov Models on Trees

We now more carefully define a rather general model of DNA evolution along a tree. Although we use 4-state characters, the connection to the 2-state character version developed in the previous section should be clear. It is also straightforward to formulate a 20-state version appropriate for describing protein sequences, or a 61-state version for a codon model ($61 = 4^3 - 3$, since the 3 'stop' codons are generally excluded).

A general DNA base-substitution model

The possible states of our character are A,G,C,T. We always use this order for the four bases, so that the purines precede pyrimidines, with each in alphabetical order.

Consider a fixed rooted tree T^ρ. Then parameters for the *general Markov model* model on T^ρ consist of the following:

1) A *root distribution vector* $\mathbf{p}_\rho = (p_A, p_G, p_C, p_T)$, with all entries non-negative and $p_A + p_G + p_C + p_T = 1$. We interpret these entries as giving the probabilities that an arbitrary site in a DNA sequence at ρ is occupied by the corresponding base, or, equivalently, as the frequencies with which we would expect to observe these bases in a sequence at ρ. (Note the i.i.d. assumption here.)

2) For each edge $e = (u, v)$ of T directed away from ρ, a 4×4 *Markov matrix*, M_e, whose entries are non-negative and whose rows sum to 1. The i, j-entry of M_e we interpret as the conditional probability that if base i occurs at the site in the parent vertex on the edge, then base j occurs at the descendant vertex. Using abbreviated notation with $p_{ij} = \mathcal{P}(S_v = j \mid S_u = i)$, where S_u, S_v denote sequences at u and v respectively, we let
$$M_e = \begin{pmatrix} p_{AA} & p_{AG} & p_{AC} & p_{AT} \\ p_{GA} & p_{GG} & p_{GC} & p_{GT} \\ p_{CA} & p_{CG} & p_{CC} & p_{CT} \\ p_{TA} & p_{TG} & p_{TC} & p_{TT} \end{pmatrix}.$$

Note that the only mutations described by this model are base substitutions. No events such as insertions, deletions, or inversions are included in the formulation.

This basic model, called the *general Markov model*, will be our starting point. Shortly we will discuss more restrictive models, where we place additional requirements on the model parameters, and then later will present some generalizations. Although the general Markov model is seldom directly used for data analysis, it provides the basic framework for all models in widespread use.

As a final notational point, since we have fixed the ordering A, G, C, T, and will often need to refer to particular entries of vectors and matrices, it will also

be convenient to sometimes use numbers to refer to the bases, with

$$1 = \texttt{A},\ 2 = \texttt{G},\ 3 = \texttt{C},\ 4 = \texttt{T}.$$

A common rate-matrix model

Rather than allow completely unrelated Markov matrices for the base substitution process on each edge of the tree, most model-based phylogenetic analyses specify that the substitution processes along the various edges of the tree have some commonality. The usual way to do this is to replace point (2) above with the following:

2a) A continuous-time 4×4 rate matrix Q, whose rows add to 0, and whose off-diagonal entries are nonnegative. With

$$Q = \begin{pmatrix} q_{AA} & q_{AG} & q_{AC} & q_{AT} \\ q_{GA} & q_{GG} & q_{GC} & q_{GT} \\ q_{CA} & q_{CG} & q_{CC} & q_{CT} \\ q_{TA} & q_{TG} & q_{TC} & q_{TT} \end{pmatrix},$$

we interpret an entry q_{ij} as the instantaneous rate (in substitutions at a site per unit time) at which base i is replaced by base j.

2b) For each edge e of the tree a non-negative scalar length t_e. Then the Markov matrix

$$M_e = M(t_e) = e^{Qt_e}$$

can be interpreted as above in (2) for the edge e.

One attractive feature of this model is that it gives meaning to edge lengths t_e in a metric tree as specifying the 'amount' of substitution that will occur along that edge. With the general Markov model it is not immediately clear how to associate a single number to an edge to indicate such a measure. On the other hand, the assumption of a common process across the entire tree is a strong one (though overwhelmingly common in data analysis). In practice, the choice of Q is usually further restricted, as we will discuss later.

The distribution of character states at the leaves

With either general Markov parameters, or continuous-time parameters for a model on a tree T^ρ specified, we can compute probabilities of observing any particular combination of bases in aligned sequences at the vertices of a tree. We focus only on the probabilities of various base observations at the leaves, since that is the only sort of data that we typically can obtain from organisms. (For statistical inference, we will *not* know appropriate parameters to use for our model, but will have to somehow infer them from leaf data, inverting the process we are explaining here. That is the subject of later chapters.)

To see how to compute leaf distributions, we begin with several examples for very simple trees.

A one-edge tree

This case is similar to that discussed in the purine/pyrimidine example which began the chapter. But now we adopt the viewpoint that we know the root distribution, and a Markov matrix describing the substitution process over the edge, and we wish to compute the probability of observing each base at the single leaf of the tree.

If $\mathbf{p}_\rho$ and M_e are the parameters on a one edge tree from root ρ to descendant S1, then

$$\mathbf{p}_1 = \mathbf{p}_\rho M_e$$

gives a vector of probabilities of observing the four bases at any site in the descendant sequence. For instance, from the definition of matrix multiplication, the first entry of $\mathbf{p}_1$, which should refer to the probability of observing an `A` in the descendant sequence, is

$$p_1 M_e(1,1) + p_2 M_e(2,1) + p_3 M_e(3,1) + p_4 M_e(4,1) = \\ p_A p_{AA} + p_G p_{GA} + p_C p_{CA} + p_T p_{TA},$$

which has the claimed probabilistic interpretation. We are summing 4 terms for the 4 possible ancestral bases; each term gives the probability we had that ancestral base and it changed to (or remained as) an `A`. The other entries of $\mathbf{p}_1$ are produced similarly.

A two-edge tree

Suppose now S1, S2 are two children of a root ρ, with M_i the Markov matrix on the edge leading to Si, as in Figure 6.1.

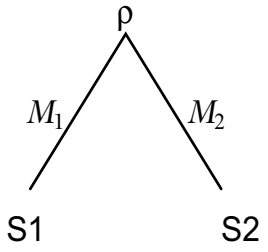

Figure 6.1: A two-edge tree

Then, if we ignore S2, we can compute the probabilities of the various bases appearing in S1 by the product $\mathbf{p}_\rho M_1$, and similarly for S2. But what is the *joint probability* that at a particular site we observe base i in S1, and base j at S2? Since we will have to consider every pair i, j of bases at the two leaves, we should organize the probabilities of these observations into a matrix P, with $P(i,j)$ denoting the probability of observing i at S1 and j at S2.

For an observation (i, j) to be produced, we might have any base k at ρ. Then this k must become an i in S1, and a j in S2. For a particular k, this means the probability is

$$p_k M_1(k, i) M_2(k, j).$$

But since k could be anything, we need to sum over the possibilities to get

$$P(i, j) = \sum_{k=1}^{4} p_k M_1(k, i) M_2(k, j) = p_1 M_1(1, i) M_2(1, j) + p_2 M_1(2, i) M_2(2, j) + p_3 M_1(3, i) M_2(3, j) + p_4 M_1(4, i) M_2(4, j).$$

This can be more succinctly expressed as a matrix equation. Letting $\text{diag}(\mathbf{v})$ denote a diagonal matrix with the entries of a vector $\mathbf{v}$ on its diagonal, the 4×4 matrix P whose entries give the joint distribution of bases at the leaves is given by

$$P = M_1^T \, \text{diag}(\mathbf{p}_\rho) M_2. \tag{6.7}$$

We leave checking this claim as Exercise 7.

We should also point out that once all the entries of any joint distribution P are computed, it is easy to recover the distribution of bases at a single leaf, by summing over the other indices, a process usually called *marginalization*. For instance, the probabilities of observing the base j at S2, without regard to what appears at S1 is found by summing over the index corresponding to S1, giving

$$\sum_{i=1}^{4} P(i, j).$$

Checking that this agrees with the jth entry of $\mathbf{p}_\rho M_2$ is Exercise 8.

A many-edge tree

Suppose now T^ρ has many edges, such as in Figure 6.2

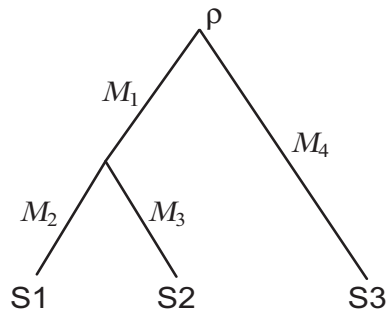

Figure 6.2: A small, many-edged tree

In general, we will not be able to give a simple formula for the joint distribution of bases at the leaves of the tree using standard matrix notation. Indeed, we should not expect to, because the joint distribution itself will be specified by an array of more than 2 dimensions. For instance, in the tree above, since there are 3 leaves, the joint distribution tensor P will be a $4 \times 4 \times 4$ array, with the entry $P(i, j, k)$ giving the probabilities that a site has bases i, j, k at the leaves $S1, S2, S3$, respectively. Using slightly different terminology, $P(i, j, k)$ gives the probability of observing the *pattern* ijk at a site in aligned sequences for $S1, S2, S3$.

A formula for any particular entry $P(i, j, k)$ of the joint distribution tensor is easily given, however, by following the same sort of reasoning as for the two-edge tree. We simply imagine each possible assignments of bases to all internal nodes of the tree, and then write a product expression for the probability of this particular evolutionary history. Afterwards, we sum over all such interior node assignments, since these are all the distinct ways of obtaining the pattern we are interested in. For example, for the tree above we have

$$P(3, 1, 1) = \sum_{i=1}^{4} \sum_{j=1}^{4} p_i M_1(i, j) M_2(j, 3) M_3(j, 1) M_4(i, 1) \tag{6.8}$$

as one of the 64 entries of P. Since there are 2 interior nodes in this tree, this sum has $4^2 = 16$ terms in it. Since there are 4 edges in the tree, each term is a product of $1 + 4 = 5$ parameters, one for each edge and one for the root. The other 63 entries are given by quite similar formulas.

For a tree relating more taxa, of course the formulas get more complicated, but the idea of how they are formed should be clear. It is worth emphasizing, though, that the formulas for the entries of P depend not just on the number of taxa, but also on the particular tree topology. Indeed, until we fix a tree topology to consider, we can't even relate Markov matrices to the particular edges.

Assumptions of the models

While we've been calling the matrices of conditional probabilities Markov matrices, we should say a little more about this terminology.

A probabilistic model of a system is called a *Markov model* if it incorporates an assumption of the following sort:

> The behavior of the system over any given time period depends only on the state the system is in at the beginning of that period. The earlier history of the system can affect what happens in the future only through having already affected the system's current state.

More formally, in a Markov model the probability that a particular state change occurs given the system is in state i is the same as the probability of the same change, given any entire earlier history of states ending in state i. In particular, a 'memory' of what state changes occurred during earlier times is

useless for predicting future changes. We say the probabilities of state changes are *independent* of the earlier history.

The model we have given here, of molecular evolution occurring through random base substitutions, satisfies the Markov assumption. Since time proceeds from the root to the leaves, and the probabilities of the various possible state changes on any given edge depend only on the state at the ancestral node on that edge, the above condition is met.

In fact, the general Markov model makes an even stronger, but equally reasonable, assumption that mutation process on one edge is not affected by what occurs on any edge that is not ancestral to it. Any correlations we observe between state observations at two leaves arise solely from the state of their most recent common ancestor.

In applications of the model we will also assume that each site in the sequence behaves identically, and independently of every other site (i.i.d). Though the model describes substitutions at a single site, it applies equally well to every site in the sequence. We used this assumption in our introductory example in order to find the various probabilities we needed from our sequence data, by thinking of each site as an independent trial of the same probabilistic process. We will continue to do this when we use more elaborate methods to determine appropriate parameter values from data.

The i.i.d. assumption is probably *not* very reasonable for DNA in many circumstances. For instance, since the genetic code allows for many changes in the third site of each codon that have no affect on the product of the gene, one could argue that substitutions in the third sites might be more likely than in the first two sites, violating the assumption that each site behaves identically. Also, since genes may lead to the production of proteins which have functional roles in life's processes, the chance of change at one site may well be tied to changes at another, through the need for the gene product to have a particular form. This violates the assumption of independence. A similar issue arises from the *secondary structure* of RNA formed by the transcription of genes. In the single-stranded RNA sequence, some bases form bonds to bases further down the sequence, producing features called 'stems' and 'loops' . This means that mutations at one site may well be tied to mutations at sites that are not even nearby in the sequence.

It's easy to come up with a host of other problems with the i.i.d. assumption: If sequences encode both coding and noncoding regions, should they really evolve the same way? Might some genes tend to mutate faster than others? Might not the bases in some sites be so crucial to a sequence's functioning that they are effectively prevented from mutating? Modifications to the basic modeling framework will at least partially address some of these issues, but some form of an i.i.d. assumption is always needed in order to have well-behaved inference in standard statistical frameworks. What matters is not that it is exactly true, but rather that it is a good enough approximation of the truth to apply statistical methods.

A useful analogy for understanding why the i.i.d. assumption is so crucial,

is to imagine data from a hundred coin flips. If your model is that the same (or an essentially identical coin) was flipped in the same way a hundred times, then the i.i.d. assumption holds, and by computing the frequency of heads, you can obtain a good estimate of the probability a single coin flip gives heads.

However, if you instead insist that these coin flips should be modeled as a hundred flips of varying coins, so we drop the identically distributed assumption, it is possible that all the heads were produced by coins weighted to always land heads up, and the tails by ones biased oppositely. If this were the case, then computing the frequency of heads among the hundred would not tell us anything useful about a single coin flip, or enable us to say anything about what might happen at additional sites if the sequences were longer.

If we instead drop the independent assumption, then we might flip one fair coin, and have it determine all the other outcomes, so that our data could only be all heads, or all tails. In this case, the frequency of heads among the hundred would either be 0 or 1, but that again tells us nothing about the probability of outcomes of a single coin flip.

6.3 Jukes-Cantor and Kimura Models

The general Markov model, or its continuous-time variant, is usually further restricted to special forms for data analysis. We present a few of these, starting from the most restricted, and then relaxing assumptions.

The Jukes-Cantor model

The simplest Markov model of base substitution, the *Jukes-Cantor* model, adds several additional assumptions to the general Markov model.

First, it assumes the root distribution vector describes all bases occurring with equal probability in the ancestral sequence. Thus

$$\mathbf{p}_\rho = \begin{pmatrix} 1/4 & 1/4 & 1/4 & 1/4 \end{pmatrix}.$$

Second, as a continuous-time model it assumes a rate matrix of the form

$$Q = \begin{pmatrix} -\alpha & \alpha/3 & \alpha/3 & \alpha/3 \\ \alpha/3 & -\alpha & \alpha/3 & \alpha/3 \\ \alpha/3 & \alpha/3 & -\alpha & \alpha/3 \\ \alpha/3 & \alpha/3 & \alpha/3 & -\alpha \end{pmatrix}. \tag{6.9}$$

This indicates that the rate of all specific base changes, `A` $\leftrightarrow$ `T`, `A` $\leftrightarrow$ `C`, `A` $\leftrightarrow$ `G`, `C` $\leftrightarrow$ `T`, `C` $\leftrightarrow$ `G`, and `T` $\leftrightarrow$ `G` are the same, $\alpha/3$. The total rate at which any specific base is changing to the other 3 bases is therefore α.

The associated Markov matrix on an edge of length t can now be calculated as

$$M(t) = e^{Qt}.$$

This requires first finding eigenvectors and eigenvalues for Q (see exercise 18) to obtain the diagonalization formula

$$Q = S\Lambda S^{-1},$$

with

$$S = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}, \quad \Lambda = \operatorname{diag}\left(0, -\frac{4}{3}\alpha, -\frac{4}{3}\alpha, -\frac{4}{3}\alpha\right). \tag{6.10}$$

Thus the Jukes-Cantor Markov matrix for an edge of length t is

$$\begin{aligned} M(t) &= e^{Qt} \\ &= Se^{\Lambda t}S^{-1} \\ &= S^{-1}\operatorname{diag}\left(1, e^{-\frac{4}{3}\alpha t}, e^{-\frac{4}{3}\alpha t}, e^{-\frac{4}{3}\alpha t}\right) S \\ &= \begin{pmatrix} 1-a & a/3 & a/3 & a/3 \\ a/3 & 1-a & a/3 & a/3 \\ a/3 & a/3 & 1-a & a/3 \\ a/3 & a/3 & a/3 & 1-a \end{pmatrix}, \end{aligned}$$

where

$$a = a(t) = \frac{3}{4}\left(1 - e^{-\frac{4}{3}\alpha t}\right). \tag{6.11}$$

Since the entries of $M(t)$ are probabilities, we interpret $a(t)$ as the probability that any specific base at time 0 will have changed to any of the 3 other bases at time t. This might have happened by only one base substitution occurring, or it might have happened through a succession of substitutions. The continuous-time model accounts for all possible ways the final state could have been achieved from the initial one. We are likely, of course, to use a different value of a for each edge of the tree, since the formula for a depends on the edge length. Larger values of t produce larger values of a, as more mutation occurs on that edge.

The Jukes-Cantor model also implies a *stable base distribution* at all vertices of the tree. To see this, we simply compute the effect of the substitution process as we proceed down an edge:

$$\begin{aligned} \mathbf{p}_\rho M &= \begin{pmatrix} 1/4 & 1/4 & 1/4 & 1/4 \end{pmatrix} \begin{pmatrix} 1-a & a/3 & a/3 & a/3 \\ a/3 & 1-a & a/3 & a/3 \\ a/3 & a/3 & 1-a & a/3 \\ a/3 & a/3 & a/3 & 1-a \end{pmatrix} \\ &= \begin{pmatrix} 1/4 & 1/4 & 1/4 & 1/4 \end{pmatrix}. \end{aligned}$$

Of course the simple, highly-symmetric form of the Jukes-Cantor matrix means that for every substitution from state i to j we should expect a substitution from j to i, so that is it not surprising the base distribution never changes.

We will return to equation (6.11) in the next chapter, as αt has an important interpretation. Since α is a mutation rate, measured in units (number of substitutions at a site)/(unit of time), and t represents an amount of time, the product tells us the number of substitutions that should occur at a site over the elapsed time t. While both α and t depend on our choice of units of time, the product has a meaning independent of those units.

Mutation rates such as α for DNA in real organisms are not easily found, since estimating them appears to require both an ancestral and descendant sequence, and knowledge of the evolutionary time separating them. For reasons that will be explained in the next chapter, its possible to avoid finding an ancestral sequence, but we do need an independent estimate of the divergence time of two descendants from their common ancestor, perhaps obtained from a fossil. Various estimates of α place it around 1.1×10^{-9} mutations per site per year for certain sections of chloroplast DNA of maize and barley and around 10^{-8} mutations per site per year for mitochondrial DNA in mammals. The mutation rate for the influenza A virus has been estimated to be as high as .01 mutations per site per year. The rate of mutation is generally found to be a bit lower in coding regions of nuclear DNA than in non-coding DNA.

The Kimura models

The Jukes-Cantor model is a particularly simple model of mutation since it depends on only one single parameter α to specify the rate of mutation.

The model can be made more flexible by allowing several parameters. A good example of this is the *Kimura 2-parameter* model, which allows for different probabilities of transitions and transversions.. If we let

$$Q = \begin{pmatrix} * & \beta & \gamma & \gamma \\ \beta & * & \gamma & \gamma \\ \gamma & \gamma & * & \beta \\ \gamma & \gamma & \beta & * \end{pmatrix},$$

then β is the rate at which each state undergoes transitions, and 2γ is the rate that any state undergoes transversions, with transversions equally likely to produce either possible outcome. The entries denoted by * should be $-\beta - 2\gamma$, since that is what is needed so that row sums are 0.

Although we leave the details as an exercise, a computation of the matrix exponential shows the associated Markov matrix has the form

$$M(t) = e^{Qt} = \begin{pmatrix} * & b & c & c \\ b & * & c & c \\ c & c & * & b \\ c & c & b & * \end{pmatrix},$$

with

$$b = \frac{1}{4}(1 - 2e^{-2(\beta+\gamma)t} + e^{-4\gamma t}), \quad c = \frac{1}{4}(1 - e^{-4\gamma t}).$$

Here the entries denoted * are $1-b-2c$, since the rows must add to 1. Notice that if the probabilities of a transition and each transversion are equal so $\beta = \gamma$, then this model includes the Jukes-Cantor one as a special case with $\alpha = 3\beta = 3\gamma$.

An even more general model is the Kimura 3-parameter model, which assumes a rate matrix of the form

$$Q = \begin{pmatrix} * & \beta & \gamma & \delta \\ \beta & * & \delta & \gamma \\ \gamma & \delta & * & \beta \\ \delta & \gamma & \beta & * \end{pmatrix},$$

which leads to a Markov matrix of the form

$$M = \begin{pmatrix} * & b & c & d \\ b & * & d & c \\ c & d & * & b \\ d & c & b & * \end{pmatrix}.$$

By an appropriate choice of the parameters, this includes both the Jukes-Cantor and Kimura 2-parameter models as special cases. (While the structure of the Kimura 2-parameter model is suggested by biology, the generalization to the 3-parameter is primarily motivated by mathematical properties.)

Both Kimura models also assume that the root distribution vector is the uniform one,

$$\mathbf{p}_\rho = \begin{pmatrix} 1/4 & 1/4 & 1/4 & 1/4 \end{pmatrix}.$$

A quick calculation shows that for either of these models this distribution will be stable, occurring at all vertices in the tree.

There is a clear relationship between the form of the rate matrix for the Jukes-Cantor and Kimura models and the form of the associated Markov matrices, with both exhibiting a very similar pattern of entries. However, this is a very special feature of these models, and such simple relationships do not occur more generally. While the entries of the Markov matrix will always be expressible in terms of those of the rate matrix and t, there is generally no easy way to find these expressions, except as outlined in the computation above for the Jukes-Cantor model.

A note of caution: When a Kimura model is used on a tree, there are two ways it might be used. It may be that one Kimura rate matrix (with fixed β,γ) is used for the entire tree with the 1 parameter of a scalar length assigned to each edge. Alternately, one may assign different and unrelated Kimura Markov matrices on each edge (giving 2 or 3 parameters per edge). The first approach is more common in standard approaches to phylogenetic inference, and is likely to be what is assumed in software performing Maximum Likelihood or Bayesian analyses. The second is a more general model, and often appears in more theoretical works, especially those dealing with Hadamard conjugation.

6.4 Time-reversible Models

An important feature of the Jukes-Cantor and Kimura models is that they are *time-reversible*. What this means is that given an ancestral and a descendant sequence separated by one edge of the tree, if we reverse the flow of time, interchanging which sequence we view as ancestor and descendant, we would describe evolution by exactly the same model parameters.

To see what this means mathematically, suppose $\mathbf{p}$ is the ancestral base distribution and M the Markov matrix describing the descent. Let P denote the joint distribution of bases in the ancestral and descendant sequences, so $P(i,j)$ gives the probability of an ancestral i and a descendant j appearing at a site. Then since $P(i,j) = p_i M(i,j)$, we have the matrix equation

$$P = \operatorname{diag}(\mathbf{p})M.$$

Now interchanging our view of what sequence is ancestral means interchanging the order of indices, or equivalently transposing P. Thus time reversibility means $P = P^T$, or

$$\operatorname{diag}(\mathbf{p})M = (\operatorname{diag}(\mathbf{p})M)^T = M^T \operatorname{diag}(\mathbf{p}). \tag{6.12}$$

In fact, this equation implies the intuitive fact that time-reversibility requires that $\mathbf{p}$ be a stable base distribution for M. (See Exercise 27.)

For the Jukes-Cantor and Kimura models it is easily checked that equation (6.12) holds in the discrete-time formulation. However there are many other possible parameter choices for a time-reversible model.

To elaborate, in the continuous-time formulation of a model, for time-reversibility to hold we need

$$\operatorname{diag}(\mathbf{p})Q = Q^T \operatorname{diag}(\mathbf{p}). \tag{6.13}$$

If we let

$$\mathbf{p} = \begin{pmatrix} p_A & p_G & p_C & p_T \end{pmatrix}$$

be the root distribution, then a little work (Exercise 28) shows Q should be of the form

$$Q = \begin{pmatrix} * & p_G\alpha & p_C\beta & p_T\gamma \\ p_A\alpha & * & p_C\delta & p_T\epsilon \\ p_A\beta & p_G\delta & * & p_T\eta \\ p_A\gamma & p_G\epsilon & p_C\eta & * \end{pmatrix}, \tag{6.14}$$

with $\alpha, \beta, \gamma, \delta, \epsilon, \eta \geq 0$ and where the diagonal entries are chosen so rows sum to zero.

Perhaps the most commonly-used basic continuous-time model used in data analysis is the *general time-reversible model* (GTR). It is specified by the following parameters:

1. an arbitrary choice of a root distribution $\mathbf{p} = \begin{pmatrix} p_A & p_G & p_C & p_T \end{pmatrix}$
2. arbitrary choices of 6 parameters $\alpha, \beta, \gamma, \delta, \epsilon, \eta$, with the common time-reversible rate matrix Q on all edges, given by equation 6.14, and

3. lengths of edges in the tree

The form of Q then ensures that $\mathbf{p}$ is stable under the model, and thus is the distribution of bases at all vertices in the tree.

It's easy to see that with special choices of the parameters, the GTR includes the Jukes-Cantor and Kimura models. But it has more flexibility due to the larger number of parameters, and thus is capable of describing a larger class of data sets well.

Practically, the GTR model seems to be a good compromise between simplicity and complexity. Having a common rate matrix reduces the number of parameters considerably from what would be needed without this assumption. This can help avoid 'overfitting' of data, keeping the variance in the inferred tree lower with the same amount of data. It also allows for faster run-times of software, as there are fewer parameters to be varied in searching for a good fit. A common rate matrix also imposes some commonality on the mutation process throughout the tree, which in some circumstances is biologically reasonable. That the base distribution is stable may also be reasonable, if all the data sequences have a similar base composition. Time reversibility means we will be able to ignore issues of root location when we try to find optimal trees, thereby reducing search time slightly. Nonetheless, it's hard to imagine a full justification of this model solely on biological grounds. Base compositions are not always identical across taxa, and the model cannot capture that. It is also hard to imagine a biological justification for time-reversibility.

While models that drop some of the characteristics of the GTR model are occasionally used in data analysis, the vast majority of phylogenetic analyses currently use the GTR model, or special cases of it, as their basis. While many of these special cases have their own names, they can all be viewed as simply imposing relationships among the GTR parameters to reduce their number. For instance the F81 model (introduced by Felsenstein in a paper in 1981) allows for any choice of the base distribution, but sets the remaining GTR parameters $\alpha = \beta = \gamma = \delta = \epsilon = \eta = 1$. It can thus be viewed as an analog of the Jukes-Cantor model that does not require equidistribution of the bases. The HKY model (introduced by Hasegawa, Kishino, and Yano) also allows any base distribution, but sets $\beta = \gamma = \delta = \epsilon = 1$ and $\alpha = \eta = \kappa$ where the parameter κ can be viewed as a transition/transversion rate ratio. Thus it is an analog of the Kimura 2-parameter model for unequal base distribution.

While its tempting to think that the GTR model should always be the best one to use since it makes fewer special assumptions, that in fact is not the case. In general its desirable to use a model that fits the data reasonably well, but has few parameters. More restrictive models of this sort can lead to better inference, since they avoid overfitting data.

6.5 Exercises

1. Suppose ancestral and descendant sequences of purines and pyrimidines are

   S0 = RRYRYYRRRRYYYRYYRRYYRYYR
   S1 = RYYRYRRRRRRYYRRYRRYYRYYY

   Use this data to estimate a distribution vector for S0 and a Markov matrix describing the mutation process from S0 to S1.

2. The joint frequencies of purines and pyrimidines in ancestral and descendant sequences S0 and S1 is summarized in Table 6.1. Use this data to estimate a distribution vector for S0 and a Markov matrix describing the mutation process from S0 to S1.

$S0 \diagdown S1$	R	Y
R	183	32
Y	15	211

   Table 6.1: Frequencies from site comparisons for a pair of sequences

3. An ancestral DNA sequence of 40 bases was

   CTAGGCTTACGATTACGAGGATCCAAATGGCACCAATGCT,

   but in a descendant it had mutated to

   CTACGCTTACGACAACGAGGATCCGAATGGCACCATTGCT.

   a. Give an initial base distribution vector and a Markov matrix to describe the mutation process.

   b. These sequences were actually produced by a Jukes-Cantor simulation. Is that surprising? Explain. What value would you choose for the Jukes-Cantor parameter a to approximate your matrix by a Jukes-Cantor one?

4. Data from two comparisons of 400-base ancestral and descendant sequences are shown in Table 6.2.

   a. For one of these pairs of sequences a Jukes-Cantor model is appropriate. Which one, and why?

   b. What model would be appropriate for the other pair of sequences? Explain.

5. The Markov matrices that describe real DNA mutation tend to have their largest entries along the main diagonal in the (1,1), (2,2), (3,3), and (4,4) positions. Why should this be the case?

$S_0 \diagdown S_1$	A	G	C	T
A	92	15	2	2
G	13	84	4	4
C	0	1	77	16
T	4	2	14	70

$S_0' \diagdown S_1'$	A	G	C	T
A	90	3	3	2
G	3	79	8	2
C	2	4	96	5
T	5	1	3	94

Table 6.2: Frequencies from 400 site comparisons for two pairs of sequences

6. On the tree in Figure 6.2, consider the Jukes-Cantor model where all the M_i have $a = 0.1$. Compute the probabilities of observing each of the following characters

   a) S1: `G`, S2: `G`, S3: `G`

   b) S1: `G`, S2: `G`, S3: `T`

   c) S1: `G`, S2: `T`, S3: `G`

   Are the largest and smallest of these probabilities what you might have expected? Explain.

7. Check that the matrix equation (6.7) is correct.

8. Let u denote a column vector with all entries 1. Explain why for a Markov matrix M that $Mu = u$. Then use this fact to show that summing over the first index of

$$M_1^T \operatorname{diag}(\mathbf{p}_\rho) M_2$$

   gives

$$\mathbf{p}_\rho M_2.$$

   Interpret this as explaining why the marginalization of a joint distribution of bases at two leaves gives the distribution of bases at one leaf.

9. For the 4-taxon tree $((a, b), (c, d))$ give a formula like that in equation (6.8) for the joint distribution of bases at the leaves in terms of parameters of the general Markov model. Do the same for the 4-taxon tree $(((a, b), c), d)$.

10. Suppose T^ρ is a rooted binary n-taxon tree. How many terms would be summed in the formula analogous to Equation (6.8) for computing the probability of a particular character? How many parameters would be multiplied in each term of this sum?

11. Make up a 4×4 Markov matrix M with all positive entries, and an initial $\mathbf{p}_0$. To be biologically realistic, make sure the diagonal entries of M are the largest.

    a. Use a computer to observe that after many time steps $\mathbf{p}_t = \mathbf{p}_0 M^t$ appears to approach some equilibrium. Estimate the equilibrium vector as accurately as you can.

b. Is your estimate in part (a) a left eigenvector of M with eigenvalue 1? If not, does it appear to be close to having this property?

c. Use a computer to compute the eigenvectors and eigenvalues of M. (In MATLAB the command `[S D]=eig(M)` computes right eigenvectors, so you will have to apply it to `M'`). Is 1 an eigenvalue? Is your estimate of the equilibrium close to its eigenvector?

12. Express the Kimura 2-parameter model using a 4×4 matrix, but with the bases in the order `A`,`C`,`G`,`T`. How is your matrix different from the one presented in the text? Explain.

13. Suppose we wish to model molecular evolution not at the level of DNA sequences, but rather at the level of the proteins that genes encode.

    a. Create a simple one-parameter mathematical model (similar to the Jukes-Cantor model) describing the process. You will need to use that there are 20 different amino acids from which proteins are constructed in linear chains.

    b. In this situation, how many free parameters would the general Markov model have on a single edge of the tree?

14. Do the data in Exercise 1 appear to be fit by a time-reversible model? Explain.

15. Suppose you have compared two sequences S_α and S_β of length 1000 sites and obtained the data in Table 6.3 for the number of sites with each pair of bases.

$S_\alpha \diagdown S_\beta$	A	G	C	T
A	105	15	15	15
G	25	175	25	25
C	35	35	245	35
T	25	25	25	175

Table 6.3: Frequencies of $S_\alpha = i$ and $S_\beta = j$ in 1000 site sequence comparison

    a. By marginalizing, compute the base distribution for each of the taxa individually. Does it appear to be stable?

    b. Assuming S_α is the ancestral sequence, find an initial base distribution $\mathbf{p}_0$ and a Markov matrix M to describe the data. Is your matrix M Jukes-Cantor? Is $\mathbf{p}_0$ a stable distribution for M?

    c. Assuming S_β is the ancestral sequence, find an initial base distribution $\mathbf{p}'_0$ and a Markov matrix M' to describe the data. Is your matrix M' Jukes-Cantor? Is $\mathbf{p}'_0$ a stable distribution for M'?

    You should have found that one of your matrices was Jukes-Cantor and the other was not. This can't happen if both S_α and S_β have base distribution $(.25, .25, .25, .25)$.

16. Assuming $A = S\Lambda S^{-1}$, show that equation (6.5) implies equation (6.6). Also explain why if a matrix B is diagonal then e^B is also diagonal, with diagonal entries obtained by exponentiating those of B.

17. Show that if the rows of Q sum to 0, then the rows of e^{Qt} sum to 1. (Hint: Use the Taylor series for the exponential, and the fact that the rows of Q summing to 0 is expressible as $Q\mathbf{u} = 0$, where $\mathbf{u}$ is a column vector with all entries 1.)

18. Check that the eigenvectors and eigenvalues of the Jukes-Cantor rate matrix Q in equation (6.9) are those given in equations (6.10).

19. The matrix S of eigenvectors of a Jukes-Cantor matrix that is given in equation 6.10 is quite special, and is sometimes called a *Hadamard matrix*. Compute S^2, and explain why this shows that $S^{-1} = \frac{1}{4}S$.

20. The formula for e^{Qt} for the Jukes-Cantor model in equation (6.11) and its predecessor can be used to understand the effect of an infinite amount of mutation, by letting $t \to \infty$.

    a. If $\alpha > 0$, what is $\lim_{t\to\infty} e^{-\frac{4}{3}\alpha t}$.

    b. Use this to explain why

$$e^{Qt} \to \begin{pmatrix} .25 & .25 & .25 & .25 \\ .25 & .25 & .25 & .25 \\ .25 & .25 & .25 & .25 \\ .25 & .25 & .25 & .25 \end{pmatrix}.$$

    Note that each of the rows of this matrix is the stable distribution.

    Explain informally why this limit is what you should have expected.

    c. Why did we exclude $\alpha = 0$ from our analysis?

21. Based on the last problem, one might conjecture that powers of a Markov matrix all of whose entries are positive approach a matrix whose rows are the stable distribution. On a computer, investigate this experimentally by creating a Markov matrix, computing very high powers of it to see if the rows become approximately the same, and then checking whether this row is a left eigenvector with eigenvalue 1 of the original matrix.

22. Let $M(a)$ denote a Jukes-Cantor Markov matrix with parameter a.

    a) Show the product $M(a_1)M(a_2)$ is $M(a_3)$, and give a formula for a_3 in terms of a_1, a_2.

    b) If a Jukes-Cantor matrix $M(a)$ describes the evolution of one sequence to another, then the Hamming distance estimates a. Explain why the formula you found in part (a) indicates the Hamming distance is not usually additive in the sense defined in Exercise 6 of Chapter 5.

    c) Explain why the formula you found in (a) indicates the Hamming distance is approximately additive when its values are small.

23. Show the product of two Kimura 3-parameter Markov matrices is again a Kimura 3-parameter Markov matrix.

24. Show the Kimura 3-parameter matrices (both Markov and rate) have the same eigenvectors as those given in the text for the Jukes-Cantor matrices. What are the eigenvalues of the Kimura 3-parameter rate matrices?

25. Use the results of the last problem to find the entries of e^{Qt} where $Q = Q(\beta, \gamma, \delta)$ is the Kimura 3-parameter rate matrix. Your result should be a Kimura 3-parameter Markov matrix. Give formulas for the Markov matrix entries b, c, d in terms of β, γ, δ, t. Show that in the special case of the Jukes-Cantor and Kimura 2-parameter models, these agree with the formulas given in the text.

26. The Jukes-Cantor model can be presented in a different form as a 2×2 Markov model. Let q_t represent the fraction of sites that agree between the ancestral sequence and the descendant sequence at time t, and p_t the fraction that differ, so $q_0 = 1$ and $p_0 = 0$. Assume that the instantaneous rate at which base substitutions occurs is α, and that each of the 3 possible base substitutions is equally likely. Then

$$\begin{pmatrix} q'(t) \\ p'(t) \end{pmatrix} = \begin{pmatrix} 1-\alpha & \frac{\alpha}{3} \\ \alpha & 1-\frac{\alpha}{3} \end{pmatrix} \begin{pmatrix} q(t) \\ p(t) \end{pmatrix}, \quad \begin{pmatrix} q(0) \\ p(0) \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}.$$

    a. Explain why each entry in the matrix has the value it does. (Observe that $1 - \frac{\alpha}{3} = (1-\alpha) + \frac{2\alpha}{3}$.)

    b. Compute the stable distribution of the model by finding the eigenvector with eigenvalue 1.

    c. Find the other eigenvalue and eigenvector for the matrix.

    d. Use (b) and (c), together with the initial conditions to give a formula for $q(t)$ and $p(t)$ as functions of time.

27. Show equation (6.12) implies that $\mathbf{p}$ is a stable base distribution for M.

28. Show that a time reversible rate matrix Q can be expressed by the formula (6.14).

29. Suppose Q is a rate matrix.

    a) Show that if $\mathbf{p}Q = \lambda\mathbf{p}$ and $M(t) = e^{Qt}$ for some t, then $\mathbf{p}M(t) = e^{\lambda t}\mathbf{p}$.

    b) From a) deduce that if $\mathbf{p}Q = 0$, then $\mathbf{p}$ is a stable distribution for all $M(t)$.

30. Suppose Q is a time-reversible rate matrix with stable base distribution $\mathbf{p}$.

    a) Explain why replacing Q with any positive scalar multiple cQ can lead to exactly the same joint distributions on any tree, if edge lengths are adjusted appropriately. Why is this change equivalent to using a new time scale?

b) In light of part (a), it is sometimes convenient to choose a specific normalization of Q. Explain why $-\mathrm{Tr}(\mathrm{diag}(\mathbf{p})Q)$ gives the instantaneous rate of substitutions for the model, and why we can always rescale Q so this is 1. Here $\mathrm{Tr}(M) = \sum_{i=1}^{n} M_{i,i}$ denotes the trace of a matrix M.

31. Show that a time-reversible Markov matrix M with a stable distribution vector $\mathbf{p}$ which has all positive entries must have a full set of real eigenvalues and eigenvectors. (Hint: Show $\mathrm{diag}(\mathbf{p})^{1/2} M \,\mathrm{diag}(\mathbf{p})^{-1/2}$ is symmetric.)

32. The general Markov model is not time reversible, but has a related weaker property.

    a. Show it is not time reversible by giving a specific $\mathbf{p}$, M that do not satisfy equation (6.12).

    b. Show that with mild conditions on $\mathbf{p}$, M, there exist $\tilde{\mathbf{p}}$, $\tilde{M}$ so that

    $$\mathrm{diag}(\mathbf{p})M = \tilde{M}^T \,\mathrm{diag}(\tilde{\mathbf{p}}).$$

    Thus, by changing parameter values for the general Markov model, we can change our viewpoint as to what is ancestral and what is descendant.

    c. Explain the connection between part (b) and Bayes Theorem.

Chapter 7

Model-based Distances

With an explicit model of DNA mutation in hand, we can now develop more sophisticated dissimilarity measures than the Hamming metric. The advantage of using a probabilistic model is that it enables us to account for the hidden substitutions that might have occurred, even though they are not seen when sequences are compared. We will then be able to use a measure of total change, rather than just directly observed change, as a measure of dissimilarity. These improved measures might then be used with any distance method for inferring a tree — either an algorithmic one such as UPGMA or Neighbor Joining, or one based on an optimality criterions.

As mentioned earlier, the Hamming dissimilarity is often called the *uncorrected* distance between sequences. The distances we will develop from explicit models are called *corrected* distances, since they account for the unobservable hidden base changes to give a better estimate of the total amount of change.

7.1 Jukes-Cantor Distance

To frame the issue we want to address more clearly, let's begin with the simplest model, Jukes-Cantor. We imagine an ancestral sequence S0 has base distribution $\mathbf{p}_0 = (1/4, 1/4, 1/4, 1/4)$ and its mutation is governed by a Jukes-Cantor rate matrix

$$Q = \begin{pmatrix} -\alpha & \alpha/3 & \alpha/3 & \alpha/3 \\ \alpha/3 & -\alpha & \alpha/3 & \alpha/3 \\ \alpha/3 & \alpha/3 & -\alpha & \alpha/3 \\ \alpha/3 & \alpha/3 & \alpha/3 & -\alpha \end{pmatrix}.$$

Here α is the rate at which any given base is replaced by a different base.

As we saw in the last chapter, if an amount of time t passes, then the total

mutation process over that elapsed time can be described by

$$M(t) = e^{Qt} = \begin{pmatrix} 1-a & a/3 & a/3 & a/3 \\ a/3 & 1-a & a/3 & a/3 \\ a/3 & a/3 & 1-a & a/3 \\ a/3 & a/3 & a/3 & 1-a \end{pmatrix},$$

where a and αt are related by equation (6.11), which we recall was

$$a = a(t) = \frac{3}{4}\left(1 - e^{-\frac{4}{3}\alpha t}\right). \tag{7.1}$$

Letting S1 be the descendant sequence of S0 after time t, the distribution of characters at the sites in the 2 sequences is given by the table

$$\text{diag}(\mathbf{p}_0)M(t) = \begin{pmatrix} (1-a)/4 & a/12 & a/12 & a/12 \\ a/12 & (1-a)/4 & a/12 & a/12 \\ a/12 & a/12 & (1-a)/4 & a/12 \\ a/12 & a/12 & a/12 & (1-a)/4 \end{pmatrix}. \tag{7.2}$$

Here rows refer to the state in S0, and columns to S1. (The time-reversibility of the model results in a symmetric matrix, so if we reversed rows and columns it actually would not matter.)

While equation (7.2) is a theoretical distribution arising from our model, we could easily obtain a corresponding empirical distribution simply by computing the frequency with which we observe each pair of states at sites in the two sequences. If our model is a good one for the data, these should be close, and thus we should be able to obtain an estimate $\hat{a}$ of a from the empirical version of the matrix in equation (7.2).

Recall the expression αt appearing in equation (7.1) has a simple interpretation: It is the product of a rate, measured in units of (substitutions at a site)/(unit of time), and an elapsed time. While choosing some specific unit of time would be necessary to give α and t specific values, the product αt has meaning even without this choice. It is the total number of substitutions at a site that occur over the full time period, *including all those that are hidden due to multiple substitutions at that site*. While αt is not something we can directly observe by comparing initial and final sequences, it is the correct measure of the total amount of mutation that occurred under this model.

Notice also that equation (7.1) will not let us tease out values of α and t separately from an estimated value for a; only their product appears. If twice the time passed, but the mutation rate was halved, that would result in exactly the same distribution. This is the first indication of a rather fundamental issue that will continue to arise from these probabilistic models. While it is possible to recover the total amount of mutation on each branch of a tree from sequence data, we cannot recover times or mutation rates without using some additional information. The elapsed time and the mutation rate are *unidentifiable* from the basic model, although their product is identifiable.

A molecular clock assumption of a constant mutation rate would ensure the amount of mutation is just a rescaled measure of time, but that also suggests our trees should be ultrametric. Since that is not usually the case for trees inferred from data sequences, we have to conclude the rate is generally not constant. However, when we specified a rate matrix for a model, we appeared to be assuming the rate was constant. The way around this apparent contradiction is by viewing time in our model not as measured by a normal clock, but rather by one that might speed up and slow down on each edge independently. Changes in generation time could be a simple biological explanation of this, but there might be other causes as well. By simply allowing non-ultrametric trees, we can incorporate this simple sort of rate variation into our models.

To estimate αt from two sequences our strategy is simple. First by comparing the sequences obtain an estimate $\hat{a}$ of a in the table in equation (7.2). Since an empirical version of (7.2) will not have exactly the pattern of the theoretical one, but rather show variation in the off-diagonal entries, we define $\hat{a}$ to be the sum of all 12 off-diagonal entries. Then if the model fits well we should have $\hat{a} \approx a$. Since the off-diagonal entries are the frequencies of the various ways the states may disagree at a site in the two sequences,

$$\hat{a} = \frac{\text{number of sites that show different states}}{\text{total number of sites}}$$

which is just the Hamming dissimilarity measure, or p-distance, introduced in Chapter 5. To estimate a, then, there is really no need to even create the table (except to judge whether a Jukes-Cantor model might be plausible).

Since solving for αt in equation 7.1 yields

$$\alpha t = -\frac{3}{4} \ln \left(1 - \frac{4}{3} a \right),$$

we can estimate the total amount of mutation by

$$\widehat{\alpha t} = -\frac{3}{4} \ln \left(1 - \frac{4}{3} \hat{a} \right).$$

We therefore define the *Jukes-Cantor distance* between DNA sequences S0 and S1 as

$$d_{JC}(\text{S0}, \text{S1}) = -\frac{3}{4} \ln \left(1 - \frac{4}{3} \hat{a} \right).$$

Provided the Jukes-Cantor model accurately describes the evolution of one sequence into another, this distance is an estimate of the total number of substitutions per site that occurred during the evolution.

Example. Suppose an ancestral sequence $ATTGAC$ has evolved into a descendant sequence $ATGGCC$. We estimate $\hat{a} = 2/6 \approx .3333$, so that on average we observe 1/3 of a substitution per site when we compare the sequences. Then

$$d_{JC}(S0, S1) = -\frac{3}{4} \ln \left(1 - \left(\frac{4}{3} \right) \left(\frac{2}{6} \right) \right) \approx 0.4408.$$

Note that the Jukes-Cantor distance is larger than $\hat{a}$, as it should be since it accounts for hidden substitutions as well as observed ones. We have estimated that, on average, there were actually about 0.4408 substitutions of bases in each site during the evolution of S0 to S1.

Of course if this were real data we'd have little faith in this estimate, since the sequences were so short. We'd be more confident of an estimate derived from longer sequences, with hundreds of sites.

In practice, ancestral and descendant sequences are rarely available to compare. Typically only the sequences at the leaves of a tree, from currently living taxa, are available. However, because the Jukes-Cantor model is time reversible, we can get around this issue. In modeling the descent of sequences S1 and S2

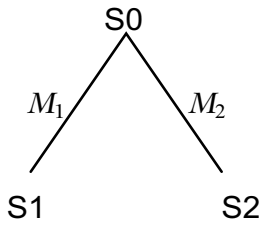

Figure 7.1: Two descendants of an ancestor. Under a time-reversible model, we may view $S1$ as ancestral to $S0$ which is ancestral to $S2$.

from S0, we originally think of the root as S0, with Jukes-Cantor matrices $M_1 = M(t_1)$ and $M_2 = M(t_2)$ describing the state change process. However, time-reversibility ensures S0 and S1 will have the same distribution of characters if we instead think of S1 as ancestral to S0, using the same Markov matrix $M(t_1)$. Then the relationship between S1 and S2 can be viewed as S1 evolving into S0 which then evolves into S2. The combined process from S1 through S0 to S2 is described by the product $M_1 M_2 = M(t_1 + t_2)$. Thus the Jukes-Cantor distance $d_{JC}(S1, S2)$ estimates $\alpha(t_1 + t_2)$, the total mutation that occurred on the path from S1 to S2.

If one had infinitely long sequences produced exactly in accord with the Jukes-Cantor model on a metric tree, then one could use the Jukes-Cantor distance formula to give dissimilarities that exactly match the tree metric distance between any pair of taxa, up to scaling by α. Since these distances would then exactly fit a metric tree (the tree on which evolution occurred, with 'time' scaled by α), one could use Neighbor Joining, or other methods, to recover the tree from the dissimilarities. In the real world of finite length sequences, and a substitution process that is at best roughly described by the Jukes-Cantor model, the Jukes-Cantor distance will not exactly match a tree metric, but should be close if our modeling assumptions are reasonable. If the error is not too large, one can prove that Neighbor Joining and many other distance methods will still recover the correct tree topology, and give good estimates of edge lengths.

To further explore the Jukes-Cantor distance, fix the rate to be $\alpha = 1$. This choice is arbitrary, but amounts to setting a time scale so that a site undergoes mutations at a rate of 1 substitution per unit of time. Then equation 7.1 becomes $a = a(t) = \frac{3}{4}(1 - e^{-\frac{4}{3}t})$, which is graphed in Figure 7.2. This figure can also be read with a as the independent variable, in which case the graph is of $t = -\frac{3}{4}\ln\left(1 - \frac{4}{3}a\right)$. Therefore it relates Hamming distances on the vertical axis to Jukes-Cantor distances on the horizontal axis.

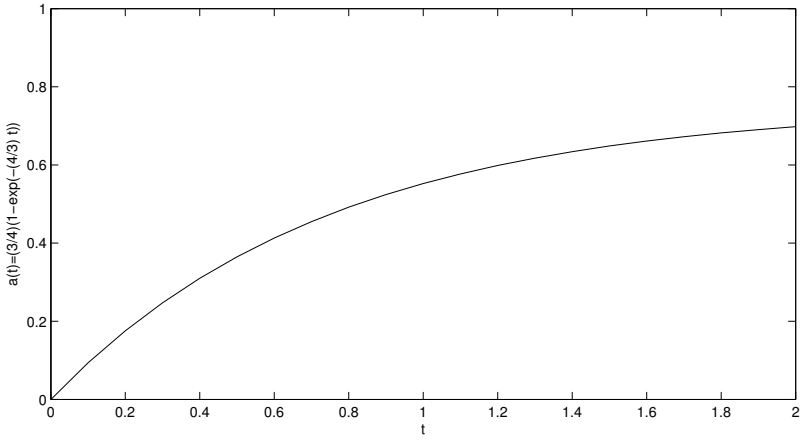

Figure 7.2: The relationship between elapsed time t and the probability a of differing states in a character for 2 taxa under the Jukes-Cantor model, with rate $\alpha = 1$. This graph can also be read as the Hamming distance $\hat{a}$ between two sequences on the vertical axis, and the resulting Jukes-Cantor distance on the horizontal axis.

In the figure we of course see that $a(0) = 0$, since at time $t = 0$ no substitutions have yet occurred. For small values of t (say $0 \leq t \leq .2$), we find $a(t) \approx t$, so whether we use the Jukes-Cantor distance or simply the Hamming distance as a dissimilarity measure has little effect. However, when t is large, $a(t)$ approaches the 'saturation point' of 3/4, where we find 3 out of 4 sites are observed to have changed. Equivalently, when the Hamming distance is near 3/4, the Jukes-Cantor distance will be much greater than 3/4, and using this corrected distance instead of the Hamming distance may have a dramatic effect on inferring a tree.

The saturation value of 3/4 deserves more explanation. Imagine picking two unrelated sequences at random, using the base distribution assumed by the Jukes-Cantor model. Then regardless of what base is chosen at a site in the first sequence, we have a 1/4 probability of picking the same base in the second sequence. Thus we expect that 3/4 of the sites in the sequences will show disagreement. The graph in Figure 7.2 shows that as more time passes, two

related sequences will come closer to resembling ones that have no relationship whatsoever.

The shape of the graph in Figure 7.2 also has implications for how much confidence we should place in a Jukes-Cantor distance estimate. Suppose the 'true' value of a differs slightly from the Hamming distance $\hat{a}$ computed from data, so $\hat{a} = a + \epsilon$. Then if a and $\hat{a}$ are both small, locating them on the lower end of the vertical axis shows they correspond to differing values of t and Jukes-Cantor distance $\hat{t}$. In the region of the graph where $a \approx t$, we see that $\hat{t} \approx t + \epsilon$. Thus the error in our time estimate is roughly the same as it was in the estimate of a.

On the other hand, if a and $\hat{a}$ are larger, locating them higher on the vertical axis shows they correspond to values of t and $\hat{t}$ that are much further apart than ϵ. The error is thus magnified by the Jukes-Cantor distance formula. More formally, one could show that a confidence interval for the estimate of t is much larger when $\hat{a}$ is large. Informally, large distances are likely to be less reliable than smaller ones.

Before we leave the Jukes-Cantor model, we note that the explanation given here for the Jukes-Cantor distance formula, while clearly based on a mathematical model, has shortchanged some statistical issues. In particular, while it is certainly reasonable, no justification has been given that the Hamming distance is the best estimate to use for the true but unknown value a in the distance formula. We'll return to this issue when we discuss Maximum Likelihood methods in Chapter 8.

7.2 Kimura and GTR Distances

Given a Markov model of base substitutions one can try to imitate the steps above in the derivation of the Jukes-Cantor distance formula. For this to make sense, however, we need a time-reversible continuous-time model, so that it includes a notion of elapsed time, and we can freely take the viewpoint that any of our data sequences represent the ancestral one.

Such distances have been found for a number of models, ranging from Jukes-Cantor to GTR. As the models become more complex, with more parameters, the formulas of course become more complicated.

For instance, the distance formula for the Kimura 3-parameter model (see Exercise 10) is

$$d_{K3}(S1, S2) = -\frac{1}{4}\left(\ln(1 - 2\hat{b} - 2\hat{c}) + \ln(1 - 2\hat{b} - 2\hat{d}) + \ln(1 - 2\hat{c} - 2\hat{d})\right),$$

where $\hat{b}$, $\hat{c}$, and $\hat{d}$ are estimates of parameters b, c, and d for a Kimura 3-parameter Markov matrix describing the mutation process between the two sequences. Specifically, $\hat{b}$ is the proportion of sites in the sequences showing transitions, $\hat{c}$ is the proportion showing transversions of the type `A` $\leftrightarrow$ `C` and

G $\leftrightarrow$ T, and $\hat{d}$ is the proportion showing transversions of the type A $\leftrightarrow$ T and G $\leftrightarrow$ C. The Kimura 3-parameter distance is an estimate of $(\beta + \gamma + \delta)t$.

For the Kimura 2-parameter model, $c = d$, so we instead let $\hat{e} = \hat{c} + \hat{d}$ be the proportion of transversions, and obtain the Kimura 2-parameter distance from this as

$$d_{K2}(S1, S2) = -\frac{1}{2}\ln(1 - 2\hat{b} - \hat{e}) - \frac{1}{4}\ln(1 - 2\hat{e}).$$

The distance formula appropriate for the GTR model is more complex, given by the formula

$$d_{GTR}(S1, S2) = -\operatorname{Tr}(\operatorname{diag}(\mathbf{p}) \ln\left(\operatorname{diag}(\mathbf{p})^{-1}(1/2)\left(F + F^T\right)\right).$$

where F is the 4×4 table of observed frequencies of base pairs in the two sequences. The logarithm here is a matrix one, the inverse of the matrix exponential. (See Exercise 14 for more details.)

7.3 Log-det Distance

One drawback of the distances developed so far is that they assume an underlying continuous-time models with a stable base distribution and a common substitution process occurring at all parts of the tree. For some data sets these assumptions are inappropriate. For instance, the sequences for different taxa may have substantial differences in base composition, or there may be reason to suspect the substitution process has varied. Although the GTR model and its submodels are not appropriate in this situation, the general Markov model, in which possibly unrelated Markov matrices are placed on each edge of the tree and an arbitrary root distribution is chosen, may be.

However, there is no notion of a flow of time in the general Markov model; the Markov matrices simply represent the full substitution process from one end of an edge to the other. Thus to develop a distance appropriate for this model we focus on mathematical properties of matrices and of distances.

Our motivation is thus not on reconstructing the total number of base substitutions that occurred, but rather on the properties we need if our distance is to be a restriction of a tree metric.

These are:

1) $d(\mathrm{S0}, \mathrm{S1}) \geq 0$, and $d(\mathrm{S0}, \mathrm{S1}) = 0$ if, and only if, S0 = S1,

2) $d(\mathrm{S0}, \mathrm{S1}) = d(\mathrm{S1}, \mathrm{S0})$,

3) $d(\mathrm{S0}, \mathrm{S2}) = d(\mathrm{S0}, \mathrm{S1}) + d(\mathrm{S1}, \mathrm{S2})$ provided S1 lies at an vertex along the path between S0 and S1.

The first property says that a distance should indicate when sequences are different. The second is similar to time-reversibility of a model, in that it says the distance will not be affected by which sequence we view as ancestral. However,

it does not require that the model itself be time-reversible. The last of these properties, called *additivity* means that individual distances down a lineage will add to give the total distance. In the earlier discussion of the Jukes-Cantor distance, these last two properties were the ones we used in arguing that we could compute a Jukes-Cantor distance between sequences on leaves of a tree to estimate the length of the path between them.

As motivation for the distance we will soon define, we focus on property (3). If M_1, and M_2 are the Markov matrices describing the substitution process from S0 to S1 and from S1 to S2, respectively, then the product M_1M_2 describes the process from S0 to S2. Now to associate a scalar distance to each edge, we might try the determinant of the matrices, since that is a natural way of obtaining a single number from a matrix. Moreover, a key property of the determinant is

$$\det(M_1M_2) = \det(M_1)\det(M_2).$$

By taking a logarithm (which requires that the number be positive) this can be converted into an additive statement.

$$\ln|\det(M_1M_2)| = \ln|\det(M_1)| + \ln|\det(M_2)|.$$

Thus the logarithm of the determinant of the Markov matrix relating two sequences has some of the features we need. Unfortunately it isn't quite a good definition of a distance, since it fails to satisfy properties (1) and (2).

A closely related quantity that does have the desired properties when applied to data from infinitely long sequences produced in accord with the general Markov model is obtained as follows:

Definition. Let $\hat{F}$ be the 4×4 frequency array obtained by comparing sites in sequences S0 and S1, with the i, j entry of $\hat{F}$ being the proportion of sites with base i in the S0 sequence and j in the S1 sequence. Let $\mathbf{f}_0$ and $\mathbf{f}_1$ be the frequency vectors for the bases in S0 and S1, respectively, which are obtained from $\hat{F}$ by row and column marginalizations.

Then the *log-det distance* between S0 and S1 is defined by

$$d_{LD}(\text{S0}, \text{S1}) = -\frac{1}{4}\left(\ln\left|\det(\hat{F})\right| - \frac{1}{2}\ln(g_0g_1)\right),$$

where g_i is the product of the 4 entries in $\mathbf{f}_i$.

The log-det distance is also called the *paralinear distance*, as it was given different names when independently constructed by two authors.

That the formula for the log-det distance, when applied not to $\hat{F}$ but to a theoretical distribution F arising from the general Markov model, gives a quantity that satisfies properties (2) and (3) will be shown in Exercise 13. (Property (1) is a little messier to establish.) As $\hat{F}$ is an estimate of F, the properties hold approximately when log-det distance are computed from finite length data sequences produced roughly in accord with the general Markov model.

Unlike the other distance formulas discussed here, the log-det distance cannot always be interpreted as the total number of mutations per site that must have occurred over the evolutionary history. Still, because the distance has the three formal properties above it is a reasonable measure of the amount of mutation that has occurred. In special circumstances, such as when the Jukes-Cantor or Kimura models apply exactly, and theoretical rather than empirical distributions are used, it gives the same result as some other distance formulas. (See Exercise 11.)

Finally, if a Jukes-Cantor, Kimura, or other submodel of the GTR model is adequate for describing sequence data, it is better to use distance formulas designed for the most restrictive yet adequate model. The use of a more general model increase the possibility of 'overfitting' the data, making statistical infererence of the amount of mutation that occurred less reliable. Since distances computed from data are unlikely to ever fit a metric tree exactly, getting values closer to the 'true' distances by using the most restrictive model that is appropriate will improve our chance of recovering the 'true' tree by whatever distance method is used. The use of the log-det distance is justified if either the sequences being analyzed have significant differences in base composition, or there is reason to doubt that the substitution process is the same on all edges of the tree. Both of these possibilities rule out the use of standard continuous-time models, but not the general Markov model which is the basis of log-det.

7.4 Exercises

1. Calculate $d_{JC}(S0, S1)$ for the two 40 base sequences

   $S0$: `CTAGGCTTACGATTACGAGGATCCAAATGGCACCAATGCT`
   $S1$: `CTACGCTTACGACAACGAGGATCCGAATGGCACCATTGCT`.

2. Ancestral and descendant sequences of 400 bases were simulated according to the Jukes-Cantor model. A comparison of aligned sites gave the frequency data in Table 7.1.

$S0 \diagdown S1$	A	G	C	T
A	90	3	3	2
G	3	79	8	2
C	2	4	96	5
T	5	1	3	94

Table 7.1: Frequencies of $S0 = i$ and $S1 = j$ in 400 site sequence comparison

   a. Compute the Jukes-Cantor distance between the sequences, showing all steps.

b. Compute the Kimura 2-parameter distance between the sequences, showing all steps.

c. Are the answers to (a) and (b) identical? Explain.

3. Ancestral and descendant sequences of 400 bases were simulated according to the Kimura 2-parameter model with $\beta/\gamma = 5$. A comparison of aligned sites gave the frequency data in Table 7.2.

$S0 \diagdown S1$	A	G	C	T
A	92	15	2	2
G	13	84	4	4
C	0	1	77	16
T	4	2	14	70

Table 7.2: Frequencies of $S1 = i$ and $S0 = j$ in 400 site sequence comparison

a. Compute the Jukes-Cantor and Kimura 2-parameter distances, showing all steps.

b. Which of these is likely to be a better estimate of the number of substitutions per site that actually occurred? Explain.

4. Compute the Kimura 3-parameter and log-det distances for the sequences of the last two problems. Why would these distances be less appropriate for these data?

5. Reasoning from the *formula* for the Jukes-Cantor distance, answer the following:

a. If two sequences are identical, why will $d_{JC} = 0$?

b. If two sequences differ in 3/4 or more of the sites, why will d_{JC} not make sense? Should this cause problems when trying to use the formula on real data?

c. If two sequences differ in just under 3/4 of the sites, why will the value of d_{JC} be very large?

6. The Jukes-Cantor distance formula is sometimes given as

$$d_{JC} = -\frac{3}{4} \ln \left(\frac{4q - 1}{3} \right),$$

where q is the proportion of bases that are the same in the two sequences. Show this this formula is equivalent to the one in the text.

7. When transitions are more frequent than transversions, the Kimura 2-parameter distance often gives a larger value than the Jukes-Cantor distance when applied to the same pair of sequences. Explain this informally by explaining why hidden mutations are more likely under this circumstance.

8. Suppose that the Jukes-Cantor model perfectly describes sequence evolution along a metric tree T with positive edge lengths. The rate parameter $\alpha = 1$ is fixed in the Jukes-Cantor rate matrix Q. Each edge e_i has length t_i, with associated Markov matrix $M_i = e^{Qt_i}$.

   a) Suppose a path from taxon S0 to taxon S1 in a tree is composed of edges of length $t_1, t_2, \ldots, t_n$. Show that if sequence data is exactly described by the distribution the Jukes-Cantor model predicts, then $d_{JC}(S0, S1) = \sum_{i=1}^{n} t_i$, and thus that the Jukes-Cantor distance agrees with the tree metric.

   b) What if the rate parameter α is some number other than 1? How will the tree metric and the Jukes-Cantor distance between taxa be related?

9. Show that the formula for the Jukes-Cantor distance can be recovered from the formula for the Kimura 2-parameter distance by letting b, e be appropriate expressions involving a.

10. Derive the formula for the Kimura 3-parameter distance. Use the result of Exercise 25 of Chapter 6, in which you found the entries b, c, and d in $M = e^{Qt}$ were given in terms of β, γ, δ, t by the formulae:

$$b = \frac{1}{4}\left(1 - e^{-(2\beta+2\delta)t} + e^{-(2\gamma+2\delta)t} - e^{-(2\beta+2\gamma)t}\right),$$
$$c = \frac{1}{4}\left(1 + e^{-(2\beta+2\delta)t} - e^{-(2\gamma+2\delta)t} - e^{-(2\beta+2\gamma)t}\right),$$
$$d = \frac{1}{4}\left(1 - e^{-(2\beta+2\delta)t} - e^{-(2\gamma+2\delta)t} + e^{-(2\beta+2\gamma)t}\right).$$

11. The goal of this problem is to show that the Jukes-Cantor distance is a special case of the log-det distance. You will need to know the following two facts about determinants of $k \times k$ matrices:

    i) $\det(cA) = c^k \det(A)$.

    ii) $\det(A) =$ the product of A's k eigenvalues.

    a. Suppose the Jukes-Cantor model with $\alpha = 1$ and edge length t exactly describes the evolution of a sequence S0 to S. Explain why the character distribution for the two sequences is $F = \frac{1}{4}M(t)$.

    b. Explain why $\mathbf{f}_1 = \mathbf{f}_2 = (1/4, 1/4, 1/4, 1/4)$.

    c. Use the facts above to show that in this case $d_{LD}(\text{S0}, \text{S1}) = d_{JC}(\text{S0}, \text{S1})$.

12. Proceeding as in the last problem, show that the Kimura 3-parameter distance is a special case of the log-det distance.

13. Show the log-det distance formula is additive and symmetric through the following steps. You will need to know the following three facts about determinants of $k \times k$ matrices:

    i) $\det(AB) = \det(A)\det(B)$.

ii) If D is a $k \times k$ diagonal matrix, then

$$\det(D) = D(1,1) \cdot D(2,2) \cdots D(k,k).$$

iii) $\det(A^T) = \det(A)$.

a. Suppose S0 is the parent of S1 which is the parent of S2, the initial base distribution for S0 is $\mathbf{p}_0$, and Markov matrices describing base substitutions are $M_{0\to1}$ and $M_{1\to2}$, respectively. Let $M_{0\to2} = M_{0\to1}M_{1\to2}$. Explain why $\mathbf{p}_1 = \mathbf{p}_0 M_{0\to1}$ and $\mathbf{p}_2 = \mathbf{p}_1 M_{1\to2}$ are the base distributions in S1 and S2 respectively, and explain the meaning of $M_{0\to2}$.

b. For the vector $\mathbf{p}_i = (a, b, c, d)$ let

$$D_i = \begin{pmatrix} \sqrt{a} & 0 & 0 & 0 \\ 0 & \sqrt{b} & 0 & 0 \\ 0 & 0 & \sqrt{c} & 0 \\ 0 & 0 & 0 & \sqrt{d} \end{pmatrix}.$$

Then for each pair i, j with $0 \le i < j \le 2$, define the matrix

$$N_{i\to j} = D_i M_{i\to j} D_j^{-1}.$$

Show $N_{0\to1}N_{1\to2} = N_{0\to2}$, and use fact (i) to conclude

$$\ln|\det(N_{0\to1})| + \ln|\det(N_{1\to2})| = \ln|\det(N_{0\to2})|.$$

c. Show the distribution array of characters on Si and Sj is $F_{i\to j} = D_i N_{i\to j} D_j$, and then use fact (i) to show

$$\ln|\det(F_{i\to j})| = \ln|\det(N_{i\to j})| + \ln|\det(D_i)| + \ln|\det(D_j)|.$$

d. Combine (b), (c), and fact (ii) to show the log-det distance is additive.

e. Explain why $F_{j\to i} = F_{i\to j}^T$, and then use fact (iii) to show the log-det distance is symmetric.

14. Suppose Q is a time-reversible rate matrix with stable base distribution $\mathbf{p}$.

a. If you have not already, do Exercise 30 of Chapter 6.

b. Since for a one-edge tree of length t this model predicts a joint distribution matrix $P = \mathrm{diag}(\mathbf{p})e^{Qt}$, explain why $-\mathrm{Tr}(\mathrm{diag}(\mathbf{p})\ln(\mathrm{diag}(\mathbf{p})^{-1}P)$ gives the expected number of substitutions per site on the edge. (Here $\log M$ denotes the matrix logarithm, the inverse of the matrix exponential, which can be defined through applying the usual Taylor series for the natural logarithm to a matrix.)

c. To define a GTR distance, we should apply the formula obtained in part (b) to an estimate of the joint distribution P. Explain why a reasonable estimate for P is $\hat{P} = (1/2)(F + F^T)$ where F is the 4×4 table of observed frequencies of base pairs in the two sequences.

Chapter 8

Maximum Likelihood

There are two dominant statistical paradigms in common use for inference of phylogenetic trees: Maximum Likelihood, and Bayesian analysis. Although the differences in these method can be viewed as profound philosophical ones, in fact both have much in common. They both assume a probabilistic model of the evolution of sequences on trees, and then attempt to find the tree(s) and model parameters that are most in accord with the data. Though exactly what this means varies between the methods, implementing them involves many of the same calculations. These calculations are, unfortunately involved enough that it is not feasible to work out any interesting analysis by hand, so specialized software is needed.

The Maximum Likelihood and Bayesian frameworks are both very general approaches, used across all disciplines, for statistical inference. Since typical introductory statistics courses often omit introducing either as a general method, scientists often learn of them informally, in specialized settings, without gaining much understanding of exactly what they mean, what calculations must be performed for inference, or how they differ. Though our goal in these notes is to develop phylogenetic inference, we will first step back to viewing much simpler statistical inference questions to try to make the methods clearer. We begin with Maximum Likelihood.

8.1 Probabilities and Likelihoods

Suppose we have a probabilistic model that we believe predicts outcomes of some experiment. Our model depends on one or more *parameters*, which are typically numerical quantities. However, we do not know what values of these parameters are appropriate to fit our data. How can we infer a 'best' choice of parameter values from experimental data?

We'll give two examples of such a problem, the first as simple as possible, the second a bit more complex, in order to motivate the ML approach to addressing such an issue.

Example (1). Our experiment is the toss of a (possibly unfair) coin. Our model is simply that the toss will produce 'heads' with probability p and 'tails' with probability $1 - p$. The parameter here is p, which might have any numerical value from 0 to 1.

We conduct many i.i.d. trials of this experiment, obtaining a sequence of 'heads' and 'tails'. While we either know or assume our model applies to this experiment, we do not know the value of the parameter p. How should we estimate p?

For Example (1), we do not need to be very sophisticated because the model is so simple. Using a frequentist interpretation of probability, p simply expresses what fraction of a large number of trials we believe will produce heads. So if, for instance, out of 100 trials we record 37 heads, we estimate $\hat{p} = 37/100 = .37$. In general, if we find m heads out of n trials, we estimate $\hat{p} = m/n$.

But there is another way we can arrive at the same result. Naively, we might decide the best estimate for p would be the numerical value that is most probable, given that we obtained the specific data we did. That is, we want $\hat{p}$ to be the value of p that maximizes

$$\mathcal{P}(p \mid \text{data}).$$

Unfortunately, it isn't at all clear how to do this, since we have no idea how to compute such a conditional probability, or even whether it makes sense. As we've seen previously for phylogenetic models, $\mathcal{P}(\text{data} \mid p)$ can be calculated, but that is not what we are interested in here.

However, we can observe that by formal properties of probabilities

$$\mathcal{P}(p \mid \text{data})\mathcal{P}(\text{data}) = \mathcal{P}(p,\ \text{data}) = \mathcal{P}(\text{data} \mid p)\mathcal{P}(p). \tag{8.1}$$

This can also be written as

$$\mathcal{P}(p \mid \text{data}) = \mathcal{P}(\text{data} \mid p)\frac{\mathcal{P}(p)}{\mathcal{P}(\text{data})}, \tag{8.2}$$

which is an instance of *Bayes' theorem* relating conditional probabilities. Now the terms $\mathcal{P}(p)$ and $\mathcal{P}(\text{data})$ in the fraction on the right are not things we can address[1]. In fact, from a traditional, frequentist viewpoint, it doesn't even make sense to talk about $\mathcal{P}(p)$, since p represents some fixed, though unknown, parameter value, and is not random in any way. As for $\mathcal{P}(\text{data})$, since we have observed the data we might say it has probability 1. However, from equation (8.1) we see it is really intended to mean the probability of the data *before* p is specified, and this is again nonsensical. Thus equation (8.2) has no rigorous meaning from a frequentist perspective, and is best considered as a motivational guide.

[1] In fact, taking these terms into account *is* done in a Baysian analysis, and is the primary difference between the frameworks.

However, proceeding informally, if we are interested in finding a value of p to make the left hand side of equation (8.2) large, we might choose to focus on the first term appearing on the right, as this *is* something we can calculate from our model. We call this function L, the *likelihood function* for our model and data:

$$L(p) = L(p \mid \text{data}) = \mathcal{P}(\text{data} \mid p)$$

Note that while the likelihood function is a conditional probability, it is *not* the conditional probability we originally were interested in. We insist on referring to it as a likelihood, rather than a probability, to remind us it is most definitely not telling us the probability of any p given the data.

Definition. Given some data presumed to be in accord with a model, a *maximum likelihood estimate* for the set of parameters p for a model is a set of values $\hat{p}$ that maximizes the likelihood function $L(p \mid \text{data})$.

Put another way, a maximum likelihood estimate of the model parameters is a choice of parameters that would make it most probable to produce the data we collected.

Note that our definition does not refer to *the* maximum likelihood estimate, since it is possible that more than one choice of $\hat{p}$ produces the maximum value of $L(\text{data} \mid p)$. We generally hope for and expect a single ML estimate $\hat{p}$, but it is possible that there is more than one.

To make this more concrete, let's find the maximum likelihood estimate for p in Example (1), the coin toss experiment. Suppose our data are m heads out of n tosses, in some particular order. Then the likelihood function, which depends on the unknown p, is $L(p) = L(p \mid \text{data}) = \mathcal{P}(\text{data} \mid p)$. But, because we assume our tosses are independent, this probability is simply the product of the probabilities of the outcomes of each individual toss.[2] With m heads and $n - m$ tails we have

$$L(p) = \mathcal{P}(\text{data} \mid p) = p^m(1-p)^{n-m}.$$

To find the maximum in the interval $[0, 1]$, we use calculus, and compute

$$\begin{aligned}\frac{d}{dp}p^m(1-p)^{n-m} &= mp^{m-1}(1-p)^{n-m} - (n-m)p^m(1-p)^{n-m-1} \\ &= p^{m-1}(1-p)^{n-m-1}\left(m(1-p) - (n-m)p\right).\end{aligned}$$

But this is zero exactly when

$$0 = m(1-p) - (n-m)p = m - np.$$

[2]This assumes we are referring to a specific ordering of heads and tails in the data sequence. If the order is discarded, and we refer only to the number of heads and tails, then the probability should be increased by a factor of $\binom{n}{m}$, to account for the many possible orders. But this extra constant factor has no effect on determining the value of p at which the likelihood is maximal.

Thus we find $\hat{p} = m/n$, exactly as before.

In computing maximum likelihood estimates of model parameters, it is often simpler to consider the logarithm of the likelihood function (often called the *log-likelihood*), rather than the likelihood function itself. The likelihood and log-likelihood are maximized at the same parameter values, so this does not affect our judgment of the best estimates to infer. However, the form of the log-likelihood function often makes the computation of the maximizer simpler, as in fact it does even for Example (1):

$$\ln L(p) = m \ln p + (n - m) \ln(1 - p),$$

so

$$\frac{d}{dp} \ln L(p) = \frac{m}{p} - \frac{n - m}{1 - p}.$$

Setting this equal to zero and solving for p again yields $\hat{p} = m/n$.

We now consider a more elaborate example, where the parameter estimate we should use is not quite so obvious.

Example (2). Suppose we have two (possibly unfair) coins, with probabilities of heads p_1, p_2 respectively, giving us two parameters. We toss the first coin, and depending on whether it gives heads or tails, either retain it, or switch to the second coin for a second toss. We then make a second toss and report the (ordered) result of these two tosses as the outcome of the experiment.

Then we find the probability of the various outcomes are

$$p_{hh} = p_1^2, \quad p_{ht} = p_1(1 - p_1), \quad p_{th} = (1 - p_1)p_2, \quad p_{tt} = (1 - p_1)(1 - p_2).$$

Now we wish to estimate p_1 and p_2 from some data: Suppose in n trials, we find $n_{hh}, n_{ht}, n_{th}, n_{tt}$ occurrences of the 4 outcomes, where

$$n = n_{hh} + n_{ht} + n_{th} + n_{tt}.$$

Before taking an ML approach, we might hope to just set $p_{ij} = n_{ij}/n$ for each $i, j \in \{h, t\}$ and solve for p_1, p_2. However, this is not likely to work, since it gives 3 independent equations in 2 unknowns. (While at first it appears we have 4 equations, the fact that $\sum_{i,j \in \{h,t\}} p_{ij} = 1$ means one of the equations is implied by the others.) But for a system of 3 equations in only 2 unknowns, we generally do not expect a solution to exist. Indeed, for most data there will be no solutions to this polynomial system.

Taking a maximum likelihood approach to the estimation of p_1, p_2, however, is straightforward. Using the assumption that the trials are i.i.d., the likelihood function is

$$\begin{aligned} L(p_1, p_2) &= (p_1^2)^{n_{hh}} \left(p_1(1 - p_1)\right)^{n_{ht}} \left((1 - p_1)p_2\right)^{n_{th}} \left((1 - p_1)(1 - p_2)\right)^{n_{tt}}, \\ &= p_i^{2n_{hh} + n_{ht}} (1 - p_1)^{n_{ht} + n_{th} + n_{tt}} p_2^{n_{th}} (1 - p_2)^{n_{tt}}. \end{aligned}$$

Thus the log-likelihood is

$$\ln L(p_1, p_2) = (2n_{hh}+n_{ht}) \log p_1 + (n_{ht}+n_{th}+n_{tt}) \log(1-p_1) + n_{th} \log p_2 + n_{tt} \log(1-p_2).$$

To find maximizers we compute partial derivatives with respect to p_1 and p_2, equate them to zero, and see

$$0 = \frac{2n_{hh} + n_{ht}}{p_1} - \frac{n_{ht} + n_{th} + n_{tt}}{1 - p_1}, \quad 0 = \frac{n_{th}}{p_2} - \frac{n_{tt}}{1 - p_2}.$$

Solving for p_2 gives

$$\hat{p}_2 = \frac{n_{th}}{n_{th} + n_{tt}}, \tag{8.3}$$

which is a formula that, in retrospect, we could have made up as a reasonable estimator. It is simply the proportion of trials in which the second coin was used (because the first coin produced a tail) that produced a head for the second coin.

Solving for p_1 gives

$$\hat{p}_1 = \frac{2n_{hh} + n_{ht}}{2n_{hh} + 2n_{ht} + n_{th} + n_{tt}}. \tag{8.4}$$

We can also see that this formula is reasonable: The denominator represents the total number of times the first coin was tossed, and the numerator is the number of these that produced a head.

We leave as an exercise checking that for 'perfect' data, where $n_{ij} = np_{ij}$, these formulas recover the correct values $\hat{p}_1 = p_1$, $\hat{p}_2 = p_2$.

This example has shown an instance of a common phenomenon, that when ML estimators are computed for simple models, they usually are given by formulas that are intuitively reasonable. For simple models we don't necessarily need the ML framework to justify these estimators, since we could reason in other ways. For more complicated models, where it is less clear how to come up with any intuitive estimation formulas, we find it reassuring that ML offers a general procedure extending our intuition.

In addition, ML estimators have important statistical properties as well. For instance, it's possible to prove in great generality that maximum likelihood estimators are *statistically consistent*. This means that if the chosen model accurately describes our experiment, then as the number of trials is increased to infinity, the estimators converge to the true parameters. ML estimators are also *asymptotically efficient*, in that they have minimal variance as the number of trials is increased. Note both of the statements refer to having large amounts of data, though, and say nothing about behavior for small data sets. In fact, there are some statistical inference problems for which ML is known to be quite poorly behaved for small data sets. However ML is a widely-accepted inference framework, and its use is quite common throughout statistics.

On a more practical level, there are often difficult computational issues involved in ML. In the examples above, we have found maximum likelihood estimators by first computing derivatives of the log-likelihood function, and then

solving several simultaneous equations. This last step in particular may be quite difficult for more complicated models. For instance, even if the model is expressed through polynomial equations, we may obtain rational expressions of high degree from the partial derivatives. If the model is given by transcendental formulas, we may be forced to solve transcendental equations. In practice, then, numerical approaches to finding approximate maxima must be used. (Often these are based on the same idea as *Newton's method*, which you may be familiar with from Calculus.) Techniques for numerically maximizing general function are highly developed, and form an entire subfield of applied mathematics called Optimization. These approaches do not give formulae for ML estimators, but do allow the development of software that will produce numerical approximations of the estimators for any data.

It is also possible for a likelihood function to have several local maxima. (In the case of a single parameter, this simply means the graph of the likelihood function has several 'bumps'.) If these local maxima are all located, then the values of the likelihood function at them must be compared in order to choose the global maximum as the ML estimator. Numerical maximization schemes, however, are usually unable to ensure that the global maximum has been located. Searches may become trapped in local maxima, so some effort must be taken to try to prevent this from happening. Often searches are performed from several different starting points, in hopes that all local maxima may be found.

Before turning back to phylogenetics, consider a final example of ML inference, this time in a biological setting:

Example (3). A population of a diploid organism has two alleles for a particular gene, which we denote A and a. If the population is in Hardy-Weinberg equilibrium, and the frequency of the alleles are p_A and $p_a = 1-p_A$, then the genotypes of individuals in the populations should have the following frequencies:

$$AA : p_A^2, \quad Aa : 2p_Ap_a, \quad aa : p_a^2.$$

If in a random sample of n individuals in the population, we have n_{AA}, n_{Aa}, and n_{aa} individuals with these genotypes, what are the ML estimates of p_A and p_a?

Notice first the naive approach of simply setting theoretical genotype frequencies to empirical ones give the equations:

$$\begin{aligned} p_A^2 &= \frac{n_{AA}}{n}, \\ 2p_A(1-p_A) &= \frac{n_{Aa}}{n}, \\ (1-p_A)^2 &= \frac{n_{aa}}{n}, \end{aligned}$$

and this system of 3 equations in 1 unknown is unlikely to be solvable. However, the log-likelihood is

$$\begin{aligned} \ln L(p_A) &= n_{AA}\ln(p_A^2) + n_{Aa}\ln(2p_A(1-p_A)) + n_{aa}\ln((1-p_A)^2) \\ &= n_{AA}2\ln(p_A)) + n_{Aa}(\ln 2 + \ln(p_A) + \ln(1-p_A))) + n_{aa}2\ln(1-p_A). \end{aligned}$$

Differentiating with respect to p_A and setting equal to 0 yields

$$0 = \frac{2n_{AA}}{p_A} + \frac{n_{Aa}}{p_A} - \frac{n_{Aa}}{1-p_A} - \frac{2n_{aa}}{1-p_A},$$

so

$$\frac{2n_{AA}+n_{Aa}}{p_A} = \frac{n_{Aa}+2n_{aa}}{1-p_A},$$

Solving for p_A then yields

$$p_A = \frac{2n_{AA}+n_{Aa}}{2(n_{AA}+n_{Aa}+n_{aa})} = \frac{2n_{AA}+n_{Aa}}{2n}. \tag{8.5}$$

With a little thought, this formula should be intuitively reasonable.

8.2 ML Estimators for One-edge Trees

As a first application of maximum likelihood ideas to phylogenetics, consider a Jukes-Cantor model on a one-edge tree, from an ancestral sequence S_0 to a descendant sequence S_1. Let the length of the edge be t, measured in units so that $\alpha = 1$ in the Jukes-Cantor rate matrix.

Then the Markov matrix along the edge is

$$M = e^{Qt} = \begin{pmatrix} 1-a & a/3 & a/3 & a/3 \\ a/3 & 1-a & a/3 & a/3 \\ a/3 & a/3 & 1-a & a/3 \\ a/3 & a/3 & a/3 & 1-a \end{pmatrix}$$

where

$$a = a(t) = \frac{3}{4}\left(1 - e^{-\frac{4}{3}t}\right),$$

while we have a uniform base distribution

$$p_0 = \begin{pmatrix} 1/4 & 1/4 & 1/4 & 1/4 \end{pmatrix}$$

in the ancestral sequence S_0.

We are interested in estimating the parameter t from aligned sequence data for S_0 and S_1. We summarize the sequence data by counting how often each pattern appears, and let $N(i,j) = n_{ij}$ be the number of sites with base i in S0 and base j in S1. This N is a 4×4 matrix of data counts, such as the ones given in Table 6.2.

We have a corresponding joint distribution matrix of probabilities of bases predicted by the model, with

$$\begin{aligned} P = (p_{ij}) &= \mathrm{diag}(p_0)M \\ &= \begin{pmatrix} (1-a)/4 & a/12 & a/12 & a/12 \\ a/12 & (1-a)/4 & a/12 & a/12 \\ a/12 & a/12 & (1-a)/4 & a/12 \\ a/12 & a/12 & a/12 & (1-a)/4 \end{pmatrix}. \end{aligned}$$

Here every p_{ij} is a function of the parameter t through the formula for $a(t)$. Thus the likelihood function is

$$L(t \mid \{n_{ij}\}) = \prod_{i,j=1}^{4} p_{ij}^{n_{ij}},$$

and the log-likelihood is

$$\ln L(t \mid \{n_{ij}\}) = \sum_{i,j=1}^{4} n_{ij} \ln p_{ij} = \ln(a/12) \sum_{i \neq j} n_{ij} + \ln((1-a)/4) \sum_{i} n_{ii}.$$

To find the maximizer $\hat{t}$, we differentiate the log-likelihood with respect to t and set the result equal to 0:

$$0 = \frac{\sum_{i \neq j} n_{ij}}{a(\hat{t})} a'(\hat{t}) - \frac{\sum_i n_{ii}}{1 - a(\hat{t})} a'(\hat{t}).$$

Since $a'(t) \neq 0$ for any value of t, we may divide by $a'(\hat{t})$, and with a little algebra obtain

$$a(\hat{t}) = \frac{\sum_{i \neq j} n_{ij}}{\sum_{i,j} n_{ij}}.$$

This should not be too surprising, since $a(t)$ described the proportion of sites in which we expect to observe a substitution, and the formula for $a(\hat{t})$ computes the proportion of sites in which we actually observe one, *i.e.*, the Hamming distance.

Maximum likelihood has now given us a firm justification for the what we did in Chapter 7 when we defined the Jukes-Cantor distance: We set the formula for $a(\hat{t})$ derived from the model equal to the Hamming distance between the sequences. Solving for $\hat{t}$ led us to the Jukes-Cantor distance formula. But now interpreting that work in the light of maximum likelihood, we see that the Jukes-Cantor distance is actually the ML estimate for the edge length t under the Jukes-Cantor model.

Using maximum likelihood to estimate the time parameter under the Kimura models, and other more complicated models, can be handled similarly, so we leave them for exercises. The resulting formulas for the parameter estimate match with the distance formulas we've given previously. This places all of those distances on firmer theoretical footing, as arising from the ML framework applied to 2-taxon trees.

8.3 Inferring Trees by ML

So how does maximum likelihood give us a framework for phylogenetic inference of trees? Suppose we have aligned sequence data for the n taxa in a set X, and we assume a particular model of molecular evolution.

To be concrete, we might use a general time-reversible model with a common rate matrix Q for all edges, and a root base distribution which is an eigenvector of Q with eigenvalue 0. With the root distribution given by

$$\mathbf{p} = \begin{pmatrix} p_A & p_G & p_C & p_T \end{pmatrix},$$

we let

$$Q = \begin{pmatrix} * & p_G\alpha & p_C\beta & p_T\gamma \\ p_A\alpha & * & p_C\delta & p_T\epsilon \\ p_A\beta & p_G\delta & * & p_T\eta \\ p_A\gamma & p_G\epsilon & p_C\eta & * \end{pmatrix},$$

where the diagonal entries are chosen so rows sum to zero. We can also choose, say, $\eta = 1$ to fix a time-scale. The parameters for our model are 1) a binary phylogenetic X-tree T, 2) any 3 of the entries of $\mathbf{p}$, 3) $\alpha, \beta, \gamma, \delta, \epsilon$, and 4) the edge lengths $\{t_e\}_{e \in E(T)}$. There are additional restrictions on parameter values so that edge lengths, the root distribution entries, and the off diagonal entries of Q are non-negative, but we will not be explicit about them here.

Notice that the tree itself is a parameter — a non-numerical one, but a parameter nonetheless. When we attempt to maximize the likelihood function, we will have to do so over all allowable numerical parameters as well as the discrete variable of the tree topology.

We thus consider a log-likelihood function for each fixed tree T,

$$\ln L_T = \sum_{(i_1,\ldots,i_n) \in \{A,G,C,T\}^n} n(i_1, \ldots, i_n) \ln(p(i_1, \ldots, i_n)), \tag{8.6}$$

where $p(i_1, \ldots, i_n)$ is a function of the numerical parameters giving the probability of observing the pattern $(i_1, \ldots, i_n)$ at a site (as computed in Chapter 6), and $n(i_1, \ldots, i_n)$ is the count of this pattern in the aligned sequence data. We emphasize that $\ln L_T$ is a function of all the numerical model parameters associated to T — essentially a function of the variable entries of $\mathbf{p}_\rho$, Q, and $\{t_e\}_{e \in E(T)}$.

We now need to find the values of the numerical parameters that maximize L_T. In practice, for more than a few taxa this will have to be done by numerical methods rather than by exact solution.

Once we have maximized L_T for a fixed T, however, we will have to do the same for all other T, comparing the maximum values we found for each. We then pick the tree T and the numerical parameter values that maximize its likelihood function as the ML estimators for our data.

As should be obvious, these computation are simply too involved to be done by hand, even in a toy problem for instructional purposes. Armed with a computer, we still have a tremendous amount of work to do, optimizing numerical parameters for each fixed tree, as we search among all trees. Heuristic approaches are necessary to make the calculations tractable. We will seldom be absolutely sure we have found the true maximum, and will be limited in how

many taxa we can deal with by the power of our computer and the design of the software.

When maximum likelihood is used for tree inference, it is typical to assume a model such as GTR (or an extension of it with rate variation, as will be discussed in Chapter 10). There are practical reasons for this, since using a common rate matrix on all edges and not needing to consider a variable root location keeps the number of parameters lower, making the optimization much more tractable. Sometimes the common rate matrix and stable base distribution assumptions of GTR are reasonable on biological grounds as well. If they are, then using a model with fewer parameters (e.g., HKY) may be desirable instead, since too general a model risks 'overfitting' the data.

To focus on only one issue with always following the typical approach, however, note that there are data sets for which it is clear the base distribution is not stable. Standard software implementations of ML include only models assuming stability, and these may lead to erroneous inference if the assumption is violated. However, a literature search will turn up special-purpose software in which researchers have introduced models allowing some types of changing base distributions.

The important lesson is ML can only be expected to perform well if the model assumptions are at least approximately correct. If the model used in data analysis does not give a good rough description of the evolution of the sequences, using ML for inference does not guarantee any good results.

8.4 Efficient ML Computation

For software to implement ML efficiently for phyologenetics, there are several problems to ovecome. Suppose we have a fixed set of N taxa, aligned data sequences for them, and have chosen a model for our analysis. (We'll imagine we are using the GTR model here.) We would like to perform the following steps:

1. Count the number $n(i_1, \ldots i_n)$ of occurrences of each pattern of bases in the aligned sequences, since this will be needed in the likelihood functions as shown in equation (8.6).

2. Consider all possible trees T that might relate the taxa. (For the GTR model, we may use unrooted trees, since time-reversibility ensures moving the location of the root doesn't affect the probability of observing any data.)

3. For each such tree T, construct the likelihood function as given in equation (8.6). (For GTR, this is a function of the root distribution (3 free parameters) and the relative rates $\alpha, \beta, \gamma, \delta, \epsilon, \eta$ (5 free parameters, since we may arbitrarily choose one of these to be 1), and all edge lengths $\{t_e\}_{e \in E_T}$ (2N-3 free parameters). The Markov matrix on each edge of the tree must

be computed as $M_e = e^{Qt_e}$ in order to use its entries in the computation of the $p(i_1, \ldots i_n)$.

4. For each tree's likelihood function constructed in step (3), find the maximum value, and values of all the parameters that produce this maximum.

5. Finally, choose the tree T, and the numerical parameters for it, that had the largest maximum, and report this as the ML tree. In some cases, we may get a tie between several trees, though this is quite rare in practice.

Step (1) here is straightforward and can be done quickly.

For step (2), we have the same problem that we had for Maximum Parsimony; if N is large, then the $(2N-5)!!$ trees to be considered is a huge number.

Step (3) looks difficult, since for DNA there are 4^N possible patterns, so the likelihood function looks like it might have a huge number of terms. However, we do not need to include terms for patterns that don't appear in the sequences, since for these $n(i_1, \ldots, i_n) = 0$. The number of different patterns that appear is generally much smaller than 4^N, so this observation can save much work. However, we need to find a way to compute $p(i_1, \ldots, i_n)$ efficiently for those patterns that do appear.

Step (4) is difficult, but optimization problems of all sorts have been heavily studied by mathematicians and computer scientists, so there are good approaches that generally perform well.

We will discuss approaches to dealing with step (2) in Chapter 9. Step (4) is a subject for an entire course in itself. But for step (3), the *Felsenstein pruning algorithm* can be used to compute the pattern probabilities efficiently.

But before given Felsenstein's algorithm, we should ask why we don't just compute the pattern probabilities as discussed in Chapter 6? There we saw we can compute the probability of observing a given pattern by a sum of terms, one for each possible assignment of states to each internal node of the tree. The individual terms were products of entries of Markov matrices on each edge, and an entry of the root distribution vector. Although conceptually clear, the problem with this approach is that this expression has too many terms. For N taxa, there are $N-1$ internal nodes in a rooted tree, so there would be 4^{N-1} terms in this sum. Since this is exponential in N, it is quite large as soon as N is large. The key feature of Felsenstein's algorithm is that it uses many fewer operations to compute the same probability — the number of operations is only linear in N rather than exponential.

Felsenstein's algorithm for computing $p(i_1, \ldots, i_n)$ is formally similar to that of Sankoff, for computing weighted parsimony scores, and fits within a standard dynamic programming approach common in computer science. It works with a rooted tree, proceeding from the leaves toward the root, performing a computation at each node. We suppose we already have a formula for the Markov matrix M_e for each edge of the tree (which would have been found by matrix exponetiation in the case of the GTR model).

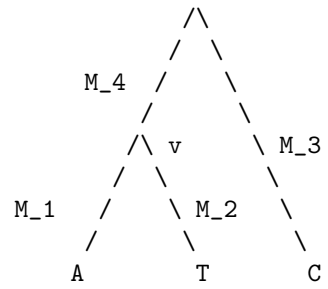

Figure 8.1: A small tree to illustrate Felsenstein's pruning algorithm

We begin with a very small example, with only three leaves, as shown in Figure 8.1. Suppose we are interested in computing $\mathcal{P}(\texttt{ATC})$, the joint probability of observing `A` at the first leaf, `T` at the second, and `C` at the third, as shown in the figure.

Using our standard order of $\texttt{A}, \texttt{G}, \texttt{C}, \texttt{T}$ for bases, we assign probability vectors of $\mathbf{c}_1 = (1, 0, 0, 0)$, $\mathbf{c}_2 = (0, 0, 0, 1)$, $\mathbf{c}_3 = (0, 0, 1, 0)$ to the leaves from left to right, since with probability 1 the specified base must appear. Now at v we compute a 4-entry vector $\mathbf{c}_v$ giving the conditional probability of what appears at the leaves below it, conditioned on the state at v. For instance, if an `A` appeared at v, the first two leaves would have pattern `AT` with probability

$$\begin{aligned}\mathbf{c}_v(A) &= M_1(\texttt{A},\texttt{A})M_2(\texttt{A},\texttt{T}) \\ &= (M_1(\texttt{A},\texttt{A})\cdot 1 + M_1(\texttt{A},\texttt{G})\cdot 0 + M_1(\texttt{A},\texttt{C})\cdot 0 + M_1(\texttt{A},\texttt{T})\cdot 0)) \\ &\qquad ((M_2(\texttt{A},\texttt{A})\cdot 0 + M_2(\texttt{A},\texttt{G})\cdot 0 + M_1(\texttt{A},\texttt{C})\cdot 0 + M_1(\texttt{A},\texttt{T})\cdot 1).\end{aligned}$$

Similarly, if a `G` appeared at v, the first two leaves would have pattern `AT` with probability

$$\begin{aligned}\mathbf{c}_v(\texttt{G}) &= M_1(\texttt{G},\texttt{A})M_2(\texttt{G},\texttt{T}) \\ &= (M_1(\texttt{G},\texttt{A})\cdot 1 + M_1(\texttt{G},\texttt{G})\cdot 0 + M_1(\texttt{G},\texttt{C})\cdot 0 + M_1(\texttt{G},\texttt{T})\cdot 0)) \\ &\qquad ((M_2(\texttt{G},\texttt{A})\cdot 0 + M_2(\texttt{G},\texttt{G})\cdot 0 + M_1(\texttt{G},\texttt{C})\cdot 0 + M_1(\texttt{G},\texttt{T})\cdot 1).\end{aligned}$$

The entries $\mathbf{c}_v(\texttt{C})$ and $\mathbf{c}_v(\texttt{T})$ are given by similar formulas.

Notice that the computation of all entries of $\mathbf{c}_v$ can be more easily presented as follows: Compute the vectors

$$\mathbf{w}_1 = M_1\mathbf{c}_1^T$$

and

$$\mathbf{w}_2 = M_2\mathbf{c}_2^T.$$

Then $\mathbf{c}_v$ is the *element-wise* product of $\mathbf{w}_1$ and $\mathbf{w}_2$; that is $\mathbf{c}_v(i) = \mathbf{w}_1(i)\mathbf{w}_2(i)$.

Now that $\mathbf{c}_v$ is computed, similar reasoning shows we can compute $\mathbf{c}_\rho$ by letting

$$\mathbf{w}_1 = M_4 \mathbf{c}_v^T$$
$$\mathbf{w}_2 = M_3 \mathbf{c}_3^T,$$

and then computing the element-wise product of these vectors. To check that the ith entry of $\mathbf{c}_\rho$ is given correctly by this calculation, first note that

$$\begin{aligned}\mathbf{w}_1(i) &= \sum_{j=1}^{4} M_4(i,j)\mathbf{c}_v(j) \\ &= \sum_{j=1}^{4} \mathcal{P}(v = j \mid \rho = i)\mathcal{P}(S_1 = \mathtt{A}, S_2 = \mathtt{T} \mid v = j) \\ &= \mathcal{P}(S_1 = \mathtt{A}, S_2 = \mathtt{T} \mid \rho = i).\end{aligned}$$

Since $\mathbf{w}_2(i) = \mathcal{P}(S_3 = \mathtt{C} \mid \rho = i)$, the entries of $\mathbf{c}_\rho$ are

$$\begin{aligned}\mathbf{c}_\rho(i) &= \mathcal{P}(S_1 = \mathtt{A}, S_2 = \mathtt{T} \mid \rho = i)\mathcal{P}(S_3 = \mathtt{C} \mid \rho = i) \\ &= \mathcal{P}(S_1 = \mathtt{A}, S_2 = \mathtt{T}, S_3 = \mathtt{C} \mid \rho = i).\end{aligned}$$

Here we use independence of the process on the two edges descending from ρ to justify the multiplication of the conditional probabilities.

Once we have found $\mathbf{c}_\rho$, we have the conditional probabilities of observing the given pattern at the leaves, conditioned on each possible state at the root. The final step in obtaining the probability of the pattern `ATC` uses the base distribution at the root: we simply compute

$$\mathbf{p}_\rho \mathbf{c}_\rho^T = p_A \mathbf{c}_\rho(\mathtt{A}) + p_C \mathbf{c}_\rho(\mathtt{C}) + p_G \mathbf{c}_\rho(\mathtt{G}) + p_T \mathbf{c}_\rho(\mathtt{T}).$$

For a larger tree, there are of course more steps to this computation, but it can still be performed quite quickly.

The Felsenstein algorithm also has a simple modification to deal with missing data. If no information on the base at a leaf is known, then one simply takes $\mathbf{c}$ to be $(1,1,1,1)$. If it was only known that the base was a purine, then one would set $\mathbf{c} = (1,1,0,0)$.

Mathematical Note: The relationship between the Sankoff weighted-parsimony algorithm and the Felenstein pruning algorithm goes much beyond the way they both proceed from the leaves toward the root. The entries in vectors at the leaves in the Sankoff algorithm are the the negative logarithms of those in the Felsenstein algorithm. The weight matrix of the Sankoff algorithm has as its analog in the Felsenstein algorithm the Markov matrix associated to an edge. Then, each addition in the Sankoff algorithm corresponds precisely to a multiplication in Felsenstein's. Each minimization in Sankoff's corresponds precisely to a sum in Felsenstein's. These observations can be summarized by saying the Sankoff algorithm is a *tropicalization* of Felsenstein's. While probabalistic models in phylogenetics can be studied via the branch of mathematics called *algebraic geometry*, parsimony approached fall under *tropical geometry*.

8.5 Exercises

1. When finding maxima of the likelihood functions in Examples (1) and (2) in the text, we ignored issues of whether these occurred at endpoints of the interval [0,1], or equivalently we ignored showing our unique critical points were in fact maxima. Correct this shortcoming in our presentation.

2. Show that if in Example (1) we have 'perfect' data from N trials, in the sense that $n = Np$, and $m = N(1-p)$, then the maximum likelihood estimate $\hat{p}$ recovers the true value of p.

3. Log-likelihoods values are always ≤ 0. Explain why.

4. Show that if in Example (2) we have 'perfect' data from N trials, in the sense that $n_{hh} = Np_1^2$, $h_{ht} = Np_1(1-p_1)$, $n_{th} = N(1-p_1)p_2$ and $n_{tt} = N(1-p_1)(1-p_2)$, then the maximum likelihood estimates $\hat{p}_1, \hat{p}_2$ recover the true values of p_1, p_2.

5. Check the algebra to obtain formula (8.5), and then give an intuitive (non-ML) explanation of why it is a reasonable estimator of an allele frequency.

6. Suppose a gene has 3 alleles, A_1, A_2, A_3 and is Hardy-Weinberg equilibrium in a diploid population. If the allele frequencies are $p_1, p_2, p_3 = 1 - p_1 - p_2$ respectively, then genotype frequencies should be

$$p_1^2,\ p_2^2,\ p_3^2,\ 2p_1p_2,\ 2p_1p_3,\ 2p_2p_3.$$

   Derive formulas for the ML estimates of the allele frequencies from counts of empirical genotypes.

7. Suppose the Kimura 2-parameter model is used to model describing the evolution of a sequence S_0 to S_1 along a single edge. View the entries b and c of the Markov matrix as the unknown parameters of the model. Guess reasonable formulas for estimators $\hat{b}$ and $\hat{c}$ in terms of the entries of the frequency array comparing the two sequences. Then find formulas for the maximum likelihood estimators. Do they agree?

8. Repeat the last exercise for the Kimura 3-parameter model.

9. Repeat the last exercise for the general Markov model.

10. For performing ML with the GTR model, the maximization of the likelihood function must be done only over the values of parameters that are biologically meaningful. Explicitly give the necessary restrictions on each of the GTR parameters.

11. For the Kimura 2-parameter rate matrix model, how many variables appear in the likelihood function for a particular binary tree T relating N taxa?

12. In Problem 6 of Section 6.5 you were asked to compute probabilities of observing certain base patterns on a specific 3-taxon tree with Jukes-Cantor rate Markov matrices on the edges. Redo that problem using the Felsenstein pruning algorithm.

13. Consider DNA evolution on the rooted tree $((A, B), (C, D))$, and suppose the same Kimura 2-parameter Markov matrix

$$\begin{pmatrix} .8 & .1 & .05 & .05 \\ .1 & .8 & .05 & .05 \\ .05 & .05 & .8 & .1 \\ .05 & .05 & .1 & .8 \end{pmatrix}$$

described base changes on every edge.

a) Use the Felsenstein pruning algorithm to compute the probability of observing the pattern $AGGT$.

b) Use the Felsenstein pruning algorithm to compute the probability of observing the pattern $GGAT$.

c) Which of the probabilities that you computed is larger? Is that what you should have expected?

14. At the end of section 8.4 there is a brief mention of how one can handle missing data in the Felsenstein pruning algorithm. Elaborate on this, by explaining in detail what conditional probabilities are computed by $M\mathbf{c}$, where M is the Markov matrix on the edge above the leaf, and $\mathbf{c}$ is the vector suggested for encoding an unknown purine.

Chapter 9

Tree Space

As discussed in Chapter 2, as soon as we have more than a handful of taxa we wish to relate, there are many potential trees to consider. While distance methods give us a quick way to produce one tree, both parsimony and maximum likelihood approaches to tree inference require that we consider all trees in choosing an optimal one to fit data. In practice, this is usually not possible to do in an acceptable amount of time. Instead, we imagine a 'space' of trees, and move from one to another to using some sort of search procedure, trying to find the optimal tree according to the chosen criterion.

In this chapter we will focus on binary topological trees only, with no edge lengths specified. Other notions of tree space, such as for rooted topological trees, or for either rooted or unrooted metric trees can also be developed. We emphasize the unrooted topological case here since it is the one the plays a conceptual role in most current tree inference software.

9.1 What is Tree Space?

Suppose we are interested in relating 4 taxa, by an unrooted tree, using parsimony. Then we know there are only three topological possibilities that are fully resolved, as shown in Figure 9.1.

It is convenient to think of these three possibilities as the only three points in a '4-taxon, binary, unrooted tree space.' As we look for the most parsimonious tree, we might begin at any of the three points and determine its parsimony score. Then we move to another point, and determine its score. Finally, we move to the third point, and determine its score. Having visited each point in tree space, we've then completed an exhaustive search, and can be sure which tree of the three is most parsimonious.

With only 3 points in this tree space, this language of moving between points in a space may seem artificial; we could just say we need to compute scores for every tree. However, for 100 taxa, performing an exhaustive search among the 195!! binary, unrooted, topological trees might be too time consuming to do

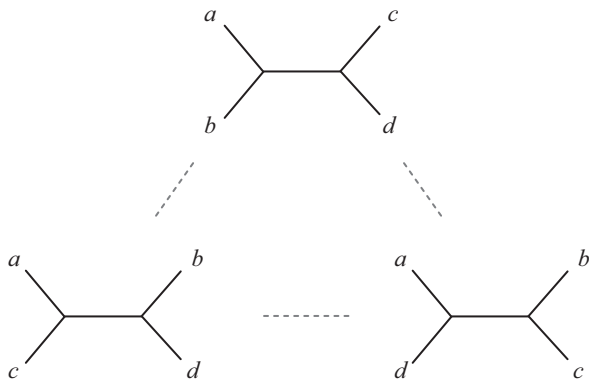

Figure 9.1: The three points of 4-taxon unrooted binary tree space.

completely. It is then more helpful to imagine a space with 195!! points in it, each corresponding to a possible tree, and possible methods of moving from one point to another.

Definition. For any fixed collection X of n taxa, the space of unrooted binary phylogenetic X-trees is a collection of $(2n-5)!!$ points, each of which corresponds to a distinct unrooted topological tree. The space of rooted binary phylogenetic X-trees is a collection of $(2n-3)!!$ points, each of which corresponds to a distinct rooted topological tree.

But this geometric language of 'space' suggests that we should consider some points to be close to one another, and others to be far apart. For instance, of the three trees in Figure 9.2, it seems natural that T_1 and T_2 should be closer to one another than either is to T_3, since even glancing at them shows they have many features in common. If we were searching for the most parsimonious tree, and had found that T_1 has a low parsimony score, we would also want to check the score of T_2, since it has so much in common, and therefore might also have a low score. On the other hand, knowing the score of T_1 doesn't help us get even a rough estimate of the score of T_3, since the trees are so different. To make precise the meaning of the basic intuition that similar trees should have similar scores requires that we pin down the phrase 'similar trees' by giving some notion of how far apart they are in tree space.

There are a number of different ways we can measure the similarity of trees. Each of these is called a *metric* on tree space, and simply gives a way of measuring how far apart we consider two trees to be. Before defining these, however, we first discuss how we might move around in tree space.

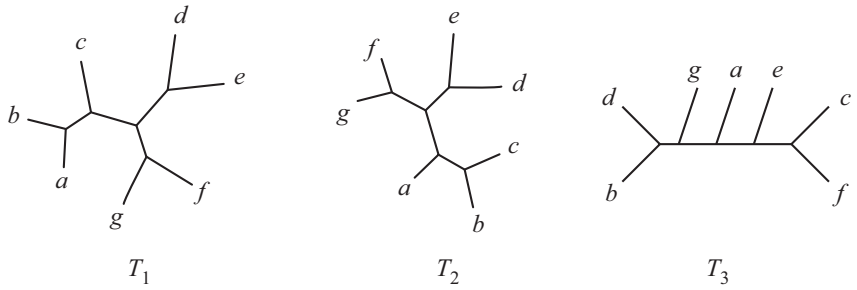

Figure 9.2: Intuitvely, T_1 and T_2 are more similar to each other than to T_3. A metric on tree space will provide a means of quantifying similarity.

9.2 Moves in Tree Space

A move in tree space should take us from one tree to another. In other words, it should begin with a tree, and modify it in a natural way, to produce another tree. While there are many imaginable complex moves, three simple ones are used most often.

Since these moves rearrange a given tree by making changes only to certain parts of it, when we depict these graphically we will group those parts of the tree that are not affected into larger pieces. Thus the triangles appearing in a diagram such as Figure 9.3 represent subtrees whose precise structure does not matter for the move we depict.

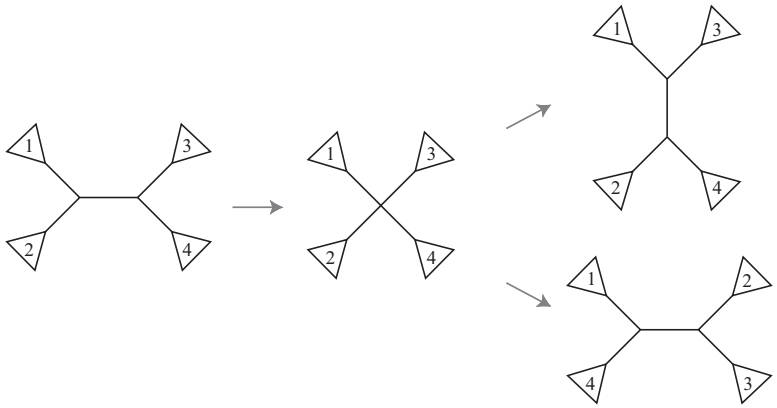

Figure 9.3: NNI moves from one tree to two others.

To create a small change in the tree, the simplest move is called a *nearest neighbor interchange* (NNI). To perform an NNI move, we pick a single internal edge of a tree, and first contract it so that the four edges leading off its endpoints

now arise from a single vertex, as shown in Figure 9.3. The resulting tree is not binary, but we can resolve it into a binary tree by pulling a pair of the edges joining this vertex away from the others, and inserting an edge between the two pairs.

If the pair of edges we pull away in this process are ones that already met in the original tree, then we will just obtain the original tree once again. But any other choice of a pair will give us a different topological tree. Since the 4 edges can be grouped into two pairs in 3 ways, and one of these choices leads back to the original tree, there are exactly 2 new trees we can reach by performing an NNI move on a particular edge.

The simplest examples of NNI moves are already present in Figure 9.1. Starting at any tree, an NNI move on the internal edge of the tree can move us to either of the other two points in the space. The contraction of the internal edge leads to all 4 taxa attached by edges to a single central node. Then picking a pair of taxa and pulling these away from the others can produce any of the three trees.

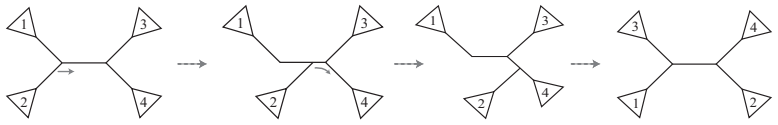

Figure 9.4: An alternate view of an NNI move from one tree to another.

Another way of thinking of an NNI move is shown in Figure 9.4. Focusing on a single internal edge, and the 4 edges that are joined to its ends, we 'grab' one of these 4 edges, and slide its attachment point down the edge toward its other end. There is no change in the tree topology as we do this, until we reach the other end. Then we must make a choice of a new edge onto which we slide the attachment point. (As we do the sliding, we must of course merge the two other edges meeting at the original attachment point into a single edge, and then split the edge on which the new attachment point is located into two edges.) Either of the choices gives a different topology from the original tree, and these are the two trees the NNI move can create. This explanation fits with the terminology 'nearest neighbor interchange', as a subtree has been moved from joining the rest of the tree at a particular location to one of the closest spots that gives a different tree topology.

Given an n-taxon unrooted binary tree, there are $n-3$ internal edges at which we can perform an NNI move. A move on an edge can be performed to reach 2 different trees. It is not hard to see that every NNI move will produce a different tree, so there are a total of $2(n-3)$ trees one NNI move away from any other. This is of course much smaller than the number of trees. It is therefore reasonable to think of an NNI move as taking a small step in tree space, the smallest step possible, in fact.

The next move we consider is called a *subtree prune and regraft* (SPR). It

proceeds by first picking some edge of the tree, and then detaching it from the tree at only one end. The detached edge will still have some taxa joined to it through its other end, forming a subtree that has been 'pruned' from the original tree. We then pick some edge of the other remnant of the tree, and reattach the subtree to it. (Technically, we must divide the chosen edge into two edges, so the subtree can be attached at the new node this introduces.) An illustration is given in Figure 9.5

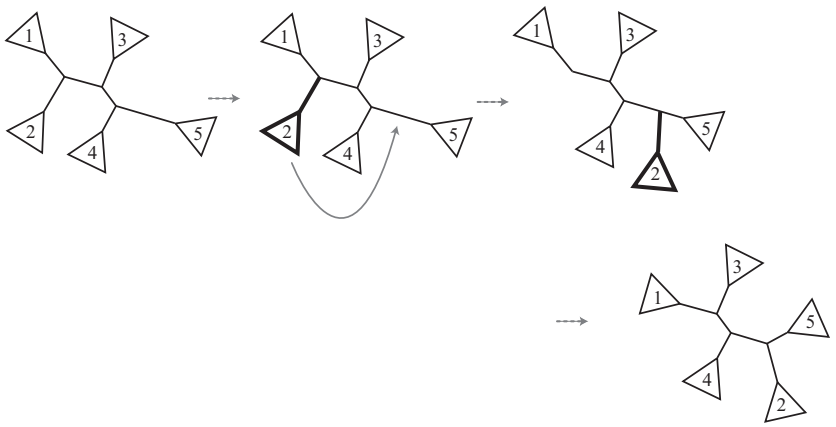

Figure 9.5: An SPR move on a tree.

It should be clear from the second description of an NNI move that any NNI move is also an SPR move; we just choose the regrafting point to be an edge that is very close by. However, for large trees there are many more possible SPR moves than NNIs.

To count how many SPRs could be performed on an n-taxon unrooted binary tree is more involved than counting NNI moves. First, it is helpful to describe the SPR move slightly differently: Instead of detaching an edge at only one end, remove the edge, leaving its endpoints in the two resulting trees components. Then chose one of these endpoints, and one edge in the *other* component of the tree, and introduce a new edge that goes between them. But note that this last choice of an endpoint and an edge in the other component for a regrafting locations is equivalent to just picking the edge for the regrafting location in *either* of the components. After all, once we've chosen the regrafting location, we must use the endpoint in the other component. Thus to specify a particular way of performing an SPR we really need only pick two edges — the one to remove/detach, and then another one where the regrafting occurs. Since the tree has $2n-3$ edges, there are $2n-3$ choices for the first edge, and then $2n-4$ choices for the second, for a total of

$$(2n-3)(2n-4) \tag{9.1}$$

choices of ordered pairs of edges.

However, this overcounts the number of trees we can obtain through an SPR. For instance, if the edge we remove has a vertex in common with the edge where regrafting occurs, the tree topology does not change. Thus we should not count pairs of edges that meet. Since each terminal edge meets 2 other edges, and each internal edge meets 4, there are

$$2n + 4(n-3) = 6(n-2) \tag{9.2}$$

such cases.

A second case of overcounting occurs when the two chosen edges do not meet but have a single edge between them. In this case, the SPR is just an NNI move on the edge between them. But if we either remove the first edge and regraft to the second, or remove the second and regraft to the first, we obtain the same outcome. Thus the order of the two chosen edges does not matter. Since we have already counted the number of different trees we can reach by NNI moves, we know there are $2(n-3)$ of them. However, our earlier count included

$$8(n-3) \tag{9.3}$$

pairs leading to this case: For each of the $n-3$ internal edges of the tree we might have chosen any of 8 ordered pairs of edges that meet it as the pair determining the SPR.

Finally, as long as the two edges determining an SPR move are separated by more than one edge, it is not hard to see that every choices of ordered edges leads to a distinct tree, different than those obtained by NNIs. Counting such choices is most easily done by taking the total count for all ordered pairs of edges in (9.1), and removing the special cases of (9.2) and (9.3). This gives

$$(2n-3)(2n-4) - 6(n-2) - 8(n-3) = 4(n-3)(n-4) \tag{9.4}$$

ordered pairs of this type.

The total number of trees we can reach by a single SPR move is thus this last number, plus the $2(n-3)$ reachable by NNIs. We summarize all these calculations by the following.

Theorem 17. For any n-taxon unrooted binary tree, there are
(i) $2(n-3)$ distinct trees one NNI move away, and
(ii) $2(n-3)(2n-7)$ distinct trees one SPR move away.

The important observation from this theorem is that there are relatively few (approximately $2n$) trees close to a given one by NNIs, while there are many more (approximately $4n^2$) one move away by SPRs. If n is large this difference is substantial. Thus if we move through tree space by NNIs, we will have fewer 'directions' to move in at any point than if we use SPRs. Using SPRs, we can take bigger jumps, changing the tree in many more ways. Neither of these moves is superior to the other; they are just different. If we have already found a pretty good tree, we might want to use NNIs to take small steps around it, exploring tree space thoroughly only in its vicinity. On the other hand, if we

want to wander widely over tree space, looking for any trees that seem good, any number of SPRs will allow us to move around more widely than the same number of NNIs.

A third useful type of move is a *tree bisection and reconnection* (TBR). A TBR move involves first picking some edge of the tree to remove. Next an edge is chosen in each of the resulting tree components. Finally, these two edges are connected by a new edge, as shown in Figure 9.6.

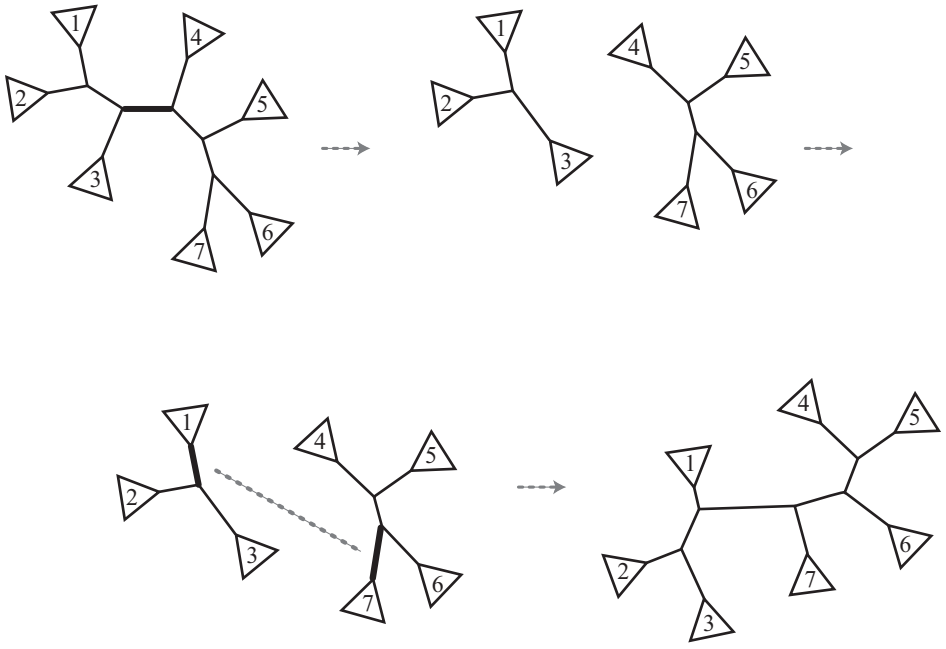

Figure 9.6: A TBR move on a tree.

Notice that while TBR moves are more general than SPRs, SPRs are special cases of TBRs. If a TBR is performed in such a way that we choose one of the edges that met the removed edge for a reconnection point, then the result of the TBR is the same as that of an SPR. Moreover, it is not too hard to see that any TBR can be performed by a succession of two SPRs (see exercises).

Counting the precise number of trees one TBR move away from a given tree is difficult. In fact, the count depends not only on the number of taxa on the tree, but also the tree topology, so there can be no formula of the sort in Theorem 17. However, we can estimate the order of this number fairly easily. To perform a TBR we must first chose an edge to remove, and then choose two other edges to reconnect, for a total choice of 3 edges. Since the number of edges grows linearly in n, choosing 3 edges should lead to something growing at most like n^3. Notice how this grows more quickly than the count for either NNIs or SPRs, as one would expect.

9.3 Searching Tree space

For either parsimony or maximum likelihood methods of tree inference, we must search tree space for the best tree using our chosen criterion. For a given tree, we have already explained how we can efficiently compute either the parsimony score (via the Fitch-Hartigan or Sankoff algorithms), or the maximum of the likelihood function over the numerical parameters of the model (using Felsenstein's pruning algorithm, and techniques of numerical optimization). But then we are faced with the problem that our criterion requires that we compute these scores for *every* tree to choose the best one. However, tree space is big, and in most circumstances an exhaustive search is simply not possible to do in an acceptable amount of time. Instead, we follow a more heuristic procedure.

A rough outline of a search procedure might be as follows:

1. Pick a type of move, say NNI.
2. Choose a tree as a starting point, and evaluate its score.
3. Consider all trees one move away from our current tree, and compute the score for each.
4. If no tree is better than the current one, stop.
5. Otherwise, choose the best tree of these, and move to it. Then go back to step 3.

The strategy here can be visualized in a very natural way. If we imagine each point in tree space as a point in a plane, then we can represent the tree's score as a height above this plane, so that all the scores form something like a surface over tree space., as cartoonishly depicted in Figure 9.7. Our goal is to maximize the score (at least for ML), or equivalently to find the highest point on the surface. We begin at some point on this surface (our initial tree) and look around at all the places we could reach by taking a single step. We then choose to go to the one that takes us the highest. In other words, we try to climb up by always going in the steepest direction. Sooner or later, we will reach the top of a hill, and will not be able to climb higher.

The problem with this strategy, of course, is that there may be several hills, of different heights. We may end up climbing to the top of a short hill, reaching a local maximum. Because our steps are of limited size, we might never be able to reach a higher hill without first descending. Thus there is no guarantee we will ever get to the global maximum. Our strategy is essentially a local one, only considering nearby points in tree space, and doesn't take a long-distance view.

We could try to consider more distant trees as well, perhaps by using an SPR instead of an NNI as our basic move. It will then take us longer to decide in which direction to move, since there will be more options. Moreover, as long as there are some trees we don't consider, we may still miss seeing the highest hill, and still end up climbing something shorter. So while using bigger steps

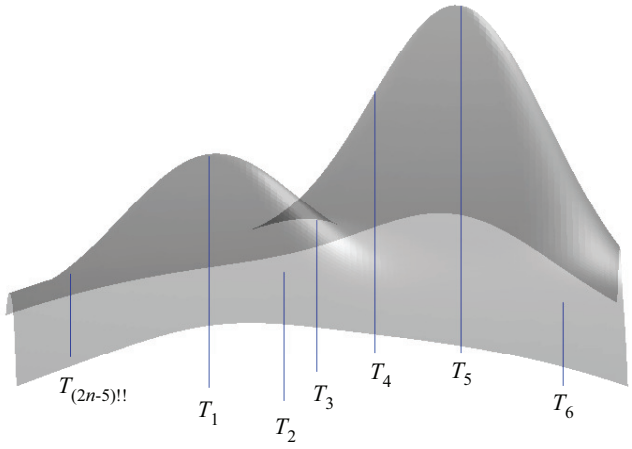

Figure 9.7: Hill climbing on tree space.

makes it less likely that we climb the wrong hill and become trapped at its top, it doesn't reduce the risk to zero.

The simplest solutions to this problem is to repeat the search many times, using different starting points. If the starting points are varied enough, we are more likely to eventually choose one near the highest hill, and climb to its top. There will still be no guarantee that we ever find the true optimum, but we make it more likely that we do so.

There are many variations on this basic search procedure that might result in reaching at least a local optimum faster. For instance, instead of considering all trees one move away from the current one before picking a better one, we could consider them one at a time, and move to a new one whenever we find it. It is hard to guess whether this would produce a faster search, since we may end up moving more often to get small improvements in the tree, when we might have been better off waiting to find the biggest improvement we can. Such a decision must be made in developing software, and may be based on observing how the software performs on real data, rather than on theoretical considerations.

But there are other decisions that may be left to the software user. For instance, should we use only NNI moves, or SPRs, or TBRs in a search? NNIs are small steps in tree space, so using them is likely to produce a very gradual improvement in the tree. The more radical moves of SPRs and TBRs are larger steps in tree space, and have the potential to produce greater improvements in a single step. However, there are many more of them to consider, so it may take longer to find a better tree. We might also end up taking too large a step, and miss the best tree before coming back to it later. One appealing approach

is to begin using SPRs or TBRs to crudely search with big steps, taking a step every time we come across a better tree, and then switch to NNIs for a finer local search around the best trees previously found.

Most importantly, though, one should always be sure the search is performed repeatedly, from multiple starting points (or with some random component to the search). This is simply because there may be several locally optimal trees, and a single search may end at one of these even though it is not globally optimal. It is tempting of course to begin a search with a tree you believe may be close to optimal. For instance, since it is quick to perform Neighbor Joining, an NJ tree might be a natural starting point for a ML search. Certainly such a tree could be one of your starting points, since it is quite plausible that you will soon find the optimal tree from it. However, if that is your only starting tree, you may miss the best. The issue here is not simply one of a mathematical possibility that doesn't arise in practice — there are real data sets in which the likelihood has several local maxima on tree space. A single search can become trapped in a local maxima and never find the optimal tree.

There are other techniques that can sometimes be useful to prevent ourselves spending too much time searching areas of tree space that are unlikely to yield good trees. These are generally adaptations of standard ideas from computer science, such as *branch and bound*, which we will not discuss.

9.4 Metrics on Tree Space

It is desirable to have some natural way of quantifying how far apart we consider two trees to be. That is, we would like to have a metric on tree space. Ideally this metric would capture whatever informal idea we have of two trees being similar; a small value would mean two trees 'look mostly alike' and a large value that they are 'very different.'

Each of the moves we've discussed on tree space leads to a way of measuring the differences between two trees. First, it is not hard to see that any tree on a set of taxa can be reached from any other tree by some succession of these moves. Then, if it is possible to move from one tree to another using k moves, but not using fewer moves, we might say these trees are k apart.

Definition. For a fixed choice, $\mu \in \{\text{NNI, SPR, TBR}\}$, of move on tree space the *μ-edit metric* measures the distance between trees T and T', denoted $d_\mu(T, T')$, as the minimal value of $k \geq 0$ such that there exists a sequence of trees

$$T = T_0, T_1, T_2, \ldots, T_k = T'$$

with T_{i+1} one μ-move from T_i for each i.

As should be clear, these give very different notions of distance on tree space. For instance, one of the exercises demonstrates that for any $k \geq 1$ there are trees T and T' such that $d_{\text{SPR}}(T, T') = 1$, yet $d_{\text{NNI}}(T, T') = k$. On the other hand, if

$d_{\mathrm{NNI}}(T, T') = 1$, then $d_{\mathrm{SPR}}(T, T') = d_{\mathrm{TBR}}(T, T') = 1$ as well. More generally,

$$d_{\mathrm{NNI}}(T, T') \geq d_{\mathrm{SPR}}(T, T') \geq d_{\mathrm{TBR}}(T, T'),$$

since any NNI move is an SPR, and any SPR move is a TBR.

It is not clear how to compute any of these distance between trees in an efficient manner, since the definition suggests we consider all sequences of trees between the given two. Since in a sequence of minimal length there can be no repeated trees, there are a finite, though huge, number of possibilities to consider. It is known that computing the NNI- and TBR-edit distances are NP-hard problems. There are also interesting results on the the maximal distance between trees under these metrics, called the *diameter* of tree space. Understanding these diameters is important both for understanding how many steps might be needed to get from one tree to another, and also for interpreting the meaning of the distance between two trees in terms of similarity. For instance, if tree space has a small diameter under a given metric, the metric could only crudely measure similarity of trees.

A metric on tree space that is easier to compute is the *splits metric*, also called the partition metric, symmetric distance, or the Robinson-Foulds metric. Recall from Chapter 4 that associated to any edge of a tree T is a split of the taxa on its leaves; by removing the edge, the taxa are partitioned into two sets, according to the two resulting tree components,

Definition. The *splits metric* on X-trees measures the distance, $d_{\mathrm{splits}}(T, T')$, between two trees as the number of splits that occur on one tree or the other, but not both.

This is straightforward to compute, since we can simply list all the splits. Of course there is no need to include trivial splits corresponding to terminal edges, since these will appear on any tree. For instance, Figure 9.8 shows two trees with 4 different splits occurring among them, so these are distance 4 apart.

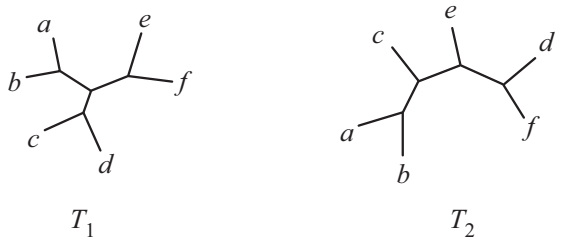

Figure 9.8: The tree T_1 has non-trivial splits $\{a, b\}|\{c, d, e, f\}$, $\{c, d\}|\{a, b, e, f\}$ and $\{e, f\}|\{a, b, c, d\}$. The tree T_2 has non-trivial splits $\{a, b\}|\{c, d, e, f\}$, $\{a, b, c\}|\{d, e, f\}$ and $\{a, b, c, e\}|\{d, f\}$. After deleting the split these trees have in common, four remain, so $d_{\mathrm{splits}}(T_1, T_2) = 4$.

While easier to compute, a drawback of the splits metric is that it can be quite large for two trees that we might consider to be similar in most of their features. Moving only a single taxon via an SPR move on a terminal edge can produce a tree that is very far from the original as measured by the splits metric. The misplacement of a single 'rogue taxon' is thus judged to have a huge effect on a tree. (See Exercise 9.)

Another metric on tree space that can be computed efficiently uses quartets instead of splits

Definition. The *quartet metric* on X-trees measures the distance, $d_{\text{quartet}}(T, T')$, between two trees as the number of 4-element subsets of X that induce different quartet trees on T and T'.

Since a collection of n taxa has $\binom{n}{4} = \frac{n(n-1)(n-2)(n-3)}{24}$ subsets of 4 taxa, one could simply list these and check whether the induced quartet trees are the same, to create a $\mathcal{O}(n^4)$ algorithm for computing this metric. However, there are more efficient ways to compute the quartet metric than this obvious approach, with one method only requiring $\mathcal{O}(n \log n)$ time.

The quartet distance is not affected as much as the split distance by a single taxon being moved from one part of the tree to another. This should not be surprising, since if a single taxon is moved, only those quartets involving that taxon are affected, so many quartets are left unchanged (See exercise 10.)

9.5 Metrics on Metric Tree Space

For searching through tree space in algorithms, moves from one topological tree to another are thought of as separate from modifications to edge lengths in a metric tree. But for measuring how close two metric trees are to one another it's desirable to have metrics on tree space that consider both topology and edge lengths.

One natural way to do this builds on the splits metric. Given n-taxa, there are $2^{n-1} - 1$ possible splits, and thus any topological tree can be represented by a vector of 0s and 1s, of length $d = 2^{n-1} - 1$; we simply choose some ordering of the possible splits and then record a 1 to indicate a split is displayed on the tree, or a 0 to indicate it is not. The splits metric between topological trees can then be found by subtracting their corresponding vectors (producing a vector with 1s and -1s indicating differing splits), taking the absolute value of each entry, and then adding them up.

In more mathematical language, the splits metric just involves mapping each tree to vector of 0s and 1s in $\mathbb{R}^d$, and then using the L^1 norm to compute distance:

$$||\mathbf{v} - \mathbf{w}|| = \sum_{i=1}^{d} |v_i - w_i|.$$

This immediately suggests the following metric splits distance: map each tree into $\mathbb{R}^d$ by recording the length of the edge of a tree associated to a split

in the coordinate corresponding to that split, with a 0 in all other coordinates. Then use the L^1 norm on these vectors to compute distances between metric trees. A formula for distance between two X-trees is thus

$$d_{ms}(T_1, T_2) = \sum_{\text{splits } e} |w_1(e) - w_2(e)|.$$

Here we use e to denote a split of X, with $w_i(e)$ being the length of the corresponding edge in T_i if the split is displayed on T_i, and $w_i(e) = 0$ if the split is not displayed on e_i.

One can replace the use of the L^1 norm here with any other norm on $\mathbb{R}^2$. A very natural choice is to use the Euclidean one, giving

$$\tilde{d}_{ms}(T_1, T_2) = \left(\sum_{\text{splits } e} (w_1(e) - w_2(e)) \right)^{1/2}.$$

Notice that in the case of topological trees, whether one uses the Euclidean or L^1 norm, the same tree metric is computed, but for metric trees these give different results.

9.6 Additional Remarks

Although we've motivated this look at tree space by the need to search for optimal trees, the concepts of this chapter are useful for other issues.

SPR moves appear naturally in modeling *recombination* of genes. Whether this recombination occurs through sexual reproduction, or by lateral gene transfer across species, the result is essentially that a subtree has been pruned from one location on a tree, and regrafted elsewhere. Of course there are constraints on where the regrafting can occur, since recombination can occur only between organisms alive at the same time. Thus the most proper framework requires we work with rooted metric trees for this application

Having chosen a metric on tree space also makes it possible to define a *median tree* for a collection of trees. This is another form of a consensus tree, which might be used to as a summary statistic for a collection of trees. Just as the median of a set of numbers is the one 'in the middle,' a median tree T_m is one that minimizes the total distance to all trees in the collection. That is, for a collection $\{T_1, T_2, \ldots, T_k\}$,

$$T_m = \arg\min_T \left(\sum_{i=1}^{k} d(T, T_i) \right).$$

9.7 Exercises

1. The text's explanation of the NNI move claimed that there were exactly 3 ways 4 edges could be grouped into 2 pairs. Explain this.

2. The text describes an alternate way of thinking of an NNI move, as sliding one edge down another past a junction of edges. It seems that since we could pick any of 4 edges to 'slide', and for each have a choice of 2 places to slide it to, there should be 8 possible new trees we could get doing this. Explain why there are only 2. (You might just consider each of the 8 possibilities, and group them by their outcomes.)

3. For the 5-taxon tree $((A, B), C, (D, E))$, draw all 4 trees that can be obtained from it by a single NNI move.

4. For 5-taxa, give an example of two trees that are 2 NNI moves apart, but only 1 SPR apart.

5. Generalize the last problem to produce an example of two trees with a large number of taxa that are n NNI moves apart, but only 1 SPR apart.

6. Check all the the arithmetic leading to part (ii) of Theorem 17.

7. How many 5-taxon unrooted binary trees are there? For any of these trees, how many trees are 1 SPR away? Draw all the trees that are 2 or more SPR moves from $((A, B), C, (D, E))$.

8. Explain how any TBR move can be performed by a succession of two SPR moves.

9. Consider the two caterpillar trees with n-taxa,

$$T_1 = (\ldots((A_1, A_2), A_3), A_4)\ldots, A_n),$$
$$T_2 = (\ldots((A_1, A_n), A_2), A_3), \ldots, A_{n-1}).$$

   a) Explain how T_2 can be obtained from T_1 by a single SPR move.

   b) What is the splits metric distance between the two trees?

   c) For comparison, what is the maximal distance between two n-taxon trees under the splits metric? What fraction of this is your answer to part (b)?

10. Consider the trees of Exercise 9

    a) What is the quartet metric distance between the two trees?

    b) For comparison, what is the maximal distance between two n-taxon trees under the quartet metric? What fraction of this is your answer to part (a)?

11. Suppose a single SPR move is performed on a large tree.

    a) Describe which splits of the original tree remain in the result. (Hint: Think about which edges are associated to the splits that are lost. The locations of these can be described using geometric language.)

    b) Show that if $d_{\mathrm{SPR}}(T, T') = 1$ then $2 \leq d_{\mathrm{splits}}(T, T') \leq 2(\operatorname{diam} T - 2)$, where $\operatorname{diam} T$, the *diameter* of T, is the length of the longest path in T.

12. Prove the μ-edit metric satisfies the requirements to be a metric on tree space: 1) $d_\mu(T_1, T_2) \geq 0$, and $d_\mu(T_1, T_2) = 0$ only when $T_1 = T_2$, 2) $d_\mu(T_1, T_2) = d_\mu(T_2, T_1)$, and 3) $d_\mu(T_1, T_3) \leq d_\mu(T_1, T_2) + d_\mu(T_2, T_3)$.

13. The presentations of the moves in this chapter can be formalized to use the more technical language of graph theory. Each modifies the set of vertices and edges defining a tree in specific ways. Describe these precisely for:

    a. NNI

    b. SPR

    c. TBR

14. Explain why the L^1 norm and the square of the Euclidean norm give rise to the same metric on *topological* tree space in the construction given in section 9.5.

Chapter 10

Rate-variation and Mixture Models

All the Markov models of DNA mutation introduced thus far assume that every site in the sequences behaves identically; of course this assumption is far from justifiable biologically.

For instance, in coding DNA, due to the redundancy of the genetic code, the third base of many codons can undergo a substitution without any effect on the protein product. Also, if both coding and non-coding sites are included in a sequence, it's reasonable that the non-coding sites might be able to change more freely, at least if they have no regulatory purpose. Even in coding regions, it may well be that the functional constraints on different parts of a protein molecule vary, so that parts of a gene may evolve at a different rate from other parts. These functional constraints might even change over evolutionary time, so that the behavior of a site may vary from one part of the tree to another.

In general, then, it is desirable to expand the substitution models developed so far, in order to incorporate variation among the sites. It is also important that this be done so that we need not decide in advance which sites behave similarly. Seldom will we have the detailed knowledge to choose an *a priori* classification of sites into different classes. Instead, we use *mixture models* which posit different classes of behavior, but leave the proportion of sites assigned to the different classes as parameters. The class sizes, as well as the parameters determining the behavior of each class, will then be inferred from the data.

10.1 Invariable Sites Models

The simplest form of a rate-variation models is built on two classes of sites. The first class of *variable* sites mutates according to a model, such as the GTR, of the sort discussed earlier, and the second class of *invariable* sites undergoes no mutation at all. When we examine DNA data arising from such a model, if we observe a substitution at a site, then that site is certainly in the first class.

However, if we do not observe any change we generally cannot tell which class the site came from. It may have been free to mutate but did not due to the randomness in the model, or it may have been invariable. Thus the sites that will be observed as unvarying are a larger set than the invariable ones. As a result, we will not be able to easily tell which sites were invariable in order to remove them from data before we analyze it. Instead, we must formalize the model more carefully.

For concreteness, consider a model with invariable sites in which variable sites mutate according to the general Markov model, usually called the GM+I model. We'll formulate this for characters with κ states, for trees relating n taxa.

For any fixed rooted tree T relating n taxa, we have the usual parameters for the variable sites of a root distribution vector $\mathbf{p}_\rho$ and Markov matrices for each edge $\{M_e\}_{e\in E(T)}$. In addition we need a class size parameter s, indicating that with probability s a site is variable, and with probability $1-s$ it is invariable. Finally, for the invariable sites we need another distribution vector $\mathbf{p}_{inv}$ indicating the state distribution for the invariable sites. Thus a binary n-taxon tree will require

$$1 + 2(\kappa - 1) + (2n - 3)\kappa(\kappa - 1) \tag{10.1}$$

parameters, which is only κ more than the GM model.

Note that we could have further assumed the variable and invariable sites had the same state distribution, so $\mathbf{p}_{inv} = \mathbf{p}_\rho$. With that approach, the only new parameter introduced over the GM model would have been r, the class size parameter.

In order to see the relationship between the GM+I model parameters and the joint distribution describing observations of bases at the leaves of a tree, we begin by analyzing separately the two classes of sites in the model.

For the GM model, Chapters 6 and 8 described how to produce the entries of the joint distribution as polynomials in the parameters that are entries of $\mathbf{p}_\rho$, $\{M_e\}_{e\in E(T)}$. For a tree relating n-taxa, this joint distribution array will be n-dimensional, of size $\kappa \times \kappa \times \cdots \times \kappa$. So suppose we've found this array P_1, where $P_1(i_1, i_2, \ldots, i_n)$ gives the probability that a variable site shows the pattern $(i_1, i_2, \ldots, i_n)$ of states at the leaves.

For the invariable sites we could do a similar calculation, using the root distribution vector $\mathbf{p}_{inv}$ and the identity matrix I for all Markov matrices on edges, since this matrix describes no possibility of substitutions occurring. However, it should be clear that such a computation would produce an n-dimensional $\kappa \times \kappa \times \cdots \times \kappa$ array P_2 such that

$$P_2(i_1, i_2, \ldots, i_n) = \begin{cases} q_{i_1} & \text{if } i_1 = i_2 = \cdots = i_n, \\ 0 & \text{otherwise.} \end{cases}$$

That is, the joint distribution for this class gives zero probability for any patterns in which states vary, and probability equal to the state distribution for those that show no variation.

The joint distribution for the full GM+I model is then a weighted sum of the distributions for the two classes,

$$P = sP_1 + (1 - s)P_2. \tag{10.2}$$

With $0 \leq s \leq 1$, the weights s and $1 - s$ here are simply the relative sizes of the classes, or equivalently the probabilities that a randomly chosen site is in a given class. Statistical models such as this, in which several classes are modeled separately, but then combined so that we allow the class to be determined randomly, are called *mixture models*.

There are straightforward variations on this idea using other models in place of GM. For instance, a general time reversible model with invariable sites (GTR+I), simply replaces the Markov matrices of the GM model above by the appropriate exponentials of a rate matrix. The JC+I and K2P+I are defined similarly.

If we believe a mixture model such as GTR+I describes our data, how should this affect our methods of inference?

Note that the results of a parsimony analysis are unaffected by invariable sites as they give uninformative patterns. Thus parsimony is a reasonable method under the same circumstances as it is for variable sites alone. (However, most users of parsimony never mention any model, since the method makes no explicit use of one.)

In contrast, our derivations of distance formulas for the various models all tacitly assumed no invariable sites are present. If only a small fraction are likely to be invariable, we may view the distances as approximately valid. Nonetheless, the use of the model-based distance formulas essentially require that the number of invariable sites be negligible. If this is not the case, there can be a systematic bias in the values of inferred distances between taxa. For the Jukes-Cantor model it is not hard to work out exactly how this effect occurs (see exercises).

In fact, it is relatively easy to prove that even under the model JC+I the proportion of invariable sights cannot be estimated if we only allow the comparison of 2 sequences at a time. That means there can be no way to compute distances for the model JC+I, unless we already know the proportion of invariable sites by some other means. (There are ways to estimate this proportion by comparing larger numbers of sequences at a time, which have been implemented in some distance-based software.)

Fortunately, the statistical framework of maximum likelihood handles a model such as GTR+I with little additional complication. A few additional numerical parameters must be varied as we search for the maximum of the likelihood function on a given tree, but no more substantive changes are needed. It is in this setting that such models are typically used, since a more accurate model of the true substitution process is hoped to lead to more accurate inference.

10.2 Rates-Across-Sites Models

It is now easy to imagine how one might have a finite number of classes of sites, each modeled by a different choice of GM parameters on a tree, in a more complex mixture than a GM+I model. For each class we produce a joint distribution array, and then take a weighted sum of these, much as in equation (10.2). The GM+I model is just a particularly simple form, where the parameters for one of the two classes allow no mutation. In current practice, though, models with many fewer parameters than a mixture of GMs are typically used.

The most common multiclass mixture is built on a GTR model, in which all sites use the same rate matrix, but scaling factors are introduced to slow down or speed up the process at individual sites.

For a fixed tree T, the parameters will be as follows. We assume a common GTR rate matrix Q for all edges of the tree, and a root distribution vector that is a stable base distribution for Q. We have scalar edge lengths $\{t_e\}_{e\in E(T)}$ for all edges of the tree. If we choose to use m classes of sites, we create m *rate scaling parameters* $\lambda_1, \lambda_2, \ldots, \lambda_m$, which will serve as factors to speed up or slow down the substitution process for the different classes. We also need a vector $\mathbf{r} = (r_1, r_2, \ldots, r_m)$, with entries adding to 1, giving the relative sizes of the rate classes.

Now sites in the ith rate class will evolve using the rate matrix $\lambda_i Q$ throughout the tree. Thus the larger the value of λ_i, the faster substitutions will occur. For that class, then, on an edge e of the tree we have the Markov matrix $M_{e,i} = e^{t_e \lambda_i Q}$. It is now straightforward to compute P_i, the joint distribution array at the leaves for the ith class, using approaches discussed in earlier chapters.

The last step is to combine the distributions from the various classes to get the joint distribution for the mixture model,

$$P = \sum_{i=1}^{m} r_i P_i. \tag{10.3}$$

To be a bit more sophisticated, one can also imagine a continuous distribution of rates, given by a density function $r(\lambda)$, in which case we have

$$P = \int_{\lambda} r(\lambda) P_\lambda \, d\lambda.$$

Since P_λ is a multidimensional array, the integral here is meant to be performed entry-wise, so it represents a separate integral for each entry, just as the summation on equation (10.3) represents a separate sum for each entry.

In current practice, it is common to use a Γ distribution of rates (with mean 1), whose density function is given by the formula

$$r(\lambda) = \frac{\alpha^\alpha}{\Gamma(\alpha)} \lambda^{\alpha-1} e^{-\alpha\lambda}.$$

This is a flexible distribution, whose precise shape varies according to the value of α. It is useful for describing quantities that may vary over the positive numbers, which is the appropriate range for the rate scaling parameters λ. The *shape parameter* α of the GTR+Γ model then adds only one additional parameter to those of the GTR model, yet allows for a better fit to data.

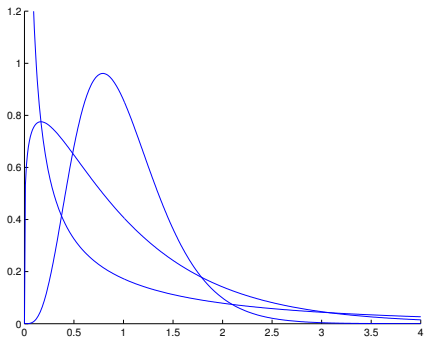

Figure 10.1: The Γ distribution for $\alpha = .3, 1.2, 4.8$, in order from top to bottom at left edge of the graph.

Figure 10.1 shows graphs of the density function $r_\alpha(\lambda)$ for three values of α. When $\alpha < 1$, the distribution has a spike at low values, and a long tail at high ones. This means some sites are nearly invariable, and there is a very wide spread of rates for the others, with some sites evolving quite quickly. For $\alpha > 1$, the distribution has a hump around 1, which becomes taller and narrower as α is increased. Thus for large α, such a distribution will produce a model little different from a standard GTR. Thus by varying α from very large to very small the model may produce behavior ranging from very similar to a straight GTR model, to one in which sites evolve at a very broad spectrum of rates.

Finally, it is easy and often desirable to add an additional class of invariable sites, producing the model designated GTR+Γ+I. When the GTR+Γ and GTR+Γ+I models are fit to the same data set, the first of these models often requires a much smaller value of α than the second. This should seem reasonable, as a low value of α enables the model to capture some invariable, or nearly invariable sites. Explicitly including an invariable class allows the α to be chosen to better fit the more variable sites.

Although the GTR+Γ and GTR+Γ+I are the models most commonly used for routine data analysis, it is important to realize that there is no clear biological justification for the use of a Γ-distribution. There is no mechanistic model of how substitutions occur that leads to the particular form of Γ, or of any other distribution. It is simply an arbitrary choice that provides flexibility with the addition of only one more parameter.

In fact, it is not really the Γ-distribution, but rather a discretization of it with a finite number of rate classes, that is used in most software implementations of maximum likelihood. Typically only a handful of rate classes, say four or five, are used. If, for instance, there are four classes, then the cutoffs for the lowest 25%, 50%, and 75% of the distribution are determined. Then the means of λ in each of the resulting four intervals are computed. These four values are then used as the rate scaling factors λ_i for four classes each of proportion 0.25. Experience with data analysis has shown that using more classes seldom leads to much of an improvement in likelihood scores, and by keeping the number of classes smaller, computations can be done a bit more quickly. Nonetheless a model using this approximation might be more appropriately designated GTR+dΓ(4) to emphasize that it in fact uses a 4-class discretized Γ.

10.3 The Covarion Model

There is another way of introducing a form of rate variation into models, though its mathematical development is more recent than the rates-across-sites approach, and it has not been implemented in all commonly used software packages. The motivation, which arose in a paper of Fitch and Markowitz in 1970, is biologically quite an attractive one, even though the mathematical formulation does not capture the full process they described.

Note that in a rates-across-sites model, each site is presumed to have a fixed rate parameter λ which remains constant throughout the tree. Whatever leads to some sites mutating at different rates than others are thus imagined to be unchanging across all lineages throughout the tree.

Particularly if we consider evolution over long time periods, however, we might expect this to be unreasonable. Perhaps early on some sites are unable to change because they code for a part of a protein that is essential for the organism to live. After other parts of the sequence have evolved, however, the part of the protein those sites code for may no longer be so essential, and so they become free to vary in a different part of the tree. Fitch and Markowitz called the codons that were free to vary at a particular time 'covarions' as shorthand for 'concomitantly variable codons.'

In more recent usage for describing probabilistic substitution models the term 'covarion' has come to refer to models in which characters may undergo some switching between being free and not free to vary as evolution proceeds down the tree. More generally, the terminology is used when characters may undergo a switching of a certain form between any sorts of different classes, such as different rate classes.

Note that the motivation of Fitch and Markowitz for a covarion model is essentially an argument *against* an assumption of independence of the mutation process at different sites. The reason they give why some sites change from being invariable to variable is because of changes in other parts of the gene. However, it is quite unclear how to formulate such a model in a mathematically tractable way. Thus the mathematical formulation of the covarion model will

still assume sites behave independently, but will include a switching mechanism so that sites may change their behavior.

The simplest form of the covarion model, introduced by Tuffley and Steel in 1998 uses only 2 classes, one of which is invariable. For a DNA model, instead of the usual 4 states A, G, C, T for our characters, we will have 8 states, which we designate by

$$A^{\text{on}}, G^{\text{on}}, C^{\text{on}}, T^{\text{on}}, A^{\text{off}}, G^{\text{off}}, C^{\text{on}}, T^{\text{off}}.$$

The superscript 'on' means the site is currently free to vary, while 'off' designates it is currently held invariable.

For the 'on' states we assume a instantaneous rate matrix Q of a GTR model, and let $\mathbf{p}$ be its stable base distribution, so $\mathbf{p}Q = \mathbf{0}$. We need two additional parameters, an instantaneous rate s_1 at which 'on' states switch to 'off' states, and an instantaneous rate s_2 at which 'off' states switch to 'on.' We construct an 8×8 rate matrix

$$\tilde{Q} = \begin{pmatrix} Q - s_1 I & s_1 I \\ s_2 I & -s_2 I \end{pmatrix},$$

where I denotes a 4×4 identity matrix.

With the ordering of bases as listed above, we interpret the entries of $\tilde{Q}$ as follows. The upper left block $Q - s_1 I$ describes changes from 'on' bases to other 'on' bases. It has the same off-diagonal entries as Q, so these represent the usual rates of substitutions from one base to another, as in a GTR model. The upper right block, $s_1 I$, describes changes from 'on' bases to 'off' bases. For instance, A^{on} switches to A^{off} with rate s_1. Since this block is diagonal, when an 'on' to 'off' instantaneous switch occurs, the nucleotide may not change simultaneously. Note the diagonal entries of the upper left block were adjusted from those of Q by subtracting off $s_1 I$ simply because a rate matrix must have rows adding to 0.

The lower left block of $\tilde{Q}$ describes 'off' bases switching to 'on'. Note that these switches occur with rate s_2. Again, when a switch occurs, the base may not change at that instant. The lower right block should describe base changes from 'off' bases to 'off' bases. Since 'off' means no such change can occur, the off-diagonal entries of this block are all zero. The diagonal entries are then chosen so that the rows of $\tilde{Q}$ add to 0.

We next must choose an 8-element root distribution for the model. Letting

$$\sigma_1 = \frac{s_2}{s_1 + s_2}, \quad \sigma_2 = \frac{s_1}{s_1 + s_2},$$

and

$$\tilde{\mathbf{p}} = (\sigma_1 \mathbf{p}, \sigma_2 \mathbf{p}),$$

one can check (Exercise 7) that

$$\tilde{\mathbf{p}}\tilde{Q} = \tilde{\mathbf{0}}, \tag{10.4}$$

and thus $\tilde{\mathbf{p}}$ is a stable distribution for $\tilde{Q}$. In fact, the rate matrix $\tilde{Q}$ and root distribution vector $\tilde{\mathbf{p}}$ form a time reversible model (Exercise 8).

Now for any tree T, with a root ρ chosen arbitrarily, we have an 8-state time-reversible model with root distribution vector $\tilde{\mathbf{p}}$, rate matrix $\tilde{Q}$, and edge lengths $\{t_e\}_{e\in E(T)}$, where the Markov matrix $M_e = \exp(t_e\tilde{Q})$ is assigned to the edge e.

There is one remaining feature of the covarion model, however, to be formulated. When we observe sequences, we are not able to distinguish whether a site is currently 'on' or 'off'. For instance, both states A^{on} and A^{off} are observed simply as A. (For those familiar with hidden Markov models, the covarion model is of that sort, with some of the state information unable to be observed.)

To incorporate this into the covarion model we make use of the 8×4 matrix

$$H = \begin{pmatrix} I \\ I \end{pmatrix},$$

constructed from two stacked identity matrices, which has the effect of hiding the 'on/off' feature of a base. More specifically, since

$$\begin{pmatrix} p_1 & p_2 & p_3 & p_4 & p_5 & p_6 & p_7 & p_8 \end{pmatrix} H = \begin{pmatrix} p_1+p_5 & p_2+p_6 & p_3+p_7 & p_4+p_8 \end{pmatrix},$$

H acts on any vector giving a distribution of the 8 states in our chosen order to give the 4 combined states corresponding to the bases in DNA.

Now for internal edges of the tree we make the Markov matrices be the M_e as described above, while for edges leading to leaves use M_eH. (Here we assume we have located the root of the tree at an internal vertex.) While M_eH is not square, it still has non-negative entries, with rows summing to 1, and so has a perfectly reasonable stochastic interpretation. With parameters on all edges of the tree, and a root distribution, we can now calculate the joint distribution at the leaves in the usual ways discussed in Chapter 8.

The basic example of a covarion model extends naturally to allow for more classes. For instance, one might formulate a covarion model with 3 rate classes, 'fast,' 'medium,' and 'slow.' If Q is a GTR rate matrix for the fast class, then for some rate scaling parameters $\lambda_1 = 1 > \lambda_2 > \lambda_3 \geq 0$, and some switching parameters $s_{ij} \geq 0$, we construct a 12×12 rate matrix

$$\tilde{Q} = \begin{pmatrix} \lambda_1 Q - (s_{12}+s_{13})I & s_{12}I & s_{13}I \\ s_{21}I & \lambda_2 Q - (s_{21}+s_{23})I & s_{23}I \\ s_{31}I & s_{32}I & \lambda_3 Q - (s_{31}+s_{32})I \end{pmatrix}.$$

While some constraints (which we omit here) must be placed on the s_{ij} so that this leads to a time-reversible model, a stable base distribution can be given. Finally, the matrix H which performs the hiding of the rate class will consist of three stacked identity matrices.

Note that because these covarion models have a stable state distribution, the fraction of sites within a given rate class is constant over the tree. Thus while

an individual site may change its class, the model assumes that other sites are also changing classes in a way that keeps the class sizes balanced.

Another feature to note is that if all of the switching parameters in a covarion model are $s_{ij} = 0$, then the model reduces to a more standard mixture model in which sites may not change classes. For instance, in the 3-class example above, setting switching parameters to 0 gives $\tilde{Q}$ a simpler block-diagonal form which means only in-class base substitutions are possible.

10.4 General Mixture Models

Once the idea of using a mixture model has arisen to capture rate variation across sites (which includes models with "+I", "+Γ", and "+I+Γ", as well as the covarion models), there is no reason to not consider more complicated mixtures. One could imagine several classes of sites, with each class having essentially unrelated parameters describing its base substitution process.

A simple example would have two classes, each of which evolves according to the GTR model on the same topological tree, but with possibly unrelated rate matrices and branch lengths. Parameters for such a model would be the topological tree, two complete sets of numerical parameters for the GTR model including rate matrices Q_1 and Q_2 and edge lengths, and one additional mixing parameter r that determines the class proportions r, $1-r$. The joint distribution for such a model is simply the weighted sum of the distributions for the two classes, just as in equation (10.2). In fact, the GTR+I model is a submodel of this one, in which we have decided ahead of time that the rate matrix Q_2 is the zero matrix, so that characters in it never change. A two-class rates-across-sites model is also a special case of this one, in which we require that the two rate matrices be multiples of one another, $Q_2 = \lambda Q_1$.

One can easily extend this type of model to have a larger number of classes. Since each additional class increases the number of parameters, though, the more classes that are used, the longer sequences will have to be to obtain good estimates of parameter values. Though models with a large number of classes are unlikely to ever be used for routine data analysis, there has been a fair amount of recent interest in exploring their behavior on real data sets. If there is reason to believe a less standard model is inappropriate, they offer the next step in modeling complexity.

While the mixtures described so far have all had only one topological tree among their parameters, one can also consider models where each class has its own tree parameter. This might be useful, for instance, if either hybridization or some form of lateral gene transfer had occurred so that different parts of the sequences had evolved on different trees. Though these models have been studied theoretically, they have not yet been explored for practical data analysis.

When a complex model is formulated, it is possible that the model has been made so complex that it is mathematically impossible to use it to validly infer parameters. More precisely, the parameter values may not be able to be uniquely *identifiable* from the theoretical distribution of the model, much less from an

approximation of it from data. For instance, it might be that two different tree topologies give rise to exactly the same distribution, so that one could not decide which of them led to a given data set. While several papers in the last few years gave alarming indications that mixture models can be problematic in this way, the most recent theoretical work has shown it should not be a real worry to a practitioner. In fact, the number of classes that can be safely used for theoretical good behavior grows with the number of taxa, and for even a moderate number of taxa is much larger than is likely to be used in practice.

Perhaps the most extreme mixture model is one in which every site is allowed to behave differently. In the framework of assuming a single tree topology for all sites, but no more commonality among them, this is usually called the *no common mechanism model*. As a statistical model it is not very useful, since the number of parameters grows with the length of the sequences. As a result longer sequences do not lead to a better ability to recover parameters using Maximum Likelihood, since each additional site introduces many new parameters.

No common mechanism models were introduced primarily to gain theoretical understanding of the interrelationship of different inference methods. Tuffley and Steel (1997) showed that ML inference using a JC no common mechanism model chooses the same tree as Parsimony. Unfortunately this was misinterpreted by some as a justification for Parsimony. Since a no common mechanism model (which doesn't *require* that the sites follow different models, but *allows* it) is perhaps closer to the truth than standard ones, they argued this result showed Parsimony was justified by the standard ML statistical framework. Unfortunately they overlooked the point that ML inference itself was not justified for this model, since the growth in parameters with the number of sites invalidated all the mathematical justifications for ML.

10.5 Exercises

1. Explain the count in formula (10.1).

2. Explain the formula in equation (10.2) by thinking of P_1 and P_2 as giving conditional probabilities of patterns. (Conditioned on what?)

3. How many parameters are needed for a GTR+I model on a binary n-taxon tree, assuming we want the invariable sites to have the same distribution as the stable distribution of variable ones?

4. If data are produced according to by a GTR+I model, but analyzed according to a (misspecified) GTR model, one might expect that the edge lengths of a tree would be estimated to be shorter than they actually were. Why? Explain informally.

5. Suppose a pattern distributions is produced from a JC+I model, with r the proportion of variable sites. With substitution rate $\alpha = 1$ and edge length t_e, what is the matrix giving the joint distribution of patterns on a 1-edge

tree? If the Jukes-Cantor distance (for a model *without* invariable sites) is used to infer the edge length, does it give t_e? If not, is the estimate it gives larger or smaller than the true value?

6. Show that any joint distribution on a 1-edge tree arising from the JC+I model *exactly* matches a joint distribution for the JC model. This means that from 2-taxon comparisons alone it will not be clear whether one needs to include invariable sites in a model used to describe a data set.

7. Show equation (10.4) holds.

8. Show the 8×8 rate matrix $\tilde{Q}$ of the Tuffley-Steel covarion model together with the root distribution vector $\tilde{\mathbf{p}}$ form a time-reversible model.

9. Explain why if M_e is an 8×8 Markov matrix, and H the 8×4 matrix composed of two stacked identity matrices then $M_e H$ will have non-negative entries, with rows summing to 1. In the context of the Tuffley-Steel covarion model, give an interpretation of its entries as conditional probabilities. Also give an interpretation of the entries of H as conditional probabilities.

Chapter 11

Consistency and Long Branch Attraction

Parsimony, distance methods, and maximum likelihood provide three ways of inferring a phylogenetic tree from data. Each has its own strengths, and under some circumstances might be considered the 'best' approach to take. For instance, if there is reason to believe multiple substitutions at a single site are rare or non-existent, then it is hard to argue against the conceptual framework of parsimony. If there are many taxa to relate, a distance algorithm may be the only method that could return a tree in an acceptable amount of time. The ability of maximum likelihood to utilize models, especially those incorporating more complex behaviors such as rate variation, offers hope of performing a more accurate analysis even for more difficult data sets.

However, it is too much to hope that any inference method could give us the true evolutionary tree underlying any data set we analyze with it. Even with the highly adaptable framework of maximum likelihood, it is essential that the model chosen for analysis captures the main features of the true evolutionary process. A *misspecified* model can easily lead to erroneous inference. Likewise, the amount of data we have will impact how well an inference method can perform even if the model completely captures the data generation process. We would like to better understand under what circumstances an inference method is likely to work well, and what circumstances might be problematic for it.

This is, of course, a big task. With few evolutionary histories known beyond all doubt, we can rarely test inference methods on real data to see how well they recover known trees and numerical parameter values. At best we can simulate data (according to some model, perhaps different from the one underlying the inference method) on some specific trees, and then see how well the method reconstructs the trees we chose. This still cannot get at the issue of how close our models are to biological reality, but it does shed light on some of the difficulties of tree inference. Simulation can also give us a feel for how the length of the data sequences affects the performance of methods. One would hope that longer

sequences led to more accurate inference, but that is not necessarily true for all methods.

In this chapter, we discuss one circumstance that is well-understood to cause difficulties in phylogenetic inference: the phenomenon of *long branch attraction*.

Figure 11.1: A metric quartet tree

Consider the metric quartet tree of Figure 11.1. Here two of the pendent edges, in different cherries, are much longer than the other edges. That such a tree can be difficult to correctly infer from data is not too surprising. The two taxa that are the most closely related metrically are not most closely related topologically. In fact, this was the problem that motivated our introduction of neighbor joining as an improvement over UPGMA when we discussed distance methods.

But the neighbor joining algorithm, or even a full maximum likelihood approach to inference, may still perform poorly for real data from such a tree. Our goal in this chapter is to be more precise about this issue.

11.1 Statistical Consistency

Suppose we consider some model of the evolution of sequences along a tree, and some method of inference. To be concrete, we might focus on the Jukes-Cantor DNA model, and Neighbor-Joining with the Jukes-Cantor distance as the method of inference. The most basic question we can ask about this pair is whether we would correctly infer the tree if we analyzed data generated by this model according to this method.

However, this is not a simple yes-or-no question. With a small amount of data, we will certainly sometimes infer the wrong tree. For an extreme thought experiment, imagine data sequences of only a few sites. Then just through randomness, we might find all the sequences were identical. All distances between sequences would then be 0, and the inferred tree would be just a single vertex, representing all taxa.

On the other hand, with much more data in the form of very long sequences, we should expect that on average the observed joint distribution of patterns will match the theoretical joint distribution predicted by the model fairly well. Then when we compute distances, we should obtain ones fairly close to the true

distances determined by the model parameters. Then Neighbor Joining should work well, and is likely to recover the true tree.

In order to formalize these ideas, we make the following definition.

Definition. A method of inference is said to be *statistically consistent* for a particular model if, for any $\epsilon > 0$, the probability that inferred model parameters are within ϵ of the true values approaches 1 as the amount of data approaches infinity. That is, if the process has parameters s, and $\hat{s}$ is the inferred value of the parameters from data produced in n independent trials, then

$$\lim_{n\to\infty} \mathbb{P}(||\hat{s} - s|| < \epsilon) = 1.$$

Consistency should seem like a very basic requirement for a method of inference to be a good one. If a method is *not* consistent, then even if you had access to unlimited amounts of data, you could not reduce your doubts that your inference was wrong to arbitrarily small values. Note also that consistency of a method of inference is always in reference to a specific model. Thus it does not address the very real problem of model misspecification leading to erroneous inference. Consistency deals with an idealized problem, where we know the correct model, and have as much data as we like. If a method does not work well under such circumstances, then we should have serious doubts about its performance in the real world. If it does work well under these idealized circumstances, then we should next consider the impact of both model misspecification and limited data.

11.2 Parsimony and Consistency

Suppose we choose to use parsimony on a 4-taxon dataset to infer a tree. For the 4 taxa a, b, c, d, we have a sequence of characters. In this setting, parsimony reduces to a simple scheme.

First, there are only a few types of characters that are informative. An informative character must have for the 4 taxa a, b, c, d, in order, states

$$xxyy, \quad xyyx, \text{ or } xyxy,$$

where x, y denote two distinct states. So letting n_1, n_2, n_3 denote the count of such characters in a data set.

We wish to compute the parsimony score for this collection of charcters on the 3 quartet trees

$$T_1 = ab|cd, \quad T_2 = ad|bc, \quad T_3 = ac|bd, .$$

For T_1, each character $xxyy$ will give a parsimony score of 1, while characters $xyxy$ and $xyyx$ will produce scores of 2. Taking into account the number of these characters, we find

$$ps(T_1) = n_1 + 2n_2 + 2n_3 = (2n_1 + 2n_2 + 2n_3) - n_1.$$

Similarly,

$$ps(T_2) = 2n_1 + n_2 + 2n_3 = (2n_1 + 2n_2 + 2n_3) - n_2, \tag{11.1}$$
$$ps(T_3) = 2n_1 + 2n_2 + n_3 = (2n_1 + 2n_2 + 2n_3) - n_3.$$

To choose the most parsimonious tree(s) T_i, we therefore simply pick the value(s) of i maximizing n_i.

Now suppose our data are generated by a probabilistic model of the sort in Chapter 6. Then we can compute expected values of the numbers n_i, and see whether parsimony will infer the correct tree. Of course, using expected values of the n_i is essentially the same as imagining we have infinitely long data sequences that are produced exactly in accord with our model. Although we are omitting some details (involving limits, and formal treatments of probabilities of correct inference) needed for a rigorous mathematical treatment, we are dealing with the issue of consistency.

More precisely, we ask the question: If parsimony is applied to longer and longer sequences produced exactly according to the model, do the inferred trees eventually stabilize on the correct one?

The first examples of parameter choices for a Markov model leading to inconsistency of parsimony are due to Felsenstein, for a 2-state model. Although the result has been generalized, for simplicity we follow the original argument.

Theorem 18. For a 2-state Markov model on quartet trees, there are parameters for which parsimony is an inconsistent inference method.

Proof. For the tree in Figure 11.1, place the root ρ at the internal vertex joined to a, b, and denote the other internal vertex by v. Consider the two-state Markov model with parameters

$$\mathbf{p}_\rho = \begin{pmatrix} 1/2 & 1/2 \end{pmatrix},$$
$$M_{(\rho,a)} = M_{(v,d)} = \begin{pmatrix} 1-q & q \\ q & 1-q \end{pmatrix},$$
$$M_{(\rho,b)} = M_{(\rho,v)} = M_{(v,c)} = \begin{pmatrix} 1-p & p \\ p & 1-p \end{pmatrix}.$$

Then the probabilities of the 3 patterns that are informative for parsimony are

$$p_1 = p_{xxyy} = (1-q)^2 p(1-p)^2 + 2q(1-q)p(1-p)^2 + q^2 p^3,$$
$$p_2 = p_{xyyx} = (1-q)^2 p^2(1-p) + 2q(1-q)p^2(1-p) + q^2(1-p)^3, \tag{11.2}$$
$$p_3 = p_{xyxy} = (1-q)^2 p^3 + 2q(1-q)p(1-p)^2 + q^2 p(1-p)^2.$$

But as the sequence length, N, goes to infinity, we have that the proportion of time we see pattern i is $n_i/N \to p_i$. Since parsimony will be consistent only when $n_1 > n_2, n_3$, we have consistency only when $p_1 > p_2, p_3$.

Now, by straightforward algebra, equations (11.2) imply

$$\begin{aligned} p_1 - p_2 &= (1-2p)(p(1-p) - q^2), \\ p_1 - p_3 &= p(1-2p)(1-2q). \end{aligned} \tag{11.3}$$

In these formulas we should only consider values of $p, q \in (0, 1/2)$ as biologically plausible. (See also Exercise 4.) For parameters in this range,

$$1 - 2p > 0, \quad 1 - 2q > 0,$$

so $p_1 > p_3$ always holds. In addition, $p_1 > p_2$ holds precisely when

$$p(1-p) > q^2.$$

However, there are values of p, q in the allowed range where this last inequality does not hold, and so parsimony is inconsistent for such choices. □

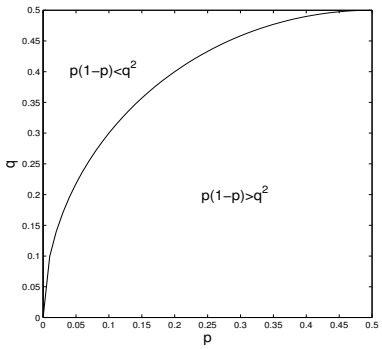

Figure 11.2: Parsimony is consistent only for parameters below the curve.

A graph of the (p, q)-parameter space for the model in the proof is given in Figure 11.2, indicating the regions for which parsimony is a consistent method of inference. We see that if p is sufficiently small in comparison to q, so the internal branch of the tree is very short, we have inconsistency of parsimony.

Note that when the parameters fall in the region of inconsistency (sometimes called the *Felsenstein zone*), the proof has shown $p_2 > p_1$, so the tree parsimony will infer from perfect data is T_2. In other words, the two long branches are erroneously joined, and we have an instance of *long branch attraction*.

An extreme interpretation of this theorem and figure is that since parsimony is sometimes inconsistent, it is not a good method to use. A more moderate view is that parsimony is in fact consistent over quite a wide range of parameters, and we now have some idea of what might lead it to behave poorly. Of course for real data we might not have much idea in advance of how the tree should look, so that whether the tendency for long branches to attract one another is

a problem can be difficult to determine. However, if we infer a tree with several long branches joined to one another, we should consider the possibility.

Of course this theorem does not apply to a 4-state model more appropriate to DNA. However, similar calculations for a Jukes-Cantor model show the same phenomenon; for some edge lengths, parsimony will be a consistent method of inference of a 4-taxon tree topology, but for others it will not.

But the situation can actually be worse than this last statement seems to indicate. In fact, for a 4-taxon tree it is possible to show that under a κ-state generalization of the Jukes-Cantor model, for any $\kappa \geq 2$, the set of distributions of parsimony-informative sites that arises on one tree topology is *exactly* the same as on any other tree topology (Allman, Holder, and Rhodes 2010). In other words, with only 4 taxa the parsimony informative sites alone give no information whatsoever about the tree topology. Thus when parsimony does pick the correct tree, it's essentially just getting lucky. (Note however that when more taxa are used, the parsimony-informative sites do carry information on the tree topology.)

But all these results must be interpreted very carefully. For instance, as Figure 11.2 shows in the 2-state case, parsimony is consistent if all branch lengths are of equal size, and in many other circumstances as well. Thus we have identified a *possible* source of error in inference by parsimony, not a fatal flaw. The problem, of course, is that it may be hard to know ahead of time whether we should expect parsimony to be perform well on any given data set.

Also note that the metric tree used in the proof above is not ultrametric; unless $p = q$ there is no place a root can be located so that all leaves will be equidistant from it. In fact, for a 4-leaf molecular clock tree, it's possible to show parsimony is consistent for simple models. Unfortunately, for larger sets of taxa parsimony can again be inconsistent even assuming the tree is ultrametric.

11.3 Consistency of Distance Methods

The other methods of inferring trees that we've discussed — distance methods and maximum likelihood — are statistically consistent when paired with certain models, though some mild restrictions may be needed on parameter values. We won't give formal proofs of these facts, but will instead sketch the ideas.

Consider first the computation of distances from data sequences produced according to a specific model. Then as sequence length grows, it is easy to show that provided we use a distance formula that is associated with the model generating the data, we will infer distances that approach the true ones for the parameter choice. (Thus if our data are generated by a Kimura 2-parameter model, we may use Kimura 2-, or 3-parameter distances, or the log-det distance, but not the Jukes-Cantor distance.) This basic fact follows from the continuity of the distance formulas.

Once we have distances, suppose we use the Neighbor Joining algorithm to infer a tree. We've already claimed in Theorem 16 (and shown in Exercise 24

of Chapter 5) that Neighbor Joining recovers the correct tree provided we have *exact* distances from a binary tree with all positive edge lengths. It's necessary, then, to show that from dissimilarities sufficiently close to the exact distances, we recover the same tree. So what is needed is in essence a statement that the output of the algorithm is a continuous function of the input dissimilarities, at least in the neighborhood of exact distances for binary trees with all positive edge lengths. This is certainly plausible, and in fact can be rigorously proved. For other distance methods, such as the least-squares approach, consistency can be established similarly

Note that the requirements that the tree be binary and edge lengths be positive are the mild restrictions that we indicated were needed beforehand.

An important point in this, however, is that we need to be able infer distances correctly. If data are generated by a mixture model, but distances are computed by a simpler model, then there are no guarantees that further processing of these incorrect distances will lead to a correct tree. Thus consistency of any distance method is likely to depend on the consistency of the inference of pairwise distances between taxa. Since distance formulas exist only for the most basic models, we will be limited to these if we insist on consistency.

11.4 Consistency of Maximum Likelihood

There are rather general proofs that maximum likelihood is a consistent method of statistical inference in many settings, and these can be modified for phylogenetic models. However, these proofs require that one first establish that the model has *identifiable* parameters. In the phylogenetic setting, this means that any joint distribution of patterns at the leaves of a tree arises from a unique tree and choice of numerical parameters. If different choices of trees and numerical parameters lead to the same joint distribution of patterns, then even with large amounts of data they would be indistinguishable, and no method would be able to consistently infer parameters. Thus a lack of identifiability for a model would cause *any* method to be inconsistent. Maximum likelihood has the fortunate feature that it is generally consistent without any additional substantive requirements on the model beyond identifiability.

In fact, parameters for the general Markov model are not strictly identifiable. For instance, suppose two n-taxon tree topologies are chosen that are one NNI move apart. For numerical parameters on each, choose the same root distribution, and the same Markov matrices on all edges not affected by the NNI move. On the edge that the NNI move collapsed on the first tree, and the new edge it introduced in the second, let the Markov matrix be the identity. Then these two trees and parameter choices will produce exactly the same joint distribution of patterns at the leaves. The reason for this, of course, is that the choice of the identity as a Markov matrix means no substitutions occur on the edges affected by the NNI, so both trees produce sequences as if there is an unresolved polytomy, with 4 edges emerging from a single vertex. The simple way to rule this problem out is of course to not allow Markov matrices to be

the identity.

By imposing restrictions of this sort, under the general Markov model the tree topology is identifiable, by means of the log-det distance and 4-point condition, for instance. For this, it is sufficient to require that the tree be binary and all Markov matrices have determinant $\neq \pm 1, 0$. This rules out not only the identity matrix for Markov matrices, but also some other possibilities that are fortunately not relevant biologically. For instance, in a GTR submodel, the only way a Markov matrix could have determinant 0 is if the edge length were infinite.

There remain other non-identifiability issues, though. For instance, one can permute the bases at an internal node of the tree, adjusting the Markov parameters on incident edges accordingly by permuting rows or columns, and again not change the joint distribution (Exercise 7). This gives only finitely many parameter choices for each joint distribution, though. Moreover we can make a unique choice from these by imposing biologically reasonable assumptions that the diagonal entries of all matrices be the largest in their rows.

Even after eliminating these issues, it is not easy to show the parameters are identifiable. The essential difficulty is that phylogenetic models have hidden variables, representing the states at internal nodes of the tree, which cannot be observed. These greatly complicate the form of the model parameterization, leading to the many-term sum of products we saw in Chapter 6. Moreover, there are similar models with hidden variables outside of phylogenetics that are *not* identifiable.

Currently, maximum likelihood has been proven to be consistent for the following models, by first showing the models are identifiable. For some of these models, there are minor technical conditions that must be placed on parameters, about which we will not be explicit.

- the GM model, and its submodels such as GTR, K2P, JC
- GTR+Γ, and its submodels such as K2P+Γ, JC+Γ
- GTR+Γ+I, except when the GTR component is JC or F81, when two trees differing by an NNI may give the same distribution
- GM+I, and submodels such as GTR+I, K2P+I, JC+I
- rate-variation and covarion models with c classes, provided c is smaller than the number of observable states (i.e., $c < 4$ for DNA models, $c < 19$ for proteins, etc.)

Much more general mixture models on a single tree topology are also known to be identifiable for *generic* choices of parameters, provided the number of components is less than a bound depending exponentially on the number of taxa. 'Generic' here means that if the numerical parameters are chosen at random, then almost certainly they will be identifiable. Mixtures in which components have different tree topologies are similarly identifiable, provided the trees all have two 'deep splits' in common. (Rhodes and Sullivant, 2012)

These theoretical results safely cover all models used routinely for data analysis, and many that are being explored in less routine investigations. And while it is likely that even more complex models than these, that have yet to be proposed, have identifiable parameters, one should not forget that there are real limits. For instance, under the no-common-mechanism model mentioned in Chapter 10, the tree topology is not identifiable. Thus while we might find such a model appealing for its biological realism, it is not useful for data analysis.

11.5 Performance with Misspecified models

While statistical consistency is certainly desirable for an inference method, establishing it does not lay to rest all concerns we should have. First, a claim of consistency is a statement about behavior under idealized conditions: If we use the correct model, and have access to as much data as we like, then we are likely to draw the right inferences. If, say, we attempt to use maximum likelihood with a Jukes-Cantor model for inference, and the data actually is not fit well by that model, the consistency results above give us no guarantees. Since biological data are unlikely to be described perfectly by any simple model, we certainly have a violation of assumptions in any real-world application. How that violation of assumptions effects inference results is a question of *robustness*. While experience with data analysis generally indicates that the inference of the tree topology may be fairly robust to variations in choice of models, numerical parameters inferred under different models often vary more widely.

There are also circumstances that haven been found where an analysis with a misspecified model leads to errors in tree inference. For instance, Shavit Grievink, Penny, Hendy, and Holland (2009) simulated data using a covarion version of the Jukes-Cantor model, in which the proportion of invariable sites changed over the tree. Though plausible as capturing reasonable biological behavior, this is not a model that has been implemented for data anlaysis in any software, nor studied theoretically. When a Bayesian analysis of the simulated data was performed under the model's closest implemented covarion cousin, which did *not* allow for changing proportions in the covarion classes, serious errors were made in recovering the tree topology. The extent to which such processes may be misleading us with analysis of real data is simply not known.

While it can be reassuring to analyze a data set under several models and see that the inferred parameters of interest are similar, one should never forget that the models currently available in software are limited. If they do not capture some key process in the data's production, they might all give similar results yet still be misleading.

11.6 Performance on Finite-length Sequences

Even with a correctly specified model used to analyze data, in the real world sequences are always of finite length, and therefore short in comparison to those

with which statistical consistency is concerned.

How well various methods perform on sequences of the lengths similar to experimental data sets has been investigated through simulation. We can choose a model, tree, and numerical parameters, and simulate data according to these. Then we can apply the inference method to the simulated data, and see if it recovers the original tree. (See Exercise 8.) If we use the model which generated the simulated data to analyze it, then *only* the effect of sequence length will be investigated.

One important finding of such simulation is that the long branch attraction phenomenon seems quite universal, regardless of the method of inference. For instance, for any fixed sequence length, there is a region of parameter space much like the upper left corner of Figure 11.2 in which we often infer the wrong tree. The precise shape and size depends on the method of inference and the sequence length, but the region is there, even for maximum likelihood. Because maximum likelihood is consistent, the size of the region must become smaller if the sequences are made longer. However, in practice we may not be able to obtain long enough data sequences whose evolution can be modeled well, so the assurance of statistical consistency may not be helpful.

Several long edges leading from the ends of a short edge, then, are quite generally problematic. It is wise to keep this is mind when inference produces a tree with several long edges leading from a common vertex. When such a tree is inferred, it might be possible to include additional taxa in the data set in order to create additional bifurcations breaking up the long edges, and with an expanded data set, we may be able to better infer a tree. However, there may be no taxa available to break up the long edges, so this problem cannot always be overcome.

11.7 Exercises

1. Explain the computation of the parsimony scores in equations (11.1) for the quartet trees.

2. Check the formulas for p_1, p_2, p_3 in equations (11.2) in the proof of Theorem 18.

3. Check the formulas for $p_1 - p_2$ and $p_1 - p_3$ in equations (11.3) in the proof of Theorem 18 (using software, or by hand).

4. Show that if the 2-state Markov matrix

$$M(p) = \begin{pmatrix} 1-p & p \\ p & 1-p \end{pmatrix}$$

is of the form $M = e^{Qt}$ for some non-zero rate matrix Q, $t > 0$, then $p \in (0, 1/2)$. Why is this biologically plausible?

5. For the Markov model in the proof of Theorem 18, explain how Figure 11.2 fits with the intuition that as long as sufficiently few mutations occur,

parsimony should infer the correct tree. Is this intuition strictly correct? Show the curve $p(1-p) = q^2$ has a vertical tangent at (0,0), and explain why this is relevant.

6. Assuming we use a valid distance formula for our model, explain informally why UPGMA will be statistically consistent if the underlying tree has all leaves equidistant from the root, but will not be without this assumption on the tree.

7. Consider a 3-taxon tree, with root ρ at the central vertex. Suppose for the general Markov model on this tree we have parameters $\mathbf{p}_\rho$, M_1, M_2, M_3, producing a certain joint distribution of states at the leaves. If P is a permutation matrix and $\mathbf{p}'_\rho = \mathbf{p}_\rho P$, what Markov matrices M'_1, M'_2, M'_3 will, along with $\mathbf{p}'_\rho$, produce the same joint distribution as the original parameters?

8. Using software, simulate some sequences of length 300 bases according to a Jukes-Cantor model on the tree in Figure 11.1 using a variety of parameter choices (p, q) as in the proof of Theorem 18. Then use Neighbor Joining with the Jukes-Cantor distance to infer a tree. Find parameter choices for which the method seems to usually give the correct tree, and other parameter choices for which you usually see long branch attraction.

   If you are ambitious, perform many simulations for enough values of (p, q) to produce an empirical diagram like Figure 11.2, showing when this approach to this inference problem usually produces the correct tree.

Chapter 12

Bayesian Inference

While Maximum Likelihood for model-based inference in phylogenetics is common, the Bayesian framework is also widely used. Like Maximum Likelihood, it is a general approach, that can be applied in any statistical setting where a probabilistic model has been formulated. Although Bayesian methods are not new, computational difficulties often prevented their application for complex problems until recent decades.

While there are significant philosophical differences between these two frameworks, their mathematical formulations are not so far apart. Despite a long-running debate between some of the supporters of each, most practitioners adopt a more pragmatic approach, accepting both as reasonable.

12.1 Bayes' theorem and Bayesian inference

As in Chapter 8, our basic problem of statistical inference is the following: Having formulated a probabilistic model of a process, where the model depends on some unknown values of parameters, from a data set we wish to infer the parameter values. In phylogenetics, this typically would mean we have chosen to use a model, such as GTR, to describe the evolution of a collection of sequences along a tree, and the data are the observed sequences from the leaves of the tree. The unknown parameters are all of the numerical parameters of the GTR model (the stable base composition and the relative rates) as well as the topological tree and all of its branch lengths. But that is a rather complicated model, so it's best to begin with a simpler example. We return to the same one used in Chapter 8.

Example. Our experiment is the toss of a (possibly unfair) coin. Our model is simply that the toss will produce heads with probability p and tails with probability $1 - p$. The parameter here is p, which might have any numerical value from 0 to 1.

With Maximum Likelihood, we sought a single best value of p based on the data, and ultimately produced the estimate $\hat{p}$ =(number of heads)/(number of

tosses).

In Bayesian inference we adopt a more elaborate viewpoint that we should not necessarily give a single estimate for $\hat{p}$, but rather a range of values, together with a measure of how much support each has. After all, if we obtained 63 heads out of 100 it is quite reasonable that p was not 0.63. Perhaps it was 0.60, or 0.58, or even 0.50. As we consider values further from .63, we may feel that they become less reasonable, but we are not inclined to completely rule them out.

To capture this more formally, the goal of Bayesian inference is not to infer a single best choice of parameters to fit data, but rather to associate to each possible choice of parameter values a probability of those being the ones that produced the data. This probability measures the support for the parameter values, with values near 1 indicating strong support, and those near 0 indicating essentially no support.

The key to obtaining these probabilities is *Bayes' Theorem*, which has already appeared in equation (8.2), though we repeat it more carefully here: For any two events A and B,

$$\mathcal{P}(A \mid B)\mathcal{P}(B) = \mathcal{P}(A \text{ and } B) = \mathcal{P}(B \mid A)\mathcal{P}(A),$$

so solving for $\mathcal{P}(A \mid B)$ yields

$$\mathcal{P}(A \mid B) = \frac{\mathcal{P}(B \mid A)\mathcal{P}(A)}{\mathcal{P}(B)}.$$

If we think of both our data and the value of the parameters p as being probabilistically determined, then we obtain the special case

$$\mathcal{P}(p \mid \text{data}) = \frac{\mathcal{P}(\text{data} \mid p)\mathcal{P}(p)}{\mathcal{P}(\text{data})}. \tag{12.1}$$

Now the Bayesian perspective requires that we give meanings to the terms on the right hand side of this equation, in order that we can compute the left hand side.

We begin with $\mathcal{P}(p)$, the probability of a specific parameter value p. In the likelihood framework, the viewpoint is that there simply is some unknown value of p, and it doesn't really make sense to talk about its probability. From a Bayesian perspective, however, we think of $\mathcal{P}(p)$ as capturing the support we feel a value p has, or equivalently, our belief that it is the true value. Since $\mathcal{P}(p)$ does not depend on the data, in this equation it must represent support for values of p before we consider the data. It is therefore called the *prior distribution* of p, since it captures our *a priori* beliefs. In contrast $\mathcal{P}(p \mid \text{data})$ measures support for p after the data has been taken into account. It is called the *posterior distribution* of p since it captures our *a posteriori* beliefs.

Already the broad outline of the Bayesian approach has been given. We begin an analysis of data by specifying a prior distribution on the parameters, which indicates our current beliefs in what values are reasonable. Then equation

(12.1) is used to take into account both the data and our prior beliefs to produce updated, posterior beliefs.

To do this, though we must examine the other terms in equation (12.1). In the numerator of the right side, we have $\mathcal{P}(\text{data} \mid p)$, which is simply the likelihood function that formed the basis for Maximum Likelihood inference, and thus something we know how to compute.

In the denominator, we have $\mathcal{P}(\text{data})$. In the likelihood framework this is a rather nonsensical concept, since after all, we collected the data so if there is a probability associated to it, it must be 1. From the Bayesian viewpoint, however, we have prior beliefs about the parameter values, and so we could use them to compute the probability of obtaining any specific data. More specifically, we can set

$$\mathcal{P}(\text{data}) = \sum_p \mathcal{P}(\text{data} \mid p)\mathcal{P}(p),$$

where the sum is over all possible choices of the parameter values.

Thus our final formula for how we update the prior distribution to obtain the posterior one is

$$\mathcal{P}(p \mid \text{data}) = \frac{\mathcal{P}(\text{data} \mid p)\mathcal{P}(p)}{\sum_p \mathcal{P}(\text{data} \mid p)\mathcal{P}(p)}, \tag{12.2}$$

Example (1). Returning to the coin toss example, for simplicity let's suppose we know the coin that is flipped is weighted in one of 3 ways: p is either 1/4, 1/2, or 3/4. With no data collected, we might choose for a prior the probabilities 1/3, 1/3, 1/3 that p has each of these values. (Note that while p itself is a probability, the prior assigns probabilities to each of its possible values, so the prior, in this case, gives us the probability of a probability.)

Now we collect some data by flipping the coin 3 times, obtaining the sequence HHT. The likelihood is computed as in Chapter 8 to be

$$\mathcal{P}(\text{HHT} \mid p) = p^2(1-p).$$

We use this, and the prior $\mathcal{P}(p)$ to compute the denominator $\mathcal{P}(\text{data})$ in equation (12.2) as

$$\begin{aligned}\mathcal{P}(\text{HHT}) &= \mathcal{P}(\text{HHT} \mid p = 1/4)\mathcal{P}(p = 1/4) + \mathcal{P}(\text{HHT} \mid p = 1/2)\mathcal{P}(p = 1/2) \\ &\quad + \mathcal{P}(\text{HHT} \mid p = 3/4)\mathcal{P}(p = 3/4) \\ &= \left(\frac{1}{4}\right)^2 \left(\frac{3}{4}\right)\left(\frac{1}{3}\right) + \left(\frac{1}{2}\right)^2 \left(\frac{1}{2}\right)\left(\frac{1}{3}\right) + \left(\frac{3}{4}\right)^2 \left(\frac{1}{4}\right)\left(\frac{1}{3}\right) \\ &= \frac{3+8+9}{4^3 \cdot 3} = \frac{20}{192} = 0.104166\ldots.\end{aligned}$$

Finally we use equation (12.2) for each of the possible values of p to obtain

the posterior distribution:

$$\begin{aligned}\mathcal{P}(p = 1/4 \mid \text{data}) &= \frac{\mathcal{P}(\text{data} \mid p = 1/4)\mathcal{P}(p = 1/4)}{\sum_p \mathcal{P}(\text{data} \mid p)\mathcal{P}(p)} \\ &= \frac{\left(\frac{1}{4}\right)^2 \left(\frac{3}{4}\right) \left(\frac{1}{3}\right)}{\frac{20}{192}} \\ &= \frac{3}{192}\frac{192}{20} = \frac{3}{20} = 0.15\end{aligned}$$

$$\begin{aligned}\mathcal{P}(p = 1/2 \mid \text{data}) &= \frac{\mathcal{P}(\text{data} \mid p = 1/2)\mathcal{P}(p = 1/2}{\sum_p \mathcal{P}(\text{data} \mid p)\mathcal{P}(p)} \\ &= \frac{\left(\frac{1}{2}\right)^2 \left(\frac{1}{2}\right) \left(\frac{1}{3}\right)}{\frac{20}{192}} \\ &= \frac{8}{192}\frac{192}{20} = \frac{8}{20} = 0.40\end{aligned}$$

$$\begin{aligned}\mathcal{P}(p = 3/4 \mid \text{data}) &= \frac{\mathcal{P}(\text{data} \mid p = 3/4)\mathcal{P}(p = 3/4)}{\sum_p \mathcal{P}(\text{data} \mid p)\mathcal{P}(p)} \\ &= \frac{\left(\frac{3}{4}\right)^2 \left(\frac{1}{4}\right) \left(\frac{1}{3}\right)}{\frac{20}{192}} \\ &= \frac{9}{192}\frac{192}{20} = \frac{9}{20} = 0.45.\end{aligned}$$

Thus the posterior probabilities of the coin having probability of heads 1/4, 1/2, and 3/4, are 0.15, 0.40, and 0.45, respectively. Note that these add to 1, as they must since they specify a probability distribution.

In comparison to the prior, this posterior indicates we have shifted our belief to one where p is larger. The most probable value of p is 3/4, but there is not much more support for that than for $p = 1/2$. On the other hand, the support for $p = 1/4$ has decreased significantly to less than half its prior probability.

As should be clear, computing posterior probabilities using equation (12.2) in even simple examples requires a long computation. In addition to needing the same likelihood function as was used in Chapter 8, and a prior distribution on the parameters, the denominator involves a sum of products of these over all possible parameter values. If in this example we had said the probability p of heads could have any of the values $0.1, 0.2, 0.3, \ldots, 0.9$, then there would have been 9 summands rather than 3. Computing this denominator thus becomes a major computational hurdle when there are many possible values of the parameters.

Example (2). In the coins toss example above, we began by saying there were only 3 possible values for the parameter p. For a more elaborate example, we

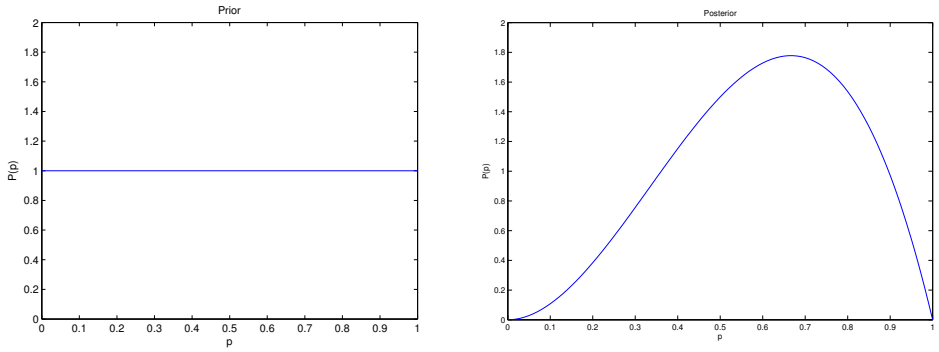

Figure 12.1: (a) A flat prior for a coin with probability p of heads, and (b) the posterior after the observation HHT.

could instead say p could have *any* of the continuum of values between 0 and 1. Then we must specify the prior distribution by a probability density function, $f(p)$, defined on this interval. The probability of the parameter lying in a small interval of size dp around the value p is then $\mathcal{P}(p) = f(p)dp$.

To express complete ignorance of the values of p, we might choose this to be a constant function $f(p) = c$. The specific value of c is determined by the need for the total probability, $\int_0^1 f(p)\,dp$ to be 1, so we use $c = 1$. (Any probability density for a continuous parameter must have total area 1 under its graph.) The formula for the posterior density then becomes

$$f(p \mid \text{data}) = \frac{\mathcal{P}(\text{data} \mid p) f(p)}{\int_0^1 \mathcal{P}(\text{data} \mid p) f(p)\,dp},$$

so

$$f(p \mid \text{HHT}) = \frac{p^2(1-p)\cdot 1}{\int_0^1 p^2(1-p)\cdot 1\,dp}.$$

Since

$$\int_0^1 p^2(1-p)\cdot 1\,dp = \int_0^1 p^2 - p^3\,dp = \frac{p^3}{3} - \frac{p^4}{4}\bigg|_0^1 = \frac{1}{3} - \frac{1}{4} = \frac{1}{12},$$

the posterior density is

$$f(p \mid \text{HHT}) = 12p^2(1-p).$$

Graphs of the prior and posterior are show in Figure 12.1. Note how there is a significant shift toward probabilities in the vicinity of 2/3 being the most probable.

To summarize, and contrast the Bayesian inference approach to the of Maximum Likelihood, we list a few key points

1. Both are model-based methods in which a probabilistic description of the data generation process is assumed.

2. Both involve computing $\mathcal{P}(\text{data} \mid p)$ for parameter values p. In Maximum Likelihood, this is the Likelihood function we seek to maximize; in a Bayesian analysis it appears in the process of computing the posterior distribution.

3. A Bayesian analysis requires specifying a prior distribution of the parameters we seek to infer, which expresses our beliefs in what parameter values are likely before the data are considered. For Maximum Likelihood we do not need to express these prior beliefs, and in fact cannot take such prior beliefs into account.

4. Maximum Likelihood produces a single estimate, called a *point estimate*, of the parameters we infer. A Bayesian analysis produces a distribution of estimates, the posterior distribution, which indicates differing levels of support for different parameter values.

12.2 Prior Distributions

The requirement that we specify a prior for a Bayesian analysis is a major difference from what is needed to perform Maximum Likelihood. Depending on one's viewpoint and the application, this can be either a positive or negative feature.

In the majority of phylogenetic analyses, the priors are chosen to express as much ignorance as possible. Such a prior is often called *uninformative*, though it may carry some information, such as the fact that a parameter must lie in a certain range. For example, the prior depicted in Figure 12.1 indicates that p lies between 0 and 1, but its "flatness" or uniformity indicates that we have no further information on its likely value. By choosing uninformative priors, one hopes that the posterior distribution will indicate signal in the data alone, and not any assumptions that a different prior might express.

For a few of the parameters in a phylogenetic model, creating an uninformative prior is easy. For instance if an model allowing some invariable sites is used, then the proportion of invariable sites can be given such a uniform prior across the range 0 to 1, just as in the coin toss example.

For the base distribution, an uninformative prior is not much more complicated. Since $\mathbf{p} = (p_A, p_G, p_C, p_T)$ with $p_A + p_G + p_C + p_T = 1$, we can use the uniform distribution which assigns to each choice of $\mathbf{p}$ the same value, 1 divided by the volume of the region in space with $x, y, z \geq 0, x + y + z \leq 1$. Since this is a bounded region, its volume is finite. This is a special case of the *Dirichlet distribution*, which has several parameters (or, as they are often called, *hyperparameters* since these are parameters of the prior and not of the phylogenetic model) that can be varied to range from the uniform distribution to ones increasingly concentrated at any point. (See Exercise 5.) The Dirichlet

distribution can thus be used to provide as informative a prior as is desired. A similar uninformative prior can be used for substitution rates, since they can be normalized to sum to 1, by adopting an appropriate time scale.

Priors for edge lengths are more problematic, since lengths can range over all non-negative real numbers. Since this set is unbounded, a uniform distribution is not possible as a valid prior, since a constant function on an unbounded interval would not have area 1 under its graph. One possibility is to use a uniform distribution on a large interval from 0 up to a value believed to safely exceed any plausible edge length. However, Yang and Rannala (2005) have, through a mixture of theoretical analysis and simulation studies, shown this can lead to posteriors that are concentrated on longer edge lengths, and thus perhaps bias results in a way that was not intended. An alternative is to use a prior that decays exponentially with branch length. Although that concentrates more probability on short edge lengths, that may actually be a better representation of what we might consider to be uninformative than a flat distribution.

Finally, for the tree topology the simplest approach is that we make all binary topologies have the same probability. Since there are only finitely many possibilities, this gives a valid prior.

These choices of priors have all been made in the hopes of expressing little to no information that will be used in the Bayesian analysis. Whether they fully succeed is a difficult issue. For instance, do we really believe all binary trees are really equally likely? What about the prior for branch lengths where there seems to be no obvious way to even say what truly uninformative would mean? This dependence on priors when we may have no objective way of choosing them is the reason some feel discomfort with Bayesian methods in general.

From a more pragmatic perspective though, we can make reasonable choices of priors, and then examine the resulting posterior distribution. If the analysis gives posteriors that are very different from the priors, then we can be fairly confident that the data have overwhelmed whatever information we inadvertently put into the priors.

12.3 MCMC Methods

In phylogenetics, as in many other modern applications of Bayesian inference, the computation of the posterior distribution from equation (12.2) is not done directly. Attempting to evaluate the denominator in that formula would be computationally intractable, since it requires summing terms for every possible choice of parameters. The tree topology parameter alone has an enormous number of possible values if the number of taxa is at all large, and then there are many numerical parameters as well.

Instead a different computationally-intensive process, called a *Markov chain Monte Carlo* (MCMC) method, is followed that rather than giving exact values of the posterior distribution attempts to give a sample chosen from that distribution. We say 'attempts' here, since the process that is followed will, provided it runs long enough, provably converge to a process that gives such a

sample. Thus the process is first run for a 'burn in' period of many iterations, in order to get from its starting point to a point more typical of the asymptotic behavior. There are no useful theoretical results on how long the burn-in period must run before it gives a sample that approximates one from the true distribution, but there are heuristic guides or diagnostics that are provided by software developers. The sample produced during burn in are then ignored, and as the process continues to run a new large sample is collected. Provided it is large enough, this new sample should closely approximate the true distribution of the posterior.

The name 'Markov chain Monte Carlo' signifies the two main ideas behind this process. The reference to the casinos of Monte Carlo indicates that it a random process is followed, and so only by taking a large sample can we be reasonably confident that we have a good approximation to the posterior. The Markov chain refers to the more theoretical underpinnings of the process, in which there is a collection of states, and conditional probabilities indicating how we move from state to state.

In the phylogenetics applications, the *Metropolis-Hastings* algorithm is the MCMC method usually used. The states for the process are a choice of parameters, *i.e.* a tree together with specifications of all other numerical parameters. We start at one such state, and then move to a new one by a certain probabilistic rule. An overview of the slightly simpler Metropolis procedure is the following:

1. Create some probabilistic *proposal process* that given any state p_1 picks a new state p_2 with probability given by some function $\mathcal{P}(p_2 \mid p_1)$. The only requirements on this process is that $\mathcal{P}(p_2 \mid p_1) = \mathcal{P}(p_1 \mid p_2)$, so that the probability of jumping from one state to another is symmetric, and that the probability of proposing any state from any other (perhaps by a succession of proposals) is non-zero.

2. Choose some starting state p_0.

3. Repeat the following steps:

   (a) If the current state is p_n, propose a new state p according to the proposal process.

   (b) Compute the acceptance ratio

   $$\alpha = \frac{\mathcal{P}(p \mid \text{data})}{\mathcal{P}(p_n \mid \text{data})}.$$

   (c) If $\alpha \geq 1$, accept the proposal and make the current state $p_{n+1} = p$, and return to step 3a.

   (d) If $\alpha < 1$, then flip a coin that is weighted to give H with probability α, and T with probability $1 - \alpha$.

   (e) If the coin gives H, then accept the proposal by making the current state $p_{n+1} = p$. If the coin gives T, then reject the proposal by making the current state $p_{n+1} = p_n$.

(f) Return to step 3a.

Informally, the acceptance of a proposed state is done in such a way that we will always accept one that has higher probability in the posterior than our current state. However, we will also accept ones that are less probable, but not always. Thus it is plausible that the process could tend toward sampling from the posterior. This in fact can be proved.

Actually, the formula for α above makes it appear that we must already know the posterior to use the algorithm. In fact, the key point is that we do *not* need to know it, since the posterior only appears in a ratio. While the denominator in the formula (12.2) for the posterior is what is intractable to compute, it cancels out in the formula for α:

$$\alpha = \frac{\mathcal{P}(p \mid \text{data})}{\mathcal{P}(p_n \mid \text{data})} = \frac{\frac{\mathcal{P}(\text{data} \mid p)\mathcal{P}(p)}{\sum_p \mathcal{P}(\text{data} \mid p)\mathcal{P}(p)}}{\frac{\mathcal{P}(\text{data} \mid p_n)\mathcal{P}(p_n)}{\sum_p \mathcal{P}(\text{data} \mid p)\mathcal{P}(p)}} = \frac{\mathcal{P}(\text{data} \mid p)\mathcal{P}(p)}{\mathcal{P}(\text{data} \mid p_n)\mathcal{P}(p_n)}.$$

Thus computing the acceptance ratio depends only on being able to compute the likelihood function, and the prior.

Notice that almost any proposal function can be used with the same theoretical guarantee of eventually approximating the posterior distribution. The Hastings modification of the algorithm even allows a non-symmetric proposal. In practice, however, the design of a good proposal process can have a large effect on how the algorithm performs. For instance, considering only tree topologies, we could imagine proposal processes that only used NNI moves, or instead took larger steps through tree space. If the posterior turns out to be highly concentrated on a single topological tree, then using NNI moves alone may work well after the burn-in period has passed, but might result in a longer burn-in as small steps are taken from the starting tree to get to the one with high posterior probability. On the other hand, if large steps are taken, we should expect that one we are past burn-in that most proposed trees will be rejected, resulting in a longer run time to build up a good sample.

12.4 Summarizing Posterior Distributions

A great strength of Bayesian inference is that it yields a distribution of estimates rather than a single one. In simple circumstances, such as the coin toss example in section 12.1, the posterior can be communicated by a graph, as in Figure 12.1. If the area under this graph is closely concentrated above a small interval on the horizontal axis, then a glance indicates there is strong support for the parameter value being in that interval. If the area is more spread out, then we quickly see there are a large range of reasonable parameter values, and the data were not able to give us a tight conclusion.

In the phylogenetic setting, however, where a parameter choice may mean a metric tree together with a base distribution, a rate matrix, and possibly several

other numerical parameters, things are a little more complicated. Rather than a single parameter which could be placed on a single axis, we have a rather large number of parameters. Moreover, the tree topology is also a parameter, and there is no natural way to depict that on an axis in a graph.

When faced with difficulties in presenting the full posterior distribution, one alternative is to report the single choice of parameters (assuming there is only one) that maximizes the posterior. This *maximum a posteriori* (MAP) estimate is the most probable choice of parameters given the data and prior. Reporting only it, however, throws away most of the information in the posterior distribution. Most importantly, we lose an indication of the spread of the distribution across other parameter values.

For a distribution of phylogenetic trees, a reasonable approach is to report the MAP tree topology, but then further indicate the support for each split in the unrooted tree case, or for each clade if the tree is rooted, in the full distribution. Each tree topology has a associated posterior probability (obtained by restricting the distribution to trees of the given topology and then integrating over all numerical parameter values). The probability of a split or clade is then the sum of the probabilities of the topological trees displaying that split or clade. Thus a probability of 1 indicates the split or clade appears in every tree that appears with non-zero probability in the posterior, while lower probabilities indicate appearance in a lower proportion of the trees. Thus each edge of the reported tree can be assigned the probability of the associated split, indicating its support across the full posterior distribution.

Instead of the MAP tree, one could instead use a consensus tree for the splits or clades. Here the posterior probabilities of each possible split or clade in the full distribution would be used to determine which splits or clades make the cutoffs for building the consensus tree. A strict consensus thus includes only those with probability 1 in the full distribution, while a majority-rule consensus uses all those with probabilities strictly greater than 0.5.

Of course one hopes that a Bayesian phylogenetic analysis will lead to very strong support for a single tree. If the MAP tree topology has probability 1, then the posterior is entirely concentrated on a single tree, and any sort of a consensus tree will have the same topology. Even if the MAP tree topology only has probability slight larger than .5, the majority rule consensus tree will agree with it.

To give metric information and other numerical parameters for the reported tree there are also several possibilities. On the MAP tree topology one could simply report MAP edge lengths, by either fixing the MAP topology and then choosing the collection of edge lengths that simultaneously maximize the restricted posterior distribution, or alternatively, for each edge integrate the restricted distribution over all other numerical parameters and then report edge lengths maximizing this marginal distribution. For a consensus tree one can report averages (weighted by the probabilities of the posterior distribution) of edge lengths across the different trees displaying the associated split or clade. If the posterior is highly concentrated, there should be very little difference in

these approaches.

Of course not every analysis leads to a highly concentrated posterior distribution. In such a case rather than depending on some simple summaries, it is wise to look carefully at the more detailed distribution, which software will typically make available for examination.

12.5 Exercises

1. a) In example (1) of the computation of the posterior distribution in section 12.1, the data were taken to be HHT. Redo the calculation using the same uniform prior on 1/4, 1/2, 3/4, but assuming the data are a sequence of 300 coin tosses, with 200 heads and 100 tails. Compare the posterior distribution you obtain to the one in the example. Are they the same? If not, explain why any differences you see are reasonable.

   b) Repeat part (a), but for example (2), with a uniform prior on the interval $0 \leq p \leq 1$.

2. Suppose a prior is such that for some specific parameter value p_0, $\mathcal{P}(p = p_0) = 0$. From equation (12.2) explain why $\mathcal{P}(p = p_0 \mid \text{data})$, the posterior probability that $p = p_0$ will also be 0. (Thus in a Bayesian framework, if something is viewed as impossible, then no amount of data will change that view.)

3. Suppose a prior is such that for some specific parameter value p_0, $\mathcal{P}(p = p_0) = 1$. From equation (12.2) explain why $\mathcal{P}(p = p_0 \mid \text{data})$, the posterior probability that $p = p_0$ will also be 1. (Thus in a Bayesian framework, if something is viewed as definite, then no amount of data will change that view.)

4. In the continuous-parameter Example(2) of Section 12.1, the posterior is shown to be $\mathcal{P}(p \mid \text{HHT}) = 12p^2(1-p)$.

   a) For what values of p is this posterior 0? Why is this reasonable for these data?

   b) Calculate the MAP estimate of p. How does it compare to the ML estimate of p? (Note: this relationship depended on the specific choice of prior that was used.)

5. The *Dirichlet distribution* for $(x_1, x_2, \ldots, x_k)$ with $x_1 + x_2 + \cdots + x_k = 1$, $x_i \geq 0$, is specified by the probability density function

$$f(x_1, x_2, \ldots, x_k; \alpha_1, \ldots, \alpha_k) = \frac{1}{B(\boldsymbol{\alpha})} \prod_{i=1}^{k} x_i^{\alpha_i - 1},$$

   where $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_k)$ is a vector of parameters and

$$B(\boldsymbol{\alpha}) = \frac{\prod_{i=1}^{k} \Gamma(\alpha_i)}{\Gamma(\sum_{i=1}^{k} \alpha_i)}$$

is a normalizing constant so that the total probability is 1.

a) Show that if $\alpha_1 = \cdots = \alpha_k = 1$ this is a uniform distribution, and thus when $k = 4$ gives the uninformative prior for base distributions.

b) To better understand the distribution for other values of $\boldsymbol{\alpha}$, consider $k = 3$. Produce 3-dimensional plots of the unnormalized version of the distribution, $f(x_1, x_2, x_3) = x_1^{\alpha_1 - 1} x_2^{\alpha_2 - 1}(1 - x_1 - x_2)^{\alpha_3 - 1}$, for $\boldsymbol{\alpha} = (1, 1, 1)$, $(10, 10, 10)$, $(100, 100, 100)$. Roughly give the location of the peak, and explain how the concentration of probability for the Dirichlet distribution changes for these values of $\boldsymbol{\alpha}$. (Note: the only relevant portion of the graph is where $x_1 + x_2 \leq 1$.)

c) Produce 3-dimensional plots of the unnormalized version of the distribution, $f(x_1, x_2, x_3) = x_1^{\alpha_1 - 1} x_2^{\alpha_2 - 1}(1 - x_1 - x_2)^{\alpha_3 - 1}$, for $\boldsymbol{\alpha} = (1, 2, 3)$, $(10, 20, 30)$, $(100, 200, 300)$. Roughly give the location of the peak, and explain how the concentration of probability for the Dirichlet distribution changes for these values of $\boldsymbol{\alpha}$.

d) Analytically determine the location of the maximum of $f(x_1, x_2, \ldots, x_k; \boldsymbol{\alpha})$ by finding the maximum of its logarithm. (In taking derivatives, you will need to let $x_k = 1 - x_1 - \cdots - x_{k-1}$ and treat the logarithm as a function of $x_1, x_2, \ldots, x_{k-1}$ alone.)

6. Suppose you collect a data set D_1 and using a prior $\mathcal{P}(p)$ you compute the posterior distribution $\mathcal{P}(p \mid D_1)$ according to equation (12.2). You then collect a second data set D_2, and using $\mathcal{P}(p \mid D_1)$ as a prior for it, you compute a second posterior distribution. Under the assumption that D_1 and D_2 are independent regardless of the value of p, show that this two-step analysis gives exactly the same posterior as would a single-step one to find $\mathcal{P}(p \mid D_1 \text{ and } D_2)$.

Chapter 13

Gene trees and species trees

After twelve chapters discussing trees, models, and methods of inference, its time (or past time) to bring up the issue of exactly what the trees we are inferring represent.

When DNA or other sequences are collected, they come from individuals of a taxon, and not the taxon as a whole. Moreover, the sequence is not the entire genome of the individual, but more typically the sequence of one gene. Thus the trees we find should represent the evolutionary relationships of these individual genes or loci, and are better called *gene trees*. For reasons we will discuss in more detail in this chapter, these trees need not represent the evolutionary relationships of the full taxa, or even of the full individual. It is possible that they represent the relationships of the taxa on a *species tree*, and under some circumstances it is even likely. However, it is also quite likely that the gene trees and species trees will be in conflict.

The chapter develops the framework for modeling a primary source of conflict between gene trees and species trees, *incomplete lineage sorting*. While the distinction between gene trees and species trees has been understood from the early days of phylogenetics, only recently have serious attempts been undertaken to develop statistical tools to deal with it directly. For many years, inferred gene trees were simply accepted as likely proxies for species trees, with more careful scientists acknowledging this distinction. As it has become cheaper and easier to sequence many genes in a collection of taxa, it has become less easy to ignore the discordance of gene trees among each other.

The model used to capture incomplete lineage sorting is the *multispecies coalescent*. It modifies Kingman's basic coalescent model of population genetics so that several populations are linked to form a tree. Although this modeling framework is now fairly well established, it is not yet clear what methods of species tree inference inspired by it will ultimately prove most useful. Thus any enthusiasm or criticism we offer of particular approaches should be taken lightly; more progress can be expected in the next few years.

13.1 Gene Lineages in Populations

The primary reason we should not expect gene trees to always match species trees is shown in Figure 13.1. Here a tree with wide pipe-like branches represents the history of the species in their descent from a common ancestor, where the width of a branch is meant to convey that a species is actually a population of individuals of some size. The thin trees within it represent a gene tree, of the sort we might construct by standard phylogenetic methods, that relates some particular sample of sequences collected from individuals in those populations. Because of the many individuals present in the species at any time (represented by the width of the species tree branches), it is possible that several gene lineages may exist within a branch, without merging together. This then makes it possible for the lineages to eventually merge in some earlier population on the species tree in a way that gives a gene tree topology that differs from the species tree topology. Multiple gene lineages persisting in this way, not merging within a single branch of the species tree, is referred to as *incomplete lineage sorting*.

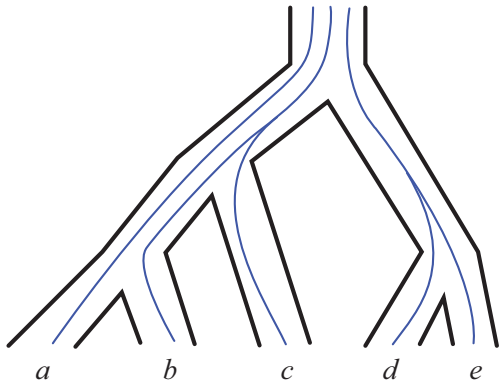

Figure 13.1: Gene trees may differ from species trees, since species trees are built from populations, and multiple gene lineages may persist through a species tree population (edge) and then merge with other lineages in a way that conflicts with the species tree topology. Here the species tree has topology $(((a, b), c), (d, e))$, while the gene tree might have topology $((A, (B, C)), (D, E))$, or several others, depending on how the lineages coalesce "above the root" of the species tree.

To model this phenomenon, we first step back from considering a full species tree, and instead consider the simpler situation of gene lineages in a single population.

The *Wright-Fisher* model imagines that there are some number N of individuals in the population at all times, with time tracked in discrete steps, corresponding to generations. In case of a haploid or diploid organism, there are thus either N or $2N$ copies of a specific gene present at each time step. We depict these as in Figure 13.2, with each row representing a generation, and

each dot representing a gene.

To create gene lineages, we begin at generation 0 (the present) and imagine each gene picks a parent gene uniformly at random from the previous generation. Thus from a gene in the current population at the bottom of the figure we can draw a random lineage, back one generation at a time, through the ancestral generations. (There are some obvious idealizations in this model: By picking parents at random, we assume the population is panmictic, we ignore sex and the grouping of two genes in one diploid organism, and we assume neutrality under selection.) An example of a simulation from this process is shown in Figure 13.2.

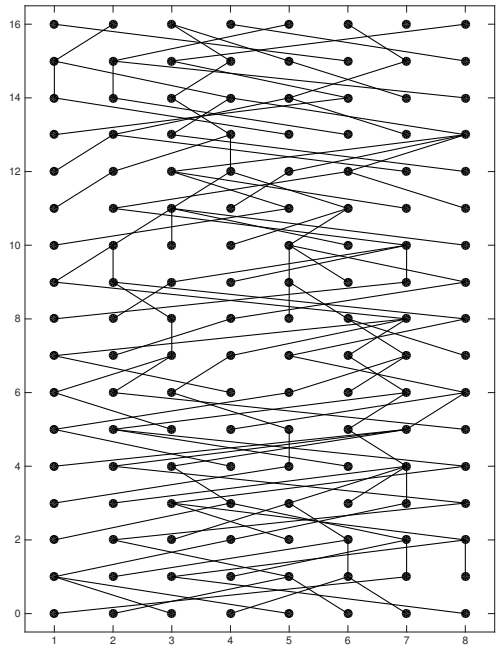

Figure 13.2: A simulation from the Wright-Fisher model, with 8 genes. A gene lineage is produced by a current gene (at bottom) choosing a random parent in the previous generation, which then chooses its parent, etc.

Figure 13.2 is too much of a tangle to interpret easily. However, by appropriate sorting of genes in each generation to prevent lineages from crossing, and suppressing lineages with no descendants in generation 0, we obtain the more understandable Figure 13.3. Working backwards in time, we see that gene lineages merge, or *coalesce* with some regularity. At each generation, there may be coalescent events which reduce the number of lineages coming from the extant genes. After moving enough generations into the past, all the lineages from the extant genes will have merged into a single one.

Viewing Figure 13.3 in the other direction, from the past to the present, we see many genes will have no progeny in the present. Indeed, if going backwards in time all lineages from the present have merged into a single one, then all genes not on that lineage will have no current progeny.

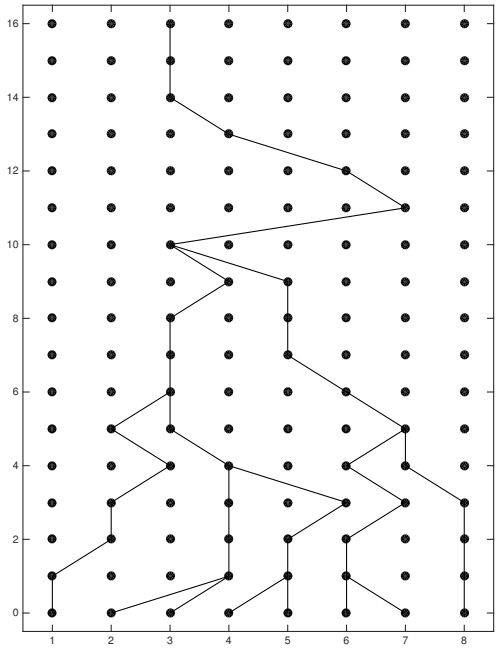

Figure 13.3: A Wright-Fisher simulation, after sorting of genes to untangle lineages, and retaining only lineages with extant descendants.

It is relatively easy to compute a few probabilities associated to this model. In the haploid case with N genes per generation, the probability that any two will choose the same parent, and thus have their lineages coalesce immediately, is $1/N$. This is because no matter what parent the first gene chooses, the second must choose the same one of the N possibilities to produce an immediate coalescence.

The probability that two lineages do not coalesce in the parental generation is therefore $1 - 1/N$. The same reasoning as before then gives the probability they will coalesce in the previous generation is $1/N$. Extending this reasoning shows that, using $C_2 = n$ to mean the event that two specific extant lineages coalesce n generations before the present,

$$\begin{aligned}
\mathcal{P}(C_2 = 1) &= \frac{1}{N} \\
\mathcal{P}(C_2 = 2) &= \left(1 - \frac{1}{N}\right) \frac{1}{N} \\
\mathcal{P}(C_2 = 3) &= \left(1 - \frac{1}{N}\right)^2 \frac{1}{N} \\
&\vdots \\
\mathcal{P}(C_2 = n) &= \left(1 - \frac{1}{N}\right)^{n-1} \frac{1}{N}
\end{aligned}$$

The probability of coalescence by generation n is thus

$$\mathcal{P}(C_2 \leq n) = \sum_{i=1}^{n} \mathcal{P}(C_2 = i) = \frac{1}{N} \sum_{i=1}^{n} \left(1 - \frac{1}{N}\right)^{i-1} = 1 - \left(1 - \frac{1}{N}\right)^{n} \quad (13.1)$$

As the number of generations n grows to infinity, we see the probability of coalescence approaches 1.

We can also compute the expected number of generations to coalescence of two specific lineages:

$$\sum_{n=1}^{\infty} n\mathcal{P}(C_2 = n) = \frac{1}{N} \sum_{n=1}^{\infty} n \left(1 - \frac{1}{N}\right)^{n-1} = N \quad (13.2)$$

This is plausible, as the larger the population size, the longer it should be before coalescence occurs, on average.

If we are interested in more than 2 lineages coalescing, things become more complicated. For instance for 3 lineages, there are likely to be two coalescent events needed for 3 lineages to merge down to 1 (though with low probability all three merge at once). We would have to consider the two generations in which these events occurred, and the expected final coalescence time would involve a double summation. Although quantities such as this can be worked out exactly, the formulas become rather complicated.

13.2 The Coalescent Model

Kingman's coalescent model can be viewed as a continuous-time approximation of the Wright-Fisher model (and also of other discrete models of population genetics, including the Moran model.) Rather than fully develop it from the Wright-Fisher model, we will instead simply define it. At an informal level the connection between them should seem reasonable. (See [Wak09] for an excellent full treatment.)

The model describes the coalescence of lineages as we move backwards in time within a single population. Time will be denoted by u, from the present

with $u = 0$ into the past with $u > 0$, and is measured in *coalescent units*. We will relate coalescent units to more familiar quantities later, but for now they are simply some measure of time. The entire model is this:

> Given pairs of lineages at any fixed time, the rate at which pairs coalesce into single lineages is constant, and equal to 1. Simultaneous coalescence of more than two lineages does not occur. Coalescence of different pairs is independent, and identically distributed.

Two small points are in order here. First, by *rate* we mean something very similar to what was meant in the discussion of the GTR model of base substitution. A rate of a continuous-time probabilistic process determines, through some calculation, a probability at any time, and so we will be able to compute probabilities of coalescences from it. Second, to make this rate 1 we simply rescale time units in a way that is convenient, just as we were able to freely rescale time with the GTR. As we will see, the resulting rescaled time measured in coalescent units need not be proportional to real time, except perhaps for very short periods.

Consider two lineages which are distinct at time 0, and $u > 0$, let $h(u)$ denote the probability that the two lineages are distinct at time u (that is, the two did not coalesce between time 0 and time u). Then the model tells us

$$\frac{d}{du}h(u) = -1 \cdot h(u),$$

where the negative sign is due to the fact that $h(u)$ should decrease.[1] Since we additionally know $h(0) = 1$, we find

$$h(u) = e^{-u}.$$

Thus if $\mathcal{P}(u)$ denotes the probability the two lineages did coalesce between time u_0 and u, we have

$$\mathcal{P}(u) = 1 - e^{-u}.$$

This is the analogue of equation (13.1) of the Wright-Fisher model, and graphing the two formulas shows they display quite similar behavior.

We can also compute the expected time to coalescence of two lineages, to obtain an analog of equation (13.2). Since the probability of coalescence in a short time interval is $\mathcal{P}(u + \Delta u) - \mathcal{P}(u) \approx \mathcal{P}'(u)\Delta u$, the expected time is

$$\int_0^\infty u\mathcal{P}'(u)du = \int_0^\infty ue^{-u}\,du = 1. \tag{13.3}$$

That this expected time to coalescence does not depend on the size of the population should seem surprising. However, as defined here the coalescent

[1] More formally, we are assuming coalescent events occur as a Poisson process, a standard probabalistic model for events that occur rarely, but with equal chance in any small interval of a fixed size.

model ignores the population size — it is never even referred to in the definition of the model. We have simply *defined* our time scale so the rate of coalescence is one

In fact the population size does matter, but it is taken care of by the definition of coalescent units. To see why this is reasonable, return to the Wright-Fisher model. Imagine that the population size changed as we move backwards in time, forming a *bottleneck*, as in Figure 13.4. If the large population below the

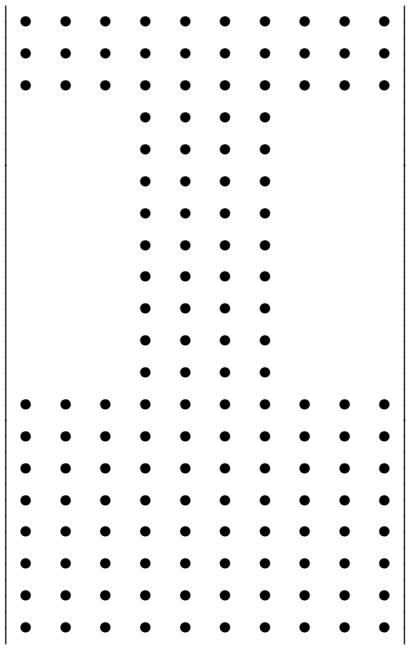

Figure 13.4: A bottleneck in the Wright-Fisher model, with population size $N_1 = 10$ except for $m_2 = 9$ generations with population $N_1 = 4$. The bottleneck causes faster coalescence of lineages, producing similar behavior to a longer time span with no bottleneck.

bottleneck is N_1, coalescence of pairs of lineages occurs with probability $1/N_1$ in each generation. When lineages enter the bottleneck, the same formula applies, but with a smaller population size N_2, the probability is now larger, $1/N_2$. Thus coalescence becomes more likely to occur. This means that if we had no access to the generational time scale, and could only query whether coalescence had occurred, the bottleneck of a relatively small number of generations of size N_2 would be indistinguishable from a larger number of generations where the population had remained constant at size N_1.

Reasoning roughly with the Wright-Fisher model, since the expected number of generations until two lineages coalesce is equal to the population size, N_2 generations in the small population has the same impact on coalescence as N_1 generations in the large one. If we introduce a new time scale for each

population, where we use

$$\Delta u = \frac{\Delta t}{N_i}$$

in a population of size N_i, where Δt is a number of generations, then N_1 generations in the large population and N_2 generations in the small population both yield $\Delta u = 1$ Thus scaling time inversely by population size enables us to treat the coalescence of lineages as proceeding at a constant rate.

For the discrete Wright-Fisher model, this change of time scales only approximately creates a uniform rate; after all, we do not have fractional generation time. However, in the coalescent model this becomes the definition of a coalescent unit. Since the coalescent is a continuous-time model, we can define units in term of 'infinitesimal' increments:

$$du = \frac{1}{N(t)} dt. \tag{13.4}$$

If the population size $N(t) = N$ is constant, then equation (13.4) integrates to give

$$u = \frac{t}{N}.$$

With this assumption, we can thus convert the expected coalescent time of 1 coalescent unit in equation (13.3) to N generations, which is exactly in accord with the Wright-Fisher result. However, equation (13.4) is more general, and allows changing population sizes to lead to non-linear relationships between u and t.

Since coalescent units are used as the time scale in formulating the model, the population size is doesn't explicitly appear in any calculations. However, the population does have an effect whenever we relate coalescent models to true time. A large population at some time means the coalescent clock 'runs slow' with respect to true time, so it takes more true time for coalescent events to occur. A small population means the coalescent clock 'runs fast' with respect to true time, so coalescent events occur more rapidly. More generally, as population size changes, the coalescent clock may be constantly changing its speed with respect to true time.

There is, of course, a price to pay for this relationship. It will be impossible to separate out the individual contributions of time and population size from their combined effect, unless we are willing to make some strong assumptions. While we might wish this was just an artifact of this model that we could do away with in some way, by thinking about the Wright-Fisher model it should become clear it would be a feature of any reasonable model we might formulate.

To demonstrate another calculation with the standard coalescent model, consider the expected time to coalescence of n lineages down to one. For all n lineages to coalesce, first 2 must coalesce so only $n-1$ lineages remain. Then 2 of these must coalesce so only $n-2$ remain, and so on, until the last 2 coalesce.

When all n lineages are present at time $u_0 = 0$, there are many pairs that might coalesce. It is thus reasonable that the first coalescent event will occur

sooner than if only two lineages were present. For a more precise calculation of the expected time of coalescence from n to $n-1$ lineages, recall that the coalescence of different pairs is i.i.d., so the overall rate of coalescence is increased by a factor of the number of pairs, $\binom{n}{2} = n(n-1)/2$. Thus with $k(u)$ being the probability that k lineages remain distinct at time $u > 0$, we have

$$\frac{d}{du}k(u) = -\frac{n(n-1)}{2}k(u),$$

with $k(0) = 1$. Thus

$$k(u) = e^{-\left(\frac{n(n-1)}{2}\right)u}.$$

Proceeding similarly to the calculation of the expected coalescent time for 2 lineages, we find that the expected time for n lineages to coalesce to $n-1$ is (see Exercise 6)

$$\frac{n(n-1)}{2}\int_0^\infty u e^{-\left(\frac{n(n-1)}{2}\right)u}\,du = \frac{2}{n(n-1)}. \tag{13.5}$$

Thus while the expected time for 2 lineages to coalesce to 1 is 1 unit, the time for 3 to coalesce to 2 is only 1/3 unit, the time for 4 to coalesce to 3 is 1/6 unit, etc. Adding these, we obtain the expected time for n lineages to coalesce to 1 is

$$\begin{aligned}\sum_{i=2}^{n}\frac{2}{i(i-1)} &= 2\sum_{i=2}^{n}\left(\frac{1}{i-1}-\frac{1}{i}\right)\\ &= 2\left(\left(1-\frac{1}{2}\right)+\left(\frac{1}{2}-\frac{1}{3}\right)+\left(\frac{1}{3}-\frac{1}{4}\right)+\cdots+\left(\frac{1}{n-1}-\frac{1}{n}\right)\right)\\ &= 2\left(1-\frac{1}{n}\right)\end{aligned}$$

As n approaches ∞, this grows, but the limit of the expected time until all lineages coalesce is 2, which is only twice that of when $n = 2$.

These calculations indicate that in a typical coalescent tree formed by a large number of lineages coalescing we should expect to see a lot of coalescence near the leaves of the tree, and longer edge lengths near the root. Roughly half the tree will have only two lineages which coalesce at the root, one third will have 3 lineages, etc. These characteristics are depicted in the tree shown in Figure 13.5.

Note that for a sexually reproducing diploid organism, each individual has two copies of most genes. Both the Wright-Fisher model, and the standard coalescent model ignore the fact that 2 gene lineages reside in each individual, and that individuals have sexes. However, since these copies come from different parents, their lineages are distinct, and in a panmictic population should have histories that are independent of one another.

If two different unlinked genes are considered in such an organism, even if they are sampled from the same individuals, there should also be little relationship between the gene trees for the two. Since the number of ancestors

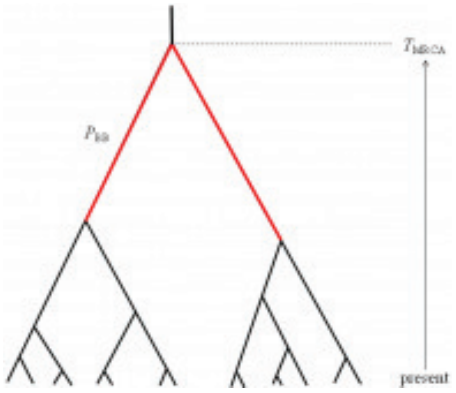

Figure 13.5: A typical tree produced by the coalescent process in a single population. Edge lengths are measured in coalescent units with the time during which there are k lineages being, on average, $2/((k)(k-1))$.

n generations in the past grows exponentially by the formula 2^n (provided, of course, the population is sufficiently large), going back even a few generations, the lineages of different genes are likely to pass through different individuals. Once this happens, under a panmictic assumption coalescence with different lineages should then be independent between the two genes. If genes are linked, then there lineages will not be independent and a more complicated model is needed to capture how lineages coalesce.

Finally the version of the coalescent presented here is not appropriate for all organisms. In some species of fish, for instance, the number of offspring of a successful breeder can be quite large. In that case, then a model must allow simultaneous coalescence of more than 2 lineages at a time. Such modifications result in a model called the Λ-coalescent.

13.3 Coalescent Gene Tree Probabilities

Suppose we sample the same gene in several individuals within a population and assume the coalescent model describes the probabilistic way their lineages coalesce. If the coalescent process is continued until all the lineages have become one, then a rooted gene tree is formed. Each possible gene tree can arise with some probability, which can be computed. In this section, we demonstrate how this can be done in the simplest situation — the gene samples are taken from a single population, which persists back in time 'forever'. (In essence, we assume a species tree with only a single taxon.) Gene tree probabilities can be computed for either topological gene trees or metric gene trees, so we give examples of both.

3-sample trees

First we consider sampling 3 extant lineages, which we will denote by A_1, A_2, A_3, at $u = 0$.

If we are only concerned with gene tree topologies, then we observe that the rooted gene tree which relates them is determined by the first pair of lineages to coalesce. Since the coalescence of pairs is i.i.d., this first coalescence involves A_1, A_2 or A_1, A_3, or A_2, A_3 with equal probability 1/3. Thus each of the 3 gene trees $((A_1, A_2), A_3)$, $((A_1, A_3), A_2)$, and $((A_2, A_3), A_1)$ arises with probability 1/3.

For a metric gene tree probability, note that distances will be given in coalescent units, and the tree must be ultrametric. Consider the gene tree $((A_1{:}u_1, A_2{:}u_1){:}u_2, A_3{:}u_3)$, where $u_3 = u_1 + u_2$. This is formed by

(a) 3 lineages and no coalescent events for a time interval of length u_1,

(b) a coalescent event of 3 lineages to 2 at time u_1, with the specific lineages A_1, A_2 coalescing,

(c) 2 lineages and no coalescent event for a time interval of length u_2,

(d) a coalescent event of 2 lineages to 1 at time $u_3 = u_2 + u_1$.

Now from previous calculations (a) has probability e^{-3u_1}, and (c) has probability e^{-u_2}. Both (b) and (d) require probabilities of coalescence at an instant, which is simply the rate of coalescence times du. Thus (d) has probability $du_3 = du_2$, while (b) involves both the probability of a coalescence, $3du_1$, and an additional factor of 1/3 that the lineages involved in the coalescence are A_1, A_2. The total probability is thus

$$\mathcal{P}(u_1, u_2) = e^{-(3u_1+u_2)} du_1 du_2,$$

so the probability density function is

$$f(u_1, u_2) = e^{-(3u_1+u_2)}. \tag{13.6}$$

When this is integrated over $0 \le u_1, u_2 \le \infty$, it gives a different computation of the probability 1/3 of the topological tree $((A_1, A_2), A_3)$. (See Exercise 7.)

Larger trees

If 4 extant genes A_1, A_2, A_3, A_4 are sampled at $u = 0$, the computing probabilities of topological trees is a little more complicated. A caterpillar gene tree like $(((A_1, A_2), A_3), A_4)$ can only be formed by a specific sequence of coalescent events. That the first lineages to merge are A_1, A_2 has probability $1/\binom{4}{2} = 1/6$. Once there are only 3 lineages, that the A_1A_2 lineage merges with the A_3 lineage next has probability $1/\binom{3}{2} = 1/3$. Then that the $A_1A_1A_3$ lineage merges with A_4 has probability 1. Thus the probability of this, or any of the other caterpillar gene trees is $(1/6)(1/3) = 1/18$.

A balanced tree like $((A_1, A_2), (A_3, A_4))$ can be formed by either of two sequences of coalescent events: A_1A_2 may merge first, followed by A_3A_4, or *vice versa*. Each of these has probability 1/18 since they are determined by a specific sequence of coalescent events. Thus the total probability of this, or any other balanced gene tree is $2(1/18) = 1/9$.

This last calculation shows a significant feature of the coalescent model in a single population. Topological gene trees that show more 'balance' tend to have higher probability than those that are less balanced, because they can be achieved by more distinct orderings of coalescent events. In fact, it is not hard to generalize the calculation of topological gene tree probabilities from 4-samples to more. Define a *ranked gene tree* as a rooted binary leaf-labelled topological tree with an ordering to the internal nodes (from the leaves to the root) such that the ranking of any node is greater than all its descendants. Then under the coalescent all ranked gene trees are equally probable. Since there are

$$R(n) = \prod_{k=2}^{n} \binom{k}{2} = \frac{n!(n-1)!}{2^{n-1}} \tag{13.7}$$

ranked gene trees (see Exercise 10), the probability of any gene tree is simply the number of rankings it may be given divided by $R(n)$.[2]

For a metric gene tree, the edge lengths determine an ordering to the coalescent events. For instance, the gene tree $((A_1{:}u_1, A_2{:}u_1){:}u_3, (A_3{:}u_2, A_4{:}u_2){:}u_4)$ is formed by coalescent events occurring at times u_1, u_2, and $u_1 + u_3 = u_2 + u_4$. If $u_1 < u_2$, then the first cherry formed was (A_1, A_2), while if $u_2 < u_1$ it was (A_3, A_4). For either case, computing the probability is very similar to the 3-sample case above (see Exercise 12). Unlike the topological tree case, there is no extra care that needs to be taken, since only oner ranking can arise for a metric tree.

13.4 The Multispecies Coalescent Model

The multispecies coalescent extends the basic coalescent model of the last section to a species tree of populations.

In terms of true time, we can picture the species tree as an ultrametric tree, with each branch represented by a pipe as in Figure 13.1. However, since we will be measuring time in coalescent units, which are only related to true time through inverse scaling by the population size, the species tree need not be ultrametric in these units. Moreover, using coalescent units means we have in some sense standardized the widths of the pipes to all be the same, so that standard Newick notation can be used to specify a species tree.

[2]The probabilities obtained for rooted topological trees here is the same as is produced by the Yule model, a model of branching that proceeds from the root toward the leaves, and is often taken as the simplest probabilistic speciation model. One could argue that it is the most natural distribution of rooted trees in biological contexts, and is perhaps a better choice of a "non-informative" prior for a Bayesian tree inference.

The multispecies coalescent model is simply the standard coalescent model on each edge of the species tree 'glued' together. But now, since species tree edges have finite length, we may have several gene lineages in a population an edge represents that fail to coalesce within that edge. When they reach the ancestral end of the edge, one or more additional lineages will enter from another branch of the species tree. If we are computing probabilities of metric gene trees, then we specify precisely where each coalescent event occurs, and though the bookkeeping can be rather cumbersome, we merely have to combine various probabilities of lineages coalescing, or not coalescing, in a single edge of a specific length, in ways very similar to the single population coalescent. For topological gene trees, the work is similar, but we first need to compute probabilities than if k lineages 'enter' an edge of length x, that $0 \leq \ell < k$ coalescent events will occur on that edge. (A future version of these notes will derive that....)

3-sample trees

We begin with a simple, yet very important example, of a 3-taxon species tree $((a{:}y, b{:}z){:}x, c{:}w)$, where we sample one gene from each taxon. We will denote the sampled genes by A, B, C, using uppercase letters corresponding to the taxon names. Before we consider a specific gene tree, note that y, z, w will have no effect on the probability of observing any *topological* gene tree. This is because under the one-sample-per-taxon scheme, as shown in Figure 13.6, there will be only one lineage in each of the pendant species tree branches, and so no opportunity for coalescence in these populations.

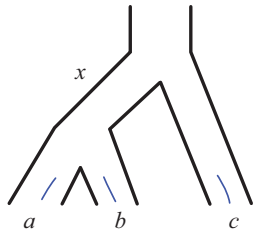

Figure 13.6: With one gene sampled per taxon, no coalescence can occur in pendant populations on the species tree, so the lengths of those edges are irrelevant to the probability of observing any topological gene tree. For the species tree $((a{:}y, b{:}z){:}x, c{:}w)$, only x will appear in probability formulas.

There are 3 possible topological gene trees: $((A, B), C)$, $((A, C), B)$, and $((B, C), A)$. To compute the probabilities of observing them, it is easiest to begin with $((A, C), B)$. The only way this gene tree can be formed is if the A and B lineages enter the population with length x and then reach the root of the tree before merging. That results in 3 lineages being present at the root, and then the A and C lineages must coalesce in the population ancestral to the root.

For the previous section we know the probability of two lineages not coalescing in x coalescent units is e^{-x}. The probability that the correct 2 lineages of the 3 then coalesce first is 1/3. Thus

$$\mathcal{P}(\,((A,C),B)) = \frac{1}{3}e^{-x}.$$

The same reasoning shows

$$\mathcal{P}(\,((B,C),A)) = \frac{1}{3}e^{-x}.$$

Since the probabilities of the 3 topological gene trees must add to 1, this also means

$$\mathcal{P}(\,((A,B),C)) = 1 - \frac{2}{3}e^{-x}.$$

Notice that if $x = 0$, so the species tree has a polytomy at the root, all of these become 1/3, as is reasonable. If $x = \infty$, on the other hand, the A and B lineages must coalesce in the infinitely long branch, and so we find $\mathcal{P}(\,((A,B),C)) = 1$, with the other gene trees having probability 0. But for $0 < x < \infty$, we have

$$0 < \mathcal{P}(\,((A,C),B)) = \mathcal{P}(\,((B,C),A)) < \mathcal{P}(\,((A,B),C)).$$

Thus in this case the most probable gene tree has the same topology as the species tree, while the two discordant gene trees have smaller probabilities.

This leads to a simple method of species tree inference in the 3-taxon, 1-sample-per-taxon case: after inferring many gene trees by standard phylogenetic methods, simply tally the number of gene trees with each topology. Then accept the most frequent one as the species tree topology. But we can actually infer even more. After estimating $\frac{2}{3}e^{-x}$ as the proportion of gene trees that are discordant with the inferred species tree, we can solve for an estimate of x. Thus the gene tree distribution not only contains information on the topology of the true species tree, but also on the lengths of its edges.

The simplicity of the 3-taxon case, though, is misleading. For n-taxa, the most probable gene tree need *not* match the topology of the species tree. This fact means the natural intuition that whatever gene tree is inferred the most often should be reflect the specie tree can be misleading.

If we instead needed to compute the probability of a metric gene tree (with edge lengths in coalescent units) we proceed more similarly to the example discussed in the previous section. But now the edge lengths in the gene tree determine in which populations on the species tree the various coalescent events occurred. For example, for the species tree $((a{:}y, b{:}z){:}x, c{:}w)$to calculate the gene tree probability $\mathcal{P}(((A{:}u_1, B{:}u_2){:}u_3, C{:}u_4))$, we must separate the cases where A and B coalesce in the species tree branch of length x, and when they coalesce in the population ancestral to the root of the species tree. Moreover, for the gene tree to have non-zero probability, we must have

$u_1 > y,\ u_2 > z,\ u_4 > w,\ u_1 - y = u_2 - z,\ u_1 + u_3 > y + x,\ u_1 + u_3 - y - x = u_4 - w.$

Assuming these conditions are met

$$\mathcal{P}(((A{:}u_1, B{:}u_2){:}u_3, C, {:}u_4)) = \mathcal{P}(u_1, u_3) = \begin{cases} e^{-(u_1-y)-(u_1+u_3-x-y)}, & \text{if } u_1 < y + x, \\ e^{-x-3(u_1-x-y)-u_3}, & \text{if } u_1 > y + x. \end{cases} \quad (13.8)$$

The formulas depend only on u_1 and u_3 since those values determine u_2 and u_4 by the restrictions above.

Larger trees

To illustrate how probabilities are computed for larger trees, we give only one (partial) example. For the species tree is $(((a{:}u_1, b{:}u_2){:}u_3, c{:}u_4){:}u_5, d{:}u_6)$, consider the topological gene tree $(((A, B), C), D)$ which has the same topology as the species tree. Since there is only one ranking of the coalescent events forming this gene tree, there are relatively few ways it could form. They are:

1. (A, B) coalesces in the branch of length u_3, $((A, B), C)$ coalesce in the branch of length u_5, and $(((A, B), C), D)$ coalesces in the population ancestral to the root of the species tree.
2. (A, B) coalesces in the branch of length u_3, and then both $((A, B), C)$ and $(((A, B), C), D)$ coalesce in the population ancestral to the root of the species tree.
3. (A, B) and $((A, B), C)$ coalesce in the branch of length u_5, and $(((A, B), C), D)$ coalesce in the population ancestral to the root of the species tree.
4. (A, B) coalesces in the branch of length u_5, and then both $((A, B), C)$ and $(((A, B), C), D)$ coalesce in the population ancestral to the root of the species tree.
5. All coalescent events occur in the population ancestral to the root of the species tree.

Since for scenario 1 there are never more than 2 lineages in any population, and we have seen that the probability that 2 lineages coalesce within an edge of length x is $1 - e^{-x}$, the probability is easy to compute as

$$(1 - e^{-u_3})(1 - e^{-u_5}).$$

Note that the coalescent event in the population ancestral to the species tree root is sure to occur, and so contributes a factor of 1 to the probability.

The probability for scenario 2 is not much harder. It is

$$(1 - e^{-u_3})(e^{-u_5})(1/3).$$

the first two factors are the probability there is a coalescence in the edge of length u_3, and that there is not one in the edge of length u_5. Since 3 lineages

are present at the species tree root, the factor of 1/3 is the probability that the correct pair of the three possible ones coalesces next.

The remaining scenario probabilities can be worked out similarly (see Exercise 14), and the sum of the five of them gives the probability of the topological gene tree.

While not all scenarios leading to a topological gene tree are equally probable, the more scenarios there are, the larger the total probability can be. In particular, if a gene tree has several rankings, then that produces more scenarios, and tends to result in a higher probability for such gene trees than one might expect. In fact it is possible that the gene tree with the greatest probability has a *different* topology than the species tree, or that several gene tree topologies are more probable than the one matching the species tree. Although this phenomenon of *anomalous gene trees* has been studied extensively by Degnan and Rosenberg, here we give only a simple argument to illustrate it.

First consider a 'star' species tree relating 4 taxa a, b, c, d, with 4 pendant edges emerging from the root. Probabilities of topological gene trees are then easy to see, since all coalescent events occur ancestral to the species tree root. That means our earlier analysis of four lineages samples from a single population applies, so that each of the 12 possible caterpillar trees has probability 1/18, and the 3 balanced trees have probability 2/18 due to their two rankings. Now if we instead consider any binary 4-taxon species tree, and make all internal edges very short, the probabilities of the gene trees will not be very different (since the probabilities are continuous functions of the internal edge lengths, and as these approach 0 we move to the star tree). Since there is such a gap between the probabilities of the caterpillar and balanced gene trees in the star species tree case, we will still have a balanced tree as the most probable for the binary species tree. In particular, for a 4-taxon caterpillar species tree with sufficiently short internal edges, the most probable topological gene tree will still be balanced, and thus not match the species tree.

A more detailed analysis, in which exact probabilities of topological gene trees as functions of species tree edge lengths are computed, can show exactly what edge lengths allow for anomalous gene trees, and exactly which gene tree is most probable.

The examples above indicate how the multispecies coalescent model can be used to understand gene tree distributions. But as with most phylogenetic computations, when there are more than a few taxa calculations are best handled by software. But before turning to the use we might make of these probabilities, we summarize by listing some key points concerning the model:

- Given a metric rooted species tree, with edge lengths in coalescent units, it is possible (but painful) to compute the probabilities of either metric or topological gene trees.

- The distribution of gene trees, whether metric or topological, that arises from the multispecies coalescent model carries information about the species

tree topology and edge lengths. Thus using many gene trees to estimate this distribution should allow us to infer the species tree. The discordance of gene trees that might otherwise be viewed as problematic is actually a source of meaningful information.

- The probabilities of metric gene trees are for gene trees with edge lengths *in coalescent units*. If we wish to relate these to gene trees inferred by standard phylogenetic methods, whose edge lengths are typically measured in amount of substitutions, we must make further assumptions.

- Species trees need not be ultrametric in coalescent units. Indeed, if the population sizes vary over the species tree, then typically it will not be ultrametric in these units. By working in these units we can allow very general changes in population sizes over time. If we wish to relate coalescent units to true time, we must make further assumptions, such as that the population size is constant over the entire tree, or that we have a specific description of the way the population size changes.

- The most probable topological gene tree need *not* match the species tree topology. If there are more than 3 taxa, and internal edges of the species tree might be short, picking the most frequent gene tree topology inferred from many genes is not a reliable way of inferring the species tree. Hoping that the species tree topology will be 'obvious' to the naked eye from considering many genes is simply naive.

13.5 Inferring Species Trees

A variety of ways of inferring species trees, either from a collection of gene trees or from a collection of sequence alignments, have been proposed and implemented in software in the past few years. However, these data analysis methods are still undergoing development, and using them is not yet routine. The scientific community does not yet have enough experience with them to understand fully their strengths and weaknesses, or under what circumstances they are likely to perform well or poorly. The collection [KK10] offers a number of articles on both theory and practice with some of these methods.

There are several different ways one could classify currently proposed methods. Some are based explicitly on the coalescent model, and have been proved to be statistically consistent. Others ignore the coalescent, and use some sort of heuristic approach to 'averaging' over different genes. One could also group them by whether they use metric gene trees, or only their topologies. Alternately, they could be classified by whether they attempt maximum likelihood, Bayesian analyses, parsimony, or other combinatorial approach to determining a species tree. Some use a combined model of sequence evolution and the multispecies coalescent, as opposed to performing inference as a two-step process by first inferring gene trees and then using these as 'data' to infer a species tree. Since any such classification scheme would emphasize one feature as being more

important than another, we will instead simply list methods in no particular order.

We describe some of these methods only in the case where we sample one gene lineage from each taxon, so that if we are studying n taxa, each gene tree will have n leaves. There are extensions of most of these methods to cases of multiple samples of each gene lineage per taxon, including allowing different number of samples for each gene. Provided the sampling in each taxon is done well, data collected with multiple samples has potential for improved inference (at least in the case where pendant species tree branches are relatively short).

Concatenation of sequences.

Given sequences for a number of different genes, an early approach that remains in widespread use is to concatenate sequences from all the genes, and treat the combined sequence as if it was one giant gene in order to perform a standard ML or Bayesian phylogenetic analysis. Software returns an inferred tree (or distribution of trees), which is then proclaimed to be the inferred species tree. Bootstrap support is often high, since the sequences are quite long, so this resampling technique showed little variability.

Though this method still has its advocates, there is no theoretical reason one should expect it to be reliable if incomplete lineage sorting is an important factor in why gene trees vary. By concatenating genes and analyzing them as one long sequence, even if a partitioned analysis allows for different numerical parameters for each gene, one is forcing the analysis to use a single tree topology. But the fact that there are likely to be different gene tree topologies is exactly what we are hoping to overcome! In other words, we are using a model that we know is seriously incorrect as the basis of our analysis. Moreover, we have in no way used our understanding of how incomplete lineage sorting occurs to inform the way we analyze the data. Finally, it has been proved by Steel and Roch that concatenation is not a statistically consistent method of inference of a species tree under the multispecies coalescent. However, if one is convinced ahead of time that all edges in the (unknown) species tree are long, then incomplete lineage sorting may be negligible, and this approach is better justified.

Maximum Likelihood and Bayesian analyses

At the other extreme from concatenation is to use both a model of gene tree formation by the multispecies coalescent together with a standard phylogenetic model of sequence evolution of the sort discussed earlier in these notes. Then either Maximum Likelihood or a Bayesian framework can be adopted for a well-justified statistical framework, using the models in succession , or combined.

We first sketch the ML approach used in the software STEM (Kubatko, *et al.* (2009)). Given many gene sequences, one first infers metric gene trees by using ML and standard phylogenetic models. One then must convert the branch lengths on the gene trees from units of total substitutions to coalescent units. Assuming a common mutation rate of μ on all edges of all gene trees, one can

divide by μ to obtain true times. Then assuming a constant population size N for all populations on the species tree, one can divide by $2N$ (for diploid organisms) to convert to coalescent units. Note however that these assumptions imply all gene trees should be ultrametric, so they must either be 'adjusted' somehow, or inferred with that as a constraint. With this done, it is now straightforward (though a substantial amount of work) to implement a standard ML analysis under the coalescent model, with a new tree search for the species tree. There are some generalizations one can make, for instance allowing independent rescalings of the various gene trees to account for the fact that they may evolve at different rates.

Bayesian approaches are similar, and have been implemented in software by two different groups, in Mr. Bayes/BEST (Hulsenbeck, *et al.* Liu, *et al.*), and in *BEAST (Heled, *et al.*) It is a bit easier to program an analysis using a combined model of both the coalsecent for formation of gene trees and a substitution model for evolution of sequences. Letting D_i denote DNA sequence data for the ith of k genes, G a metric rooted gene tree, and S the species tree, the posterior is computed in the usual way from a prior on metric species trees and the likelihood function

$$\mathcal{P}(D|S) = \prod_{i=1}^{k} \sum_{G} \mathcal{P}(D_i|G)\mathcal{P}(G|S). \tag{13.9}$$

Just as for ML, the issue of converting time scales on gene trees from substitutions to coalescent units arises, and is dealt with similarly.

Note that there is, as yet, no ML software that implements a combined model of the coalescent and the substitution process, in which a likelihood function for the species tree given the sequence data are used. STEM uses a two-step inference procedure instead, which means that while there is some statistical error in the inferred gene trees, they are treated as if they were 'data' in the coalescent model analysis. The Bayesian analyses, though, do not have this issue.

Though in principal using Maximum Likelihood and Bayesian approaches to infer species trees is attractive, current implementations have their shortcomings. A worrisome issue is the conversion of time scales on gene trees, which is done by making assumptions over all gene trees of a fixed mutation rate (i.e., a molecular clock) and over the species tree of a fixed population size. If these are approximately valid they may not cause problems, but if they are violated strongly it is unclear what the impact is. There have been reports of Bayesian analyses not converging to a stable posterior for some data sets, though the reasons for this are not clear. Liu now warns that if the gene trees appear to be strongly non-ultrametric, BEST may not perform well.

In addition, the amount of computation for the Bayesian approaches is large enough, that there are practical limits on sizes of data sets, both in the number of taxa, and in the number of genes. While future generations of programs are likely to be more efficient, it is unlikely that will be sufficient to really increase limits substantially.

Pseudolikelihood

A pseudolikelihood method follows the same approach as ML, but replaces the true Likelihood function by something simpler. Liu *et al.* (2010) defined a pseudolikelihood function for the species tree given a collection of gene trees by considering all the topological rooted triples displayed on the gene trees. For instance, a gene tree $((a,b),(c,d))$ displays the four rooted triples $((a,b),c)$,$((a,b),d)$,$((c,d),a)$,$((c,d),b)$. After counting how often each rooted triple occurs on the gene trees, one computes the probability of each rooted triple given a species tree. Treating the rooted triples as independent, the pseudolikelihood is simply the product of these probabilities, each raised to the count. This is a considerably simpler function than the true likelihood function, so that software can run much faster. Of course, the rooted triples are *not* independent, so the standard guarantees that ML behaves well do not apply. However, simulation evidence is that this works well, and it can deal with much larger datasets than the true Likelihood approach. The software implementing this is MP-EST.

Minimizing deep coalescence.

This method is in spirit very similar to the use of parsimony for inferring a gene tree. It is based in a reasonable assumption that while incomplete lineage sorting can occur, we are unlikely to see extreme examples of it. Thus if we have a collection of gene trees (previously inferred from sequence data, and assumed correct), we should choose as the 'best' species tree the one which, if all the gene trees had arisen on it, would have coalescences of lineages as close to the most recent common ancestral species as possible.

Conceptually, the method can be performed as follows. We consider each possible species tree T, and for each of the given gene trees g we compute a score $s_g(T)$ that measures the minimal number of 'deep coalescences' that are necessary for g to have arisen on T. We add these scores for all gene trees, to obtain a score for T:

$$s(T) = \sum_g s_g(T).$$

We then choose the tree(s) T that has the minimal value $s(T)$.

The score $s_g(T)$ is defined as shown in Figure 13.7. First, consider any internal node v in the gene tree, and let X_v denote its leaf descendants. Then v represents a coalescent event that could only have occurred on the species tree above (temporally before) the most recent common ancestor of X_v on the species tree. We therefore assume it occurred in the population just above the MRCA, and locate it along that edge of the species tree. Doing this for every node of g gives us a map of the gene tree to the species tree, and allows us to talk about the number of gene tree lineages entering (at the child end) and leaving (at the parent end) every edge. If the gene tree had the same topology as the species tree, there would be two lineages entering each species tree edge, and 1 lineage leaving. For a highly discordant gene tree, these counts will typically

be larger. We define the number of *excess* lineages for an edge of the species tree as 1 less than the number leaving the edge. Then $s_g(T)$ is the sum over all edges of T of the number of excess lineages. Thus for a gene tree matching the species tree we have $s_g(T) = 0$, and discordant gene trees will have higher scores.

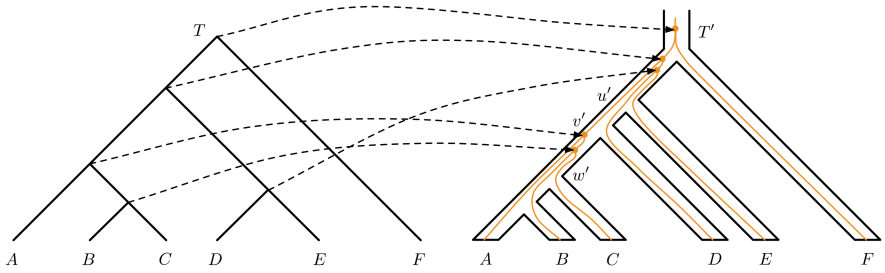

Figure 13.7: To compute the score $s_g(T)$ for a gene tree g on left and a species tree T on right, the nodes of g are first mapped to the lowest population edges on T on which they could have arisen. Then $s_g(T)$ is the sum over all edges of T of the number of 'excess' lineages in g exiting them at the ancestral end. (FIGURE from Than-Nakleh 2009)

Notice parsimony ideas have appeared twice in this method, once for assuming coalescences occur 'as soon as possible' on the species tree, and once for choosing the species tree with the lowest overall number of excess lineages.

Like parsimony for phylogenetic inference, this method is both reasonable and likely to work well under some circumstances. Moreover, Than and Nakleh have reformulated the optimization problem in ways that can be solved by techniques of either integer or dynamic programming, giving fast performance in practice. Unfortunately, and also like parsimony, it is known that minimizing deep coalescence is not a statistically consistent inference method for some species trees.

Finally, we note that this method uses only topological information on gene trees, and thus avoids attempting to relate time scales on the gene trees to those on the species tree. It also does not make any direct use of the coalescent model.

Consensus methods

If we have already inferred a collection of gene trees from sequence data, we can also simply combine them with a standard consensus method. While at first this might seem like just a combinatorial way to overcome the gene tree differences, and not one that has much to do with the coalescent, in fact this approach has some provably good properties under the coalescent model.

For a theoretical distribution of rooted gene trees, define the probability of a clade to be the sum of the probabilities of all gene trees displaying that clade.

The following theorem of Allman, Degnan, and Rhodes (2011) is the key to understanding how consensus methods behave for inferring species trees.

Theorem 19. Under the coalescent model on a binary species tree, any clade on a gene tree with probability greater than 1/3 is also a clade on the species tree.

The 1/3 cut-off in this theorem can not be reduced; for any lower number, examples can be constructed of clades with greater probabilities that are not on the species tree.

Now from a collection of gene trees one can estimate the probability of a clade by the proportion of the gene trees displaying it. If one builds a consensus tree from the clades with estimated probability greater than 1/3, then provided one has a large enough sample of gene trees, the only clades used will be ones on the species tree. As a result, the inferred tree may not be fully resolved, but any clades it shows will reflect species tree clades. Since the strict and majority-rule consensus trees will only use a subset of these clades, they may be less well resolved, but will also only reflect true species tree clades. While for an arbitrary collection of clades one cannot be sure those with frequency $\leq 1/2$ will be compatible, this result implies that if the trees come from the coalescent, all clades with frequency $> 1/3$ will be compatible, provided enough gene trees are given.

A more recent theorem of Allman, Ané, Rhodes and Warnow (2011, unpublished) establishes a similar result for splits on unrooted gene trees: Any gene tree split with probability greater than 1/3 is a split on the unrooted species tree. Thus constructing a consensus tree using gene tree splits of frequency $> 1/3$ will, for a large enough sample of gene trees, infer a tree that may not be fully resolved but whose splits will reflect ones on the species tree.

Greedy consensus, using either gene tree splits or clades, is however not statistically consistent, as has been shown by Degnan, Degiorgio, Bryant, and Rosenberg (2009). Accepting any splits or clades with frequency below 1/3 will not necessarily give the correct species tree, even with an infinite sample of gene trees.

Another approach, called R^* consensus (Degnan, *et al.* (2009), is justified by the result proved in section 13.4 that for any 3 taxa, the most probable gene tree matches the species tree. Given a collection of gene trees, one could consider every subset of 3 taxa, and the 3-taxon gene trees they induce. By counting the ocurances of each of the 3 possibilities, one can infer the 3-taxon tree induced on the species tree as the most frequent one. For 3-taxon trees, as we have seen, this is a consistent way of picking the species tree. Once one has obtained all these 3-taxon induced trees, one can then build the species tree displaying them all. In practice, one may find some of these are incompatible, so decisions must be made as to how that is to be handled. (We will not give details here.) Unlike the previous consensus methods mentioned, this method will, provided we have enough gene trees, result in a fully-resolved species tree. Moreover, it is statistically consistent.

A similar approach replaces rooted triples from rooted gene trees with quartets on unrooted gene trees. The most frequent gene tree quartet topology does consistently infer the species tree quartet topology, so one can then search for a species tree displaying most such quartets. With some elaborations, this is the approach taken in the ASTRAL software of the Warnow group.

STAR and NJ$_{st}$

There are also methods proposed by Liu and collaborators, that infer species trees not by the standard consensus approach, but by a process that in spirit extends them. These are both very fast, since they are built on distance methods of tree inference, and have been shown to have good performance in simulations,

The first of these, called STAR (Liu *et al.*, 2009) proceeds as follows to obtain a species tree from a collection of rooted topological gene trees. First, assign a length of 1 to all internal branches of all the gene trees. Then make pendant edges have whatever length is necessary so all leaves are distance n from the root, where n is the number of taxa. Now that the gene trees have all been made into ultrametric trees, compute a distance matrix for each, giving distances between the leaves. Next, average these gene tree distance matrices over all the gene trees (i.e., average each of the corresponding entries). From this average distance matrix, build a tree using your favorite distance method, such as Neighbor Joining. Finally discard the metric information on this tree and report it as the topology of the species tree.

Though at first this seems like a rather Rube Goldbergian way to infer a species tree, it is not as ridiculous as it may sound. Though its introduction was accompanied only by a proof of a 4-taxon special case, it turns out that this is a statistically consistent way to infer species trees assuming the multispecies coalescent. Moreover, simulations showed it can work reasonably well even when provided with only a small number of gene trees. A rigorous proof of consistency given by Allman, Degnan, and Rhodes (2013).

A similar proposal of Liu, *et al.* (2011) uses unrooted topological gene trees to infer an unrooted topological species tree. For this method, originally called NJ$_{st}$, all gene tree branches, internal and pendant, are assigned a length of 1, distance matrices for each are then calculated and averaged, and an unrooted tree is chosen to fit this average, for instance by Neighbor Joining. A proof of the statistical consistency of this has not yet been published is in the works. It has also been implemented efficiently in software called ASTRID (Vachaspati and Warnow, 2015), and shown in simulations to work well for very large data sets, with thousands of taxa and genes.

GLASS, STEAC

TO BE WRITTEN

Concordance factors

Although it is not based on the coalescent model, or any model of why gene trees might differ, a useful software tool for summarizing and understanding gene tree discordance in a Bayesian setting is BUCKy (Larget, *et al.* (2010)). It performs a technique called Bayesian concordance analysis (Ané, *et al.* (2007)) to take posterior distributions for many gene trees, and combine them into what can be interpreted as support for various clades. The lack of a model underlying this tool means that even if the coalescent process is not the source of the gene tree discordance, for instance, if horizontal gene transfer or hybridization occurs, one may still use it to gain some insights.

As a final comment, we note that most of these methods implicitly assume the genes being analyzed are unlinked. This is necessary so that each gene tree can be viewed as an independent trial of the coalescent process. If for instance, one wished to use mitochondrial genes in mammals, this assumption would be strongly violated due to maternal inheritance. For nuclear autosomal genes, one should also take some care not to use genes too closely located on a chromosome.

13.6 Exercises

1. Show the finite series in equation (13.1) has the stated value.

2. Show the series in equation (13.2) has the stated value. You will need to use the formula for the sum of the geometric series $\sum_{n=0}^{\infty} r^n$, and differentiate with respect to r.

3. The Wright-Fisher model behind the simulation in Figure 13.3 leads to an expected time to coalescence of any two lineages of 8 generations. Compute the average of the coalescent times for the $\binom{8}{2} = 28$ pairs of lineages in the simulation, and compare.

4. Show the integral in equation (13.3) has the give value.

5. Suppose a population is exponentially growing, so $N(t) = N_0 e^{-\alpha t}$ with $\alpha > 0$ where t is measured backwards from the present. Use equation (13.4) to give a formula relating coalescent units and true time.

6. Complete the derivation of the formula in equation (13.5).

7. By computing a double integral of the probability density in equation (13.6), recover that the probability of the topological gene tree $((A_1, A_2), A_3)$ is $1/3$.

8. Under the single population coalescent model, the probabilities of all 4-sample gene trees are computed in the text. How many such trees are there? How many of these are caterpillars? How many are balanced? Show the computed probabilities add up to 1.

9. Describe all possible orderings of coalescent events that could have led to each of the gene trees

   a) $((((A, B), C), D), E)$

   b) $(((A, B), C), (D, E))$

10. Explain the formula for the number of ranked gene trees in equation (13.7) by considering the formation of a tree by starting with n lineages and choosing pairs to coalesce according to the ranking.

11. If n genes $A_1, \ldots, A_n$ are sampled from a single population, under the coalescent model compute

    a) the probability that the gene tree relating them is the specific caterpillar $((\ldots((A_1, A_2), A_3), \ldots, A_{n-1}), A_n)$.

    b) the probability that the gene tree relating them is *any* caterpillar tree.

12. Suppose lineages A_1, A_2, A_3, A_4 are sampled in a single population. Under the coalescent model, the probability density for a metric gene tree relating them will depend only on the 3 times at which the coalescent events occurred.

    (a) Find the probability density for $((A_1{:}u_1, A_2{:}u_1){:}u_3, (A_3{:}u_2, A_4{:}u_2){:}u_4)$ when $u_1 < u_2$.

    (b) Find the probability density for the same tree when $u_2 < u_1$.

13. Check that the formulae in equation (13.8) are correct. If not, give correct ones, and inform the authors.

14. In Section 13.4, five scenarios were considered in the text in which the topological gene tree $(((A, B), C), D)$ is formed under the multispecies coalescent model on $(((a{:}u_1, b{:}u_2){:}u_3, c{:}u_4){:}u_5, d{:}u_6)$, and the probabilities were computed for two of them. Compute the probabilities of the three remaining ones, and add them to obtain the probability of $(((A, B), C), D)$.

    Note: Scenarios 3 and 4, which involves coalescence of 3 lineages to either 1 or 2 in a single branch of length u_5, are the most involved to compute. For instance, for scenario 3, let x be the time of the coalescent event, You will need to explain why the integral

    $$\int_0^{u_5} e^{-3x}(1 - e^{-(u_5 - x)})dx$$

    arises, and evaluate it.

Chapter 14

Notation

$\mathbb{R} =$ all real numbers
$\mathbb{R}^{\geq 0}$ = all positive numbers, and zero
$\mathbb{Z}^{\geq 0}$ = all integers = $\{\ldots, -3, -2, -1, 0, 1, 2, 3, \ldots\}$
$\mathbb{Z}^{\geq 0}$ = all positive integers, and zero = $\{0, 1, 2, 3, \ldots\}$
$|A|$ = the number of elements in the set A

Chapter 15

Selected Solutions

Chapter 2

8.

$$(2n-5)!! = 1 \cdot 3 \cdot 5 \cdots (2n-5) = \frac{1 \cdot 2 \cdot 3 \cdot 4 \cdots (2n-6) \cdot (2n-5)}{2 \cdot 4 \cdots (2n-6)}$$
$$= \frac{(2n-5)!}{(2^{n-3})1 \cdot 2 \cdots (n-3)} = \frac{(2n-5)!}{(2^{n-3})(n-3)!}$$

10. *Proof.* Let $T = (V, E)$ be a tree. For the sake of obtaining a contradiction, suppose for some distinct $v_1, v_2 \in V$ there is more than one path from v_1 to v_2.

    From all possible pairs of such vertices, and all possible pairs of such paths between them, choose paths which have the smallest sum of lengths. Denote these paths $u_1, u_2, \ldots, u_n$ and $w_1, w_2, \ldots, w_m$, with $v_1 = u_1 = w_1$ and $v_2 = u_2 = w_2$.

    Consider the sequence of adjacent vertices $v_1 = u_1, u_2, \ldots, u_n = v_2 = w_m, w_{m-1}, \ldots, w_1 = v_1$. This cannot be a cycle, since T is a tree. Thus it either has a repeated vertex other than v_1, or it has length 2.

    If the sequence had a repeated vertex, then since there is no repeated u_i and no repeated w_j, it must be that for some $i < n$,$j < m$ we have $u_i = w_j$. But then either $u_1, u_2, \ldots, u_i$ and $w_1, w_2, \ldots, w_j$, or $u_i, u_{i+1}, \ldots, u_n$ and $w_j, w_{j+1}, \ldots, w_m$ would be a pair of distinct paths between two vertices of shorter total length than our chosen ones. This would contradict the minimality of our chosen paths' length. Thus there can be no repeated vertices.

    Therefore the length of the sequence must be 2, which implies $n = m = 1$. But then the two chosen paths were the same. This is also a contradiction. Thus there cannot exist a pair of paths with the same endpoints. □

12. *Proof.* Let T be a metric tree with all positive edge lengths, and $v_1, v_2, v_3 \in V(T)$.

Suppose v_3 lies on the unique path between v_1 and v_2. Then this path has the form

$$v_1 = u_1, u_2, u_3, \ldots, u_i = v_3, u_{i+1}, \ldots, u_n = v_2.$$

This implies $u_1, u_2, \ldots, u_i$ is the unique path from v_1 to v_3, and $u_i, u_{i+1}, \ldots, u_n$ is the unique path from v_3 to v_2. Thus

$$\begin{aligned} d(v_1, v_2) &= \sum_{\substack{e \text{ on the path} \\ \text{from } v_1 \text{ to } v_2}} w(e) \\ &= \sum_{\substack{e \text{ on the path} \\ \text{from } v_1 \text{ to } v_3}} w(e) \;+\; \sum_{\substack{e \text{ on the path} \\ \text{from } v_3 \text{ to } v_2}} w(e) \\ &= d(v_1, v_3) \;+\; d(v_3, v_2). \end{aligned}$$

Conversely, suppose

$$d(v_1, v_2) = d(v_1, v_3) + d(v_3, v_2). \tag{15.1}$$

Let $v_1 = u_1, u_2, \ldots, u_n = v_3$ and $v_3 = w_1, w_2, \ldots, w_m = v_2$ be the unique paths from v_1 to v_3 and v_3 to v_2, respectively. Let $u_i = w_j$ be the earliest vertex in the first path which also appears in the second. (Since $u_n = w_1$ there must be such a vertex.) Then $v_1 = u_1, u_2, \ldots, u_i = w_j, w_{j+1}, \ldots, w_m$ has no repetitions, and so must be the unique path from v_1 to v_2. But then in equation (15.1), all edge weights summed in the left hand side appear on the right as well, but the right has additional terms $2w(e)$ for all edges in the path from u_i to v_3. Since $w(e) > 0$ for all e, the equality can only hold if there are no edges in the path from u_i to v_3. Thus $u_i = v_3$, and we see v_3 lies on the path from v_1 to v_2. □

That this may be false if one allows 0 lengths of edges is seen most easily on an unrooted 3-leaf tree, with v_1, v_2, v_3 the leaves. If the edge containing v_3 has length 0, then, regardless of the other lengths, $d(v_1, v_2) = d(v_1, v_3) + d(v_3, v_2)$ even though v_3 is not on the path from v_1 to v_2.

14. a. Note that if $a, b \geq 0$, then $\max(a, b) \leq a + b$. Applying this with $a = d(v_1, v_2)$ and $b = d(v_2, v_3)$ gives the claim.

b. Let v_1, v_2, v_3 be three leaves on an ultrametric tree, the vertex u their most recent common ancestor, and ρ the root of the tree. Then since $d(\rho, v_1) = d(\rho, v_2) = d(\rho, v_3)$ and u lies on the path between ρ and v_i for $i = 1, 2, 3$, we see $d(u, v_1) = d(u, v_2) = d(u, v_3)$. Thus by restricting to the subtree composed only of those edges between u and the v_i, we need only consider the case of a 3-taxon ultrametric tree.

But for a 3-taxon ultrametric tree, with pendent branch lengths a, a, b, the interleaf distances are $2a, 2b, 2b$. Plugging these into the strong triangle inequality, in all orderings, we check that the inequality holds.

c. A three-leaf unrooted tree with edge lengths 2, 2, and 1 gives distances between the leaves of 3, 3, and 4, but $4 \not\leq \max(3, 3)$.

d. Suppose the three leaf-to-leaf tree metric distances are $a \leq b \leq c$. Applying the strong triangle inequality we see $c \leq \max(a, b) = b$. Thus the leaf-to-leaf distances are $a \leq b < c$.

e. By (d), the three leaf-to-leaf tree metric distances are $a \leq b = c$. These distances then determine the edge lengths of the tree as $a/2, a/2, b - a/2$. Since $b - a/2 \geq a/2$, a root can be placed on the longest edge to yield an ultrametric tree. Specifically, the root should be $b/2$ from all leaves.

17. 2^{n-2}

18. $(2n - 3)2^{n-2}$

Chapter 3

6. Suppose the character takes on state s_1 on two taxa a, b and a different state s_2 on taxa c, d. (It may take on these states at other taxa as well.) Then consider the two trees $T_1 = (((a, b), (c, d)), t)$ and $T_2 = (((a, c), (b, d)), t)$ where t denotes Newick notation for any fixed tree on the other taxa. Then applying the Fitch-Hartigan algorithm to both trees, when we arrive at the most recent common ancestor of a, b, c, d we have a score of 1 for T_1 and 2 for T_2, but both give the set of states $\{s_1, s_2\}$. Since this set is the same for both, and the tree t is the same, the rest of Fitch-Hartigan will result in identical contributions to the score. Thus the scores for T_1 and T_2 will differ.

7. a. We only need to rule out the possibility of $\tilde{\chi}(\rho)$ being different from both $\tilde{\chi}(v_1)$ and $\tilde{\chi}(v_2)$. But if that were the case, then by redefining $\tilde{\chi}(\rho)$ to have the same value as $\tilde{\chi}(v_1)$ we could reduce the count of edges on which there was a state change by 1. That contradicts the minimality of $\tilde{\chi}$.

   b. In case (1), if neither of the $\tilde{\chi}_i$ is minimal, we could redefine $\tilde{\chi}$ on each of the subtrees T_1 and T_2 so the count of edges in those subtrees on which there were state changes would go down by at least $1 + 1 = 2$. While this might introduce 2 additional changes on the edges descending from ρ, by next changing the state at ρ to match one of its children, we can be sure the increase is by at most 1. The net change would then be a decrease of 1, which contradicts the minimality of $\tilde{\chi}$.

   One example is the tree on the right of Figure 3.9. A smaller one is the tree $((a, b), (c, d))$ with character

$$\chi(a) = A, \quad \chi(b) = A, \quad \chi(c) = C, \quad \chi(d) = T,$$

extended to state A at all internal nodes.

16. If the weight matrix W is symmetric, the score will be independent of the root. Moving the root from one vertex to an adjacent one only changes the direction of one edge. If the states at these vertices are i and j, the score of a character extension will change by replacing the weight w_{ij} with w_{ji}. If W is symmetric, these are the same.

18. An example is:

    a: AAAAA
    b: AAACC
    c: GGGAA
    d: GGGCC

    as can be verified by computing weighted and unweighted scores on each of the 3 possible trees. Informally, in unweighted parsimony the first 3 sites overwhelm the last 2, leading to the tree $((a, b), (c, d))$. With a 1:2 transition:transversion weight, the last 2 sites count more, and overwhelm the first 3, leading to the tree $((a, c), (b, d))$.

21. If this inequality did not hold, then we would view a 2-step change of states $i \to j \to k$ as less costly than a 1-step change $i \to k$. This violates the entire spirit of parsimony, since we would be saying we prefer to imagine hidden changes occurred when we have no direct observation of them.

Chapter 4

2. If the n taxa are $x_1, x_2, \ldots, x_n$ then in specifying a split $X_0|X_1$ we can always assume that $x_n \in X_0$. Each of the other $n-1$ taxa may be placed in either X_0 or X_1, so we have 2^{n-1} options. However, this overcounts by 1, since if we place all of them in X_0 then X_1 would be empty, and we would not have specified a split.

3. For some taxon set X, consider a trivial split $X_0|X_1$ and any other split $X_0'|X_1'$. Then one of the X_i is a singleton set, say $X_0 = \{a\}$. Since $X_0'|X_1'$ is a split, a is an element of one of the split sets and not an element of the other. If X_i' is the one that a is not in, then $X_0 \cap X_i' = \emptyset$. Thus the splits are compatible.

4. That there is at least one pair with $Y_i \cap Y_j' = \emptyset$ follows from the definition of compatibility. To see there cannot be two such pairs, suppose there were. After possibly relabeling, we may suppose the first is $Y_0 \cap Y_0' = \emptyset$.

   If $Y_0 \cap Y_1' = \emptyset$ as well, then $\emptyset = (Y_0 \cap Y_0') \cup (Y_0 \cap Y_1') = Y_0 \cap (Y_0' \cup Y_1') = Y_0 \cap Y = Y_0$, which contradicts that Y_0 is non-empty. Similarly one sees that $Y_1 \cap Y_0' = \emptyset$ is impossible.

Finally, if $Y_1 \cap Y_1' = \emptyset$, then we note that this implies $Y_1' \subseteq Y_0$ (see Exercise 5 for details), while $Y_0 \cap Y_0' = \emptyset$ implies $Y_0 \subseteq Y_1'$. Thus $Y_0 = Y_1'$ and the two splits are the same.

5. Suppose the splits $Y_0|Y_1$ and $Y_0'|Y_1'$ are compatible. After relabeling the split sets if necessary, we may suppose $Y_0 \cap Y_0' = \emptyset$. But then

$$Y_0 = Y_0 \cap Y = Y_0 \cap (Y_0' \cup Y_1') = (Y_0 \cap Y_0') \cup (Y_0 \cap Y_1') = (Y_0 \cap Y_1'),$$

so $Y_0 \subseteq Y_1'$.

Conversely, if $Y_0 \subseteq Y_1'$, then since Y_0' is disjoint from Y_1', we must have $Y_0 \cap Y_0' = \emptyset$.

That $Y_i \subseteq Y_j$ if, and only if, $Y_{1-i} \supseteq Y_{1-j}$ results from the fact that taking complements in Y reverses set inclusions.

7. We proceed by induction on the number of splits in the collection. the base case of one split is obvious, as there is one edge in the tree, and the labels from the split sets must be on its two ends.

Now assume the result for collections of n splits, and suppose we have $n+1$. Suppose there are two trees displaying exactly this collection. Pick one split, and in each tree "contract" the edge corresponding to it. The resulting trees display the same n splits and hence are isomorphic. In fact, they must agree with the tree we would get from Tree Popping with the remaining n splits. Thus what we must show is that there is only one way the chosen split can be reintroduced onto this tree. Color the taxa according to the split sets.

Consider any vertex that could be replaced with an edge to introduce the split. If that vertex were removed, each of the resulting connected components would have to have only taxa of a single color on it (why?). If there is more than one such component of a single color, we conclude that this vertex is on the minimal spanning tree for that color. If there is only one component of a color there are two possibilities: Either the removed vertex also had that color, and hence was on the minimal spanning tree of that color, or the edge leading into that component already gave us the split, which is a contradiction. Thus the vertex must be on the intersection of the minimal spanning trees, and we already know this vertex is unique.

Finally, if we "expand" this vertex to an edge to introduce the split, all components of one color must be attached at one end, and all of the other color at the other end. Thus the expansion is unique.

13. Suppose $X_0'|X_1'$ is a trivial split, with, say $X_0' = \{a_i\}$. If for each choice of $a_j, a_k \in X_1'$ the quartet $a_i a_n | a_j a_k$ is in $\mathcal{Q}(T)$, then we join a_n to the pendent edge leading to a_i, to form a new cherry.

Chapter 5

6. a. To justify an attempt to construct a tree from dissimilarities, we assume there is a metric tree which has distances between leaves matching the dissimilarities, at least approximately. But distances between nodes in a metric tree are defined so they are additive, so the dissimilarity should be as well.

   b. The simplest example is a sequence with only 1 site, say S0:`A`, S1:`G`, S2:`C`. Then the Hamming distance between any pair of taxa would be 1, but $1 + 1 \neq 1$. For a more extreme example, take S2:`A`, and then observe $1 + 1 \neq 0$.

   c. Note that the non-additivity in part (b) is caused by repeated substitutions at the site. If the sequences are chosen so there is only one substitution per site, then the Hamming distance would be additive. If mutations are sufficiently rare, then repeated substitutions at a site will be even more rare, and the Hamming distance will be approximately additive.

9. An n-taxon unrooted binary tree has $2n - 3$ edges and there are $\binom{n}{2} = n(n-1)/2$ pairs of taxa. Thus we have a system of $n(n-1)/2$ equations in $2n - 3$ unknowns. Note that

$$\frac{n(n-1)}{2} > 2n - 3$$

   is equivalent to

$$n^2 - 5n + 6 > 0.$$

   Since $n^2 - 5n + 6 = (n-2)(n-3)$, we will have more equations than unknowns for all $n \geq 4$.

12. Solution 1: The argument is by induction, with the bases case of $n = 3, 4$ clear. Any n-taxon tree can be obtained from an $(n-1)$-taxon tree by subdividing an edge, and attaching an additional pendent edge at the node that was introduced. The $(n-1)$-taxon tree has at least 2 cherries. If the new edge is not attached at any edge in these cherries, there will still be at least 2 cherries. If the new edge is attached in one of the cherries, it destroys that cherry, but creates a new one. Thus there will still be at least 2 cherries. (Note: A more thorough proof would explain why inserting a new taxon cannot destroy more than 1 existing cherry, except in the $n = 3$ case.)

    Solution 2: A binary n-taxon tree has at $n - 2$ internal nodes. Since each taxon is joined by an edge to an internal vertex, by the pigeonhole principle there are either at least 3 joined to the same internal vertex, or at least 2 cherries. For a binary tree, the first case can only happen when $n = 3$, in which case it is clear there are $3 > 2$ cherries.

14. The four-point condition states that for any $x, y, z, w \in X$,

$$\delta(x, y) + \delta(z, w) \leq \max\{\delta(x, z) + \delta(y, w), \delta(x, w) + \delta(y, z)\}.$$

If $z = w$ this becomes

$$\delta(x, y) \leq \max\{\delta(x, z) + \delta(y, z), \delta(x, z) + \delta(y, z)\}.$$

Since the two expressions in the maximum are the same, this is just

$$\delta(x, y) \leq \delta(x, z) + \delta(z, y).$$

16. To show the claim, it's enough to show the following: For any three real numbers A, B, C,

$$\left.\begin{array}{l} A \leq \max(B, C) \\ B \leq \max(A, C) \\ C \leq \max(A, B) \end{array}\right\} \text{ if, and only if the two largest of } A, B, C \text{ are equal.}$$

Suppose the 3 inequalities hold and we order $A \leq B \leq C$, renaming them if necessary. Then $C \leq \max(A, B)$ implies $C \leq B$ hence $C = B$. Conversely, if the two largest of A, B, C are equal, then the three inequalities hold since the maximum in all cases will give the largest value.

18. With 3 taxa, the three-point formulas determine the unique edge lengths to fit any dissimilarities to a tree. Since the four-point condition holds, by Exercise 14, so does the triangle inequality. In particular, $\delta_{bc} \leq \delta_{ab} + \delta_{ac}$, so

$$0 \leq \frac{\delta_{ab} + \delta_{ac} - \delta_{bc}}{2}.$$

This show one edge length is non-negative. The same argument with the taxa interchanged shows the other edge lengths are non-negative.

19. Suppose, without loss of generality that the 4-point condition holds with

$$\delta_{ab} + \delta_{cd} \leq \delta_{ac} + \delta_{bd} = \delta_{ad} + \delta_{bc}.$$

Then by the discussion of the four-point condition in the text, the only possible unrooted topology for the tree is $((a, b), (c, d))$. We can then easily see that the central edge must have length

$$u = \frac{\delta_{ac} + \delta_{bd} - \delta_{ab} - \delta_{cd}}{2} \geq 0.$$

If x, y, z, w denote the edge lengths of pendent edges to taxa a, b, c, d, then by the 3-point formulas and Exercise 16, they must be given by the non-negative numbers

$$\begin{aligned} x &= \frac{\delta_{ab} + \delta_{ac} - \delta_{bc}}{2} \geq 0, \\ y &= \frac{\delta_{ab} + \delta_{bc} - \delta_{ac}}{2} \geq 0, \\ z &= \frac{\delta_{ac} + \delta_{cd} - \delta_{ad}}{2} \geq 0, \\ w &= \frac{\delta_{ad} + \delta_{cd} - \delta_{ac}}{2} \geq 0. \end{aligned}$$

Note that there are alternate formulas we could have given for u, x, y, z, w, and it is not yet clear that these in fact fit the dissimilarities. Thus we must check that six equations hold. Here we only check the one for the path from b to d:

$$\begin{aligned} y + u + w &= \frac{\delta_{ab} + \delta_{bc} - \delta_{ac}}{2} + \frac{\delta_{ac} + \delta_{bd} - \delta_{ab} - \delta_{cd}}{2} + \frac{\delta_{ad} + \delta_{cd} - \delta_{ac}}{2} \\ &= \frac{\delta_{bc} + \delta_{bd} + \delta_{ad} - \delta_{ac}}{2}. \end{aligned}$$

But by the equality of the four-point condition we can replace $\delta_{bc} + \delta_{ad}$ with $\delta_{bd} + \delta_{ac}$, and then see

$$y + u + w = \delta_{bd}.$$

Checking the other 5 equations is similar, with several requiring the use of the same four-point equality.

21. a) Note that

$$d(Si, v) = \frac{d(Si, Sj) + d(Si, G) - d(Sj, G)}{2},$$

where G is the set of all taxa except Si, Sj. But

$$d(Si, G) = \frac{1}{N-2} \sum_{k \neq i,j} d(Si, Sk) = \frac{R_i - d(Si, Sj)}{N-2},$$

and similarly

$$d(Sj, G) = \frac{1}{N-2} \sum_{k \neq i,j} d(Sj, Sk) = \frac{R_i - d(Sj, Si)}{N-2}.$$

Substituting these last two expressions into the first yields the first two formulas. The third formula follows from the first two. b) This is just one of the 3-point formulas.

Chapter 6

6. a) If x denotes the state at the root, and y the state at the other internal node, then of the 16 terms there is 1 ($x = \mathtt{G}, y = \mathtt{G}$) that is $(.25)(.9)^4$, 3 ($x = \mathtt{A}, \mathtt{C}, \mathtt{T}, y = \mathtt{G}$) that are $(.24)(.9)^2(.1/3)^2$, 6 ($x = \mathtt{G}, y = \mathtt{A}, \mathtt{C}, \mathtt{T}$ and $x = y = \mathtt{A}, \mathtt{C}, \mathtt{T}$) that are $(.24)(.9)(.1/3)^3$, and 6 (all others) that are $(.24)(.1/3)^4$. The sum is 0.16475185185.

   b) Of the 16 terms there are 2 ($x = \mathtt{G}, \mathtt{T}, y = \mathtt{G}$) that are $(.25)(.9)^3(.1/3)$, 3 ($x = y = \mathtt{T}$ and $x = \mathtt{A}, \mathtt{C}, y = \mathtt{G}$) that is $(.25)(.9)^2(.1/3)^2$, 4 ($x = \mathtt{T}, y =$

$\mathtt{A}, y = \mathtt{C}$ and $x = y = \mathtt{C}, \mathtt{G}$) that are $(.24)(.9)(.1/3)^3$, and 7 (all others) that are $(.24)(.1/3)^4$. The sum is 0.012858950617284.

c) Of the 16 terms there is 1 ($x = \mathtt{G}, y = \mathtt{G}$) that is $(.25)(.9)^3(.1/3)$, 2 ($x = \mathtt{G}, \mathtt{T}, y = \mathtt{T}$) that is $(.25)(.9)^2(.1/3)^2$, 4 ($x = \mathtt{A}, \mathtt{T}, y = \mathtt{C}$ and $x = \mathtt{C}, \mathtt{T}, y = \mathtt{A}$) that are $(.25)(.1/3)^4$, and 9 (all others) that are $(.24)(.9)^1(.1/3)^3$. The sum is 0.006601234567901.

10. Since the rooted tree has $n-1$ internal vertices, and $2n-2$ edges, there are 4^{n-1} terms, each of which is a product of $2n-1$ parameters.

17. To show $e^{Qt}\mathbf{u} = \mathbf{u}$, note that $Q\mathbf{u} = \mathbf{0}$, so

$$\begin{aligned} e^{Qt}\mathbf{u} &= (I + Qt + \frac{1}{2}Q^2t^2 + \frac{1}{3!}Q^3t^3 + \frac{1}{4!}Q^4t^4 + \dots)\mathbf{u} \\ &= I\mathbf{u} + tQ\mathbf{u} + \frac{1}{2}t^2Q^2\mathbf{u} + \frac{1}{3!}t^3Q^3\mathbf{u} + \frac{1}{4!}t^4Q^4\mathbf{u} + \dots \\ &= \mathbf{u} + \mathbf{0} + \mathbf{0} + \mathbf{0} + \dots \\ &= \mathbf{u} \end{aligned}$$

27. Let $\mathbf{u}$ denote a column vector of 1s. Then using equation (6.12) we have

$$\mathbf{p}M = \mathbf{u}^T \operatorname{diag}(\mathbf{p})M = \mathbf{u}^T M^T \operatorname{diag}(\mathbf{p}) = (M\mathbf{u})^T \operatorname{diag}(\mathbf{p}) = \mathbf{u}^T \operatorname{diag}(\mathbf{p}) = \mathbf{p}.$$

30. b) If the entries of Q are q_{ij}, then

$$-\operatorname{Tr}(\operatorname{diag}(\mathbf{p})Q) = -\sum_{i=1}^{4} p_i q_{ii}.$$

Since $-q_{ii}$ is the rate at which state i changes into a different state, is just a weighted average of the rates of change of the various states, weighted by the frequency at which those states occur. Alternately, since the p_i are probabilities, it is the expected value of the rate of change of a randomly chosen state.

If this value is c, replacing Q by $\frac{1}{c}Q$ rescales so it will be 1.

Chapter 8

6. Using the notation n_{ij} for data counts of individuals of genotype A_iA_j, the log-likelihood function is

$$\begin{aligned} \ln(L(p_1, p_2)) &= n_{11}\ln(p_1^2) + n_{22}\ln(p_2^2) + n_{33}\ln((1-p_1-p_2)^2) + n_{12}\ln(p_1p_2) \\ &\qquad + n_{13}\ln(p_1(1-p_1-p_2)) + n_{23}\ln(p_2(1-p_1-p_2)) \\ &= (2n_{11} + n_{12} + n_{13})\ln p_1 + (2n_{22} + n_{12} + n_{23})\ln p_2 \\ &\qquad + (2n_{33} + n_{13} + n_{23})\ln(1-p_2-p_2). \end{aligned}$$

Setting the partial derivatives with respect to p_1 and p_2 to be 0 yields

$$0 = \frac{2n_{11} + n_{12} + n_{13}}{p_1} - \frac{2n_{33} + n_{13} + n_{23}}{1 - p_2 - p_2},$$
$$0 = \frac{2n_{22} + n_{12} + n_{23}}{p_2} - \frac{2n_{33} + n_{13} + n_{23}}{1 - p_2 - p_2}.$$

Letting $a = 2n_{11} + n_{12} + n_{13}$, $b = 2n_{22}+n_{12}+n_{23}$, and $c = 2n_{33} + n_{13} + n_{23}$, this means

$$\frac{a}{p_1} = \frac{c}{1 - p_1 - p_2} = \frac{b}{p_2},$$

so $p_2 = \frac{b}{a}p_1$. Substituting this expression for p_2 into $a(1 - p_1 - p_2) = cp_1$ gives

$$a - ap_1 - bp_1 = cp_1,$$

so

$$\hat{p}_1 = \frac{a}{a + b + c} = \frac{2n_{11} + n_{12} + n_{13}}{2n},$$

where n is the total number of individuals sampled.

Similarly,

$$\hat{p}_2 = \frac{2n_{22} + n_{12} + n_{23}}{2n}, \quad \hat{p}_3 = \frac{2n_{33} + n_{13} + n_{23}}{2n}.$$

7. Let n_{ij} be the count of sites with base i in S0 and base j in S1, and n the total number of sites. Let

$n_T = \#$ of transition sites $= n_{12} + n_{21} + n_{34} + n_{43}$,
$n_R = \#$ of transversion sites $= n_{13} + n_{14} + n_{23} + n_{24} + n_{31} + n_{32} + n_{41} + n_{42}$,
$n_C = \#$ of constant sites $= n - n_T - n_R$.

Then reasonable formulas for estimators, which we will show are in fact the ML estimators, are

$$\hat{b} = \frac{n_T}{n}, \quad \hat{c} = \frac{n_R}{2n}.$$

The log-likelihood function is

$$\ln(L(b, c)) = n_T \ln(b/4) + n_R \ln(c/4) + n_C \ln((1 - b - 2c)/4).$$

Setting the partial derivatives with respect to p_1 and p_2 to be 0 yields

$$0 = \frac{n_T}{b} - \frac{n_C}{1 - b - 2c},$$
$$0 = \frac{n_R}{c} - \frac{2n_C}{1 - b - 2c}.$$

Thus

$$\frac{n_T}{b} = \frac{n_C}{1-b-2c} = \frac{n_R}{2c},$$

so $c = \frac{n_R}{2n_T}b$. Substituting this expression for c into $n_T(1-b-2c) = n_C b$ gives

$$n_T - n_T b - n_R b = n_C b,$$

so

$$\hat{b} = \frac{n_T}{n_T + n_R + n_C} = \frac{n_T}{n}.$$

The formula for $\hat{c}$ follows from $c = \frac{n_R}{2n_T}b$.

11. There are $2N-2$ parameters. Of these, $N-3$ are edge lengths, and 1 is for the rate matrix Q. (Although the K2P model has two rate parameters in Q, one of these can be arbitrarily set to 1, since the effect of this is just to rescale edge lengths by a constant factor.)

Chapter 9

10. a) If a subset of 4 taxa includes A_n, then the two trees induce different quartet trees. If a subset does not include A_n, they induce the same quartet tree. Therefore the distance is the number of subsets of 4 taxa that contain A_n. That is is the same as the number of ways we can pick 3 taxa from $A_1, A_2, \ldots A_{n-1}$, which is $\binom{n-1}{3} = \frac{(n-1)(n-2)(n-3)}{1\cdot 2\cdot 3}$.

    b) There are $\binom{n}{4} = \frac{n(n-1)(n-2)(n-3)}{1\cdot 2\cdot 3\cdot 4}$. subsets of 4 taxa, so the proportion of them counted in part (a) is $\frac{4}{n}$.

Chapter12

5. (b) For $(10, 10, 10)$ and $(100, 100, 100)$ peaks are at $(1/3, 1/3)$, with the second distribution being more tightly concentrated around the peak.

    (c) The distributions have peaks at $(0, 1/3)$, $(9/57, 19/57)$, and $(99/597, 199/597)$, all of which are approximately $(1/6, 2/6)$, and become increasingly concentrated around those values.

    (d) Following the given suggestion, from setting partial derivatives to 0 we find the maximum occurs where

$$\frac{\alpha_i - 1}{x_i} - \frac{\alpha_k - 1}{1 - \sum_{i=1}^{k-1} x_i} = 0$$

for $i = 1, 2, \ldots, k-1$, or

$$x_i = \frac{1 - \sum_{i=1}^{k-1} x_i}{\alpha_k - 1}(\alpha_i - 1) = \frac{x_k}{\alpha_k - 1}(\alpha_i - 1).$$

Equivalently

$$(x_1, x_2, \ldots, x_k) = s(\alpha_1 - 1, \alpha_2 - 1, \ldots, \alpha_k - 1),$$

with $s = x_k/(\alpha_k - 1)$. But since $\sum_{i=1}^{k} x_i = 1$, we also see that

$$s \sum_{i=1}^{k} (\alpha_i - 1) = 1$$

so

$$s = 1/\sum_{i=1}^{k} (\alpha_i - 1).$$

Thus

$$x_i = \frac{\alpha_i - 1}{\sum_{i=1}^{k} (\alpha_i - 1)}.$$

Chapter 13

3. There are $5 \cdot 3 = 15$ pairs with coalescent time 10, $1 \cdot 4 = 4$ pairs with time 6, $2 \cdot 1 = 2$ pairs with time 5, $2 \cdot 2 = 4$ pairs with time 4, and 3 pairs with time 1. The average coalescent time is then

$$\frac{15 \cdot 10 + 4 \cdot 6 + 2 \cdot 5 + 4 \cdot 4 + 3 \cdot 1}{28} = \frac{203}{28} = 7.25.$$

This is reasonably close to 8, espcially considering we have a relatively small sample of non-independent coalescent times.

Note that what is calculated here is *not* the same as the average of the times of the 7 coalescent events, which is only 4.

11. a) The probability that A_1 and A_2 are the first to coalesce is $1/\binom{n}{2}$. The probability that this lineage and A_3 are the next to coalesce is $1/\binom{n-2}{2}$. Continuing in this way the probability of the specific caterpillar is

$$\frac{1}{\binom{n}{2}} \frac{1}{\binom{n-1}{2}} \frac{1}{\binom{n-2}{2}} \cdots \frac{1}{\binom{2}{2}} = \frac{2^{n-1}}{n!(n-1)!}.$$

b) Which specific caterpillar is formed depends on ordering the taxa so the first two coalesce first, then that ineage coalesces with the 3rd, etc. Since the n taxa can be ordered in $n!$ ways, but the order of the first pair does not matter, there are $n!/2$ caterpillars. Multiplying this times the answer in (a) yields $2^{n-2}/(n-1)!$.

Bibliography

[AR04] Elizabeth S. Allman and John A. Rhodes. *Mathematical Models in Biology: An Introduction*. Cambridge University Press, Cambridge, 2004.

[Bry03] D. Bryant. A classification of consensus methods for phylogenies. In M. Janowitz, F.-J. Lapointe, F.R. McMorris, B. Mirkin, and F.S. Roberts, editors, *BioConsensus*, pages 163–184. DIMACS AMS, 2003.

[Fel04] Joseph Felsenstein. *Inferring Phylogenies*. Sinauer Associates, Sunderland, MA, 2004.

[HRS10] Daniel H. Huson, Regula Rupp, and Celine Scornavacca. *Phylogenetic Networks. Concepts, Algorithms and Applications*. Cambridge University Presss, Cambridge, 2010.

[KK10] L. Knowles and L. Kubatko, editors. *Estimating Species Trees: Practical and Theoretical Aspects*. Wiley-Blackwell, College Station, Texas, 2010.

[SS03] Charles Semple and Mike Steel. *Phylogenetics*, volume 24 of *Oxford Lecture Series in Mathematics and its Applications*. Oxford University Press, Oxford, 2003.

[Wak09] John Wakeley. *Coalescent Theory: An Introduction*. Roberts and Company, Greenwood Village, Colorado, 2009.

[Wat95] Michael S. Waterman. *Introduction to Computational Biology: Maps, Sequences and Genomes*. Chapman and Hall, London, 1995.

[Yan06] Ziheng Yang. *Computational Molecular Evolution*. Oxford University Press, Oxford, 2006.