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

$$\operatorname{diag}(\mathbf{p}_0)M(t) = \begin{pmatrix} (1-a)/4 & a/12 & a/12 & a/12 \\ a/12 & (1-a)/4 & a/12 & a/12 \\ a/12 & a/12 & (1-a)/4 & a/12 \\ a/12 & a/12 & a/12 & (1-a)/4 \end{pmatrix}. \tag{7.2}$$

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

$$d_{JC}(\mathrm{S0}, \mathrm{S1}) = -\frac{3}{4} \ln \left(1 - \frac{4}{3} \hat{a}\right).$$

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 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.

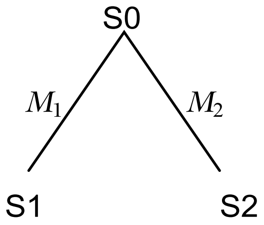

Figure 7.1: Two descendants of an ancestor. Under a time-reversible model, we may view $S1$ as ancestral to $S0$ which is ancestral to $S2$.

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.

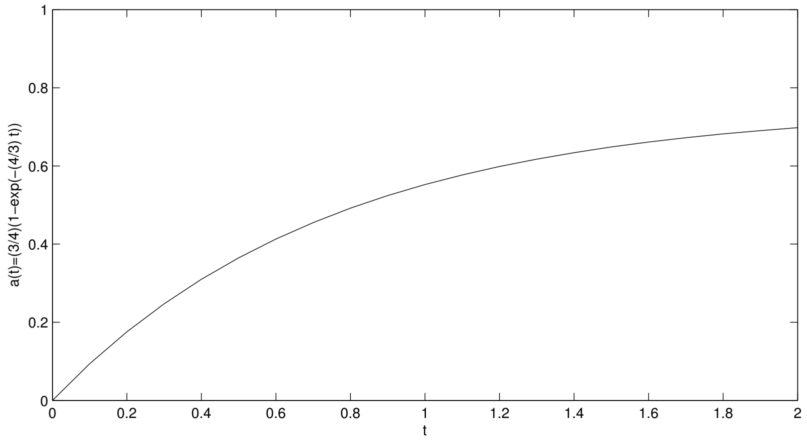

Figure 7.2: The relationship between elapsed time t and the probability a of differing states in a character for 2 taxa under the Jukes-Cantor model, with rate $\alpha = 1$. This graph can also be read as the Hamming distance $\hat{a}$ between two sequences on the vertical axis, and the resulting Jukes-Cantor 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 $0 \le t \le .2$), 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 $\mathtt{A} \leftrightarrow \mathtt{C}$ and

G ↔ T, and $\hat{d}$ is the proportion showing transversions of the type A ↔ T and G ↔ 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, $\mathrm{S0} = \mathrm{S1}$,

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

$$d_{LD}(\mathrm{S0}, \mathrm{S1}) = -\frac{1}{4}\left(\ln\left|\det(\hat{F})\right| - \frac{1}{2}\ln(g_0g_1)\right),$$

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 $t_1, t_2, \dots, t_n$. 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:

$$\begin{aligned}
b &= \frac{1}{4}\left(1 - e^{-(2\beta+2\delta)t} + e^{-(2\gamma+2\delta)t} - e^{-(2\beta+2\gamma)t}\right),\\
c &= \frac{1}{4}\left(1 + e^{-(2\beta+2\delta)t} - e^{-(2\gamma+2\delta)t} - e^{-(2\beta+2\gamma)t}\right),\\
d &= \frac{1}{4}\left(1 - e^{-(2\beta+2\delta)t} - e^{-(2\gamma+2\delta)t} + e^{-(2\beta+2\gamma)t}\right).
\end{aligned}$$

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 $d_{LD}(\mathrm{S0}, \mathrm{S1}) = d_{JC}(\mathrm{S0}, \mathrm{S1})$.

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 $M_{0\to 1}$ and $M_{1\to 2}$, respectively. Let $M_{0\to 2} = M_{0\to 1}M_{1\to 2}$. Explain why $\mathbf{p}_1 = \mathbf{p}_0 M_{0\to 1}$ and $\mathbf{p}_2 = \mathbf{p}_1 M_{1\to 2}$ are the base distributions in S1 and S2 respectively, and explain the meaning of $M_{0\to 2}$.

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 $0 \leq i < j \leq 2$, define the matrix

$$N_{i\to j} = D_i M_{i\to j} D_j^{-1}.$$

Show $N_{0\to 1}N_{1\to 2} = N_{0\to 2}$, and use fact (i) to conclude

$$\ln|\det(N_{0\to 1})| + \ln|\det(N_{1\to 2})| = \ln|\det(N_{0\to 2})|.$$

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 $P = \operatorname{diag}(\mathbf{p})e^{Qt}$, explain why $-\operatorname{Tr}(\operatorname{diag}(\mathbf{p}) \ln(\operatorname{diag}(\mathbf{p})^{-1}P)$ 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

$$\begin{aligned} L(p_1, p_2) &= (p_1^2)^{n_{hh}}\,(p_1(1-p_1))^{n_{ht}}\,((1-p_1)p_2)^{n_{th}}((1-p_1)(1-p_2))^{n_{tt}}, \\ &= p_i^{2n_{hh}+n_{ht}}(1-p_1)^{n_{ht}+n_{th}+n_{tt}}p_2^{n_{th}}(1-p_2)^{n_{tt}}. \end{aligned}$$

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

$$\begin{aligned} P = (p_{ij}) &= \operatorname{diag}(p_0)M \\ &= \begin{pmatrix} (1-a)/4 & a/12 & a/12 & a/12 \\ a/12 & (1-a)/4 & a/12 & a/12 \\ a/12 & a/12 & (1-a)/4 & a/12 \\ a/12 & a/12 & a/12 & (1-a)/4 \end{pmatrix}. \end{aligned}$$

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,

$$\ln L_T = \sum_{(i_1,\dots,i_n)\in\{A,G,C,T\}^n} n(i_1,\dots,i_n)\ln(p(i_1,\dots,i_n)), \tag{8.6}$$

where $p(i_1,\dots,i_n)$ is a function of the numerical parameters giving the probability of observing the pattern $(i_1,\dots,i_n)$ at a site (as computed in Chapter 6), and $n(i_1,\dots,i_n)$ 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 $n(i_1, \dots i_n)$ 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 $p(i_1, \dots i_n)$.

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 $n(i_1, \dots, i_n) = 0$. 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 $p(i_1, \dots, i_n)$ 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 $p(i_1, \dots, i_n)$ 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).

```
            /\
           /  \
     M_4  /    \
         /      \
        /\ v     \ M_3
       /  \       \
M_1   /    \ M_2   \
     /      \       \
    A        T       C
```

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 $\mathcal{P}(\mathtt{ATC})$, 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 $\mathtt{A}, \mathtt{G}, \mathtt{C}, \mathtt{T}$ 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

$$\begin{aligned}\mathbf{c}_v(A) &= M_1(\mathtt{A},\mathtt{A})M_2(\mathtt{A},\mathtt{T})\\ &= (M_1(\mathtt{A},\mathtt{A})\cdot 1 + M_1(\mathtt{A},\mathtt{G})\cdot 0 + M_1(\mathtt{A},\mathtt{C})\cdot 0 + M_1(\mathtt{A},\mathtt{T})\cdot 0))\\ &\qquad ((M_2(\mathtt{A},\mathtt{A})\cdot 0 + M_2(\mathtt{A},\mathtt{G})\cdot 0 + M_2(\mathtt{A},\mathtt{C})\cdot 0 + M_2(\mathtt{A},\mathtt{T})\cdot 1).\end{aligned}$$

Similarly, if a `G` appeared at v, the first two leaves would have pattern `AT` with probability

$$\begin{aligned}\mathbf{c}_v(\mathtt{G}) &= M_1(\mathtt{G},\mathtt{A})M_2(\mathtt{G},\mathtt{T})\\ &= (M_1(\mathtt{G},\mathtt{A})\cdot 1 + M_1(\mathtt{G},\mathtt{G})\cdot 0 + M_1(\mathtt{G},\mathtt{C})\cdot 0 + M_1(\mathtt{G},\mathtt{T})\cdot 0))\\ &\qquad ((M_2(\mathtt{G},\mathtt{A})\cdot 0 + M_2(\mathtt{G},\mathtt{G})\cdot 0 + M_2(\mathtt{G},\mathtt{C})\cdot 0 + M_2(\mathtt{G},\mathtt{T})\cdot 1).\end{aligned}$$

The entries $\mathbf{c}_v(\mathtt{C})$ and $\mathbf{c}_v(\mathtt{T})$ 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

$$\begin{aligned}\mathbf{w}_1 &= M_4\mathbf{c}_v^T\\ \mathbf{w}_2 &= M_3\mathbf{c}_3^T,\end{aligned}$$

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.