**Notes for 848 lecture 6: A ML basis for compatibility and parsimony**

# Notation

$$\theta \in \Theta \tag{1}$$

- $\Theta$ is the space of all possible trees (and model parameters)
- $\theta$ is a point in the parameter space = a particular tree and a set of values for all of the
- $a \in b$ means that $a$ is "in" $b$.

$$X \in \mathcal{X} \tag{2}$$

- $\mathcal{X}$ is the space of all possible character matrices
- $X$ represents our specific data matrix.

Here is a slight abuse of terminology:

$$X \sim \Pr(X = x|\theta) \tag{3}$$

- $x$ is a specific "value" that a character matrix could assume. For a character matrix the value is the exact set of data patterns found.
- $\Pr(X = x|\theta)$ is the probability that our character matrix will have an exact composition described by $x$ if $\theta$ is the true value of the parameter.
- $a \sim b$ means that $a$ is a random variable drawn from the distribution described by the distribution $b$. We view our data set as drawn from a distribution described by a set of probability statements about what type of matrices could be generated by the course of evolution[1]

$$\mathcal{L}(\theta) = \Pr(X = x|\theta) \tag{4}$$
$$\mathcal{L}(\theta) = \Pr(X|\theta) \tag{5}$$

We will refer to $\Pr(X = x|\theta)$ as the **likelihood** of the model described by $\theta$. It is conditional on the fact that we have observed a single data set $X$, that has the characters described by the point $x$ in the space of all possible datasets.

- $\mathcal{L}(\theta)$ is the notation for the likelihood of "model" $\theta$
- $\Pr(X|\theta)$ is a simplified notation for $\Pr(X = x|\theta)$

The maximum likelihood estimate of the tree (and model parameters) is $\hat{\theta}$:

$$\hat{\theta} = \arg\max \mathcal{L}(\theta) \tag{6}$$

which simply means the value of point in parameter space $\Theta$ for which $\mathcal{L}(\cdot)$ achieves it highest value. (if there are ties, then there will be multiple (or even an infinite set) of maximum likelihood estimates.

[1] This is a bit of an abuse of terminology because we should say that $X$ is drawn from a distribution, rather than from a probability statement. What we mean, more technically, is that we assume that $X$ is a single draw from a multinomial distribution with the probabilities of the different categories being defined by statements like $\Pr(X = x|\theta)$ where $x$ represents the category of the multinomial. To formally express this we would have to introduce an indexing scheme for the multinomial. So we'll just be informal.

## A homoplasy-free model

We can build a model of perfect data. For simplicity, let's assume that we have binary characters. First, we assume that homoplasy is impossible then, there can only be 0 changes or 1 change across the whole tree. A simple assumption (but biologically unrealistic approach) is to say that, when a change occurs it is equally likely to occur on any one of the branches of the unrooted tree. Finally we'll say that we only considerable variable characters and we polarize the state codes by assigning 0 to the character state displayed in the outgroup (taxon A). Let's consider the case of the fully-resolved trees for the ingroup B, C, and D.

### How many possible character patterns are there?

We refer to a vector of characters states for each taxon as a pattern (or "character pattern" or "data pattern").

Imagine that we score human, dog, and cat, and frog for two characters: presence/absence of placenta and presence/absence of hair. We have two characters, but they both can be expressed as the vector of state codes: 1110 (if we order the taxa such that frog is last). If we do code the characters in this way then each characters is a different instance of the same pattern.

We can use $\mathcal{D}$ to represent the set of all possible patterns (or the "space of all possible patterns", if you prefer) and $\mathcal{D}_v$ to refer to the set of all variable patterns. There are $N$ taxa. Each of the ingroup taxa (there are $N-1$ ingroup taxa) can display any of the $k$ states, so the number of possible patterns is:

$$\#\text{possible patterns} \quad = \quad k^{(N-1)}, \tag{7}$$

but this includes the all 0 pattern which we said that we would exclude (when we said that we'll just look at variable characters).

$$|\mathcal{D}_v| = \#\text{possible variable patterns} \quad = \quad k^{(N-1)} - 1. \tag{8}$$

Here $N = 4$ and $k = 2$, so there are 7 patterns. The $|x|$ notation, when applied to a set means the size of the set.

### How many possible character matrices are there?

That depends on how many characters we sample. If we sample $M$ characters, then there are simply the number of possible character matrices.

$$\#\text{possible matrices of variable patterns} \quad = \quad [k^{(N-1)} - 1]^M \tag{9}$$

So there are 13,841,287,201 different matrices with 12 characters and 3 ingroup taxa when we only allow variable characters.

## The likelihood of a character matrix

If we assume that different characters are independent from each other then the probability of the entire matrix is simply the product of the probabilities of the $M$ different "events," where each

event corresponds to a different characters in the matrix. Let $X_i$ refer to character number $i$ of the character matrix $X$:

$$\mathcal{L}(\theta) = \Pr(X|\theta) = \prod_{i=1}^{M} \Pr(X_i|\theta) \tag{10}$$

This dramatically simplifies our life because we can break of the calculation into $M$ manageable ones. If two different characters display the same pattern, then they will have the same "character-likelihood"

### Pattern counts

Because of our assumption of independence, the full likelihood is simply a product over all characters (as shown in equation 10, above). Multiplication is commutative (which means that we can rearrange the term without affecting the result, $ab = ba$). This means that we can imagine rearranging the order of characters in our matrix without changing the likelihood – this is reassuring because we don't have any "correct" order to list our characters in.

Thus, we can also calculate the likelihood by calculating the likelihood from the counts of each type of pattern:

$$c_i = \sum_{j=1}^{M} I(x_j = d_i) \tag{11}$$

$$\mathcal{L}(\theta) = \Pr(X|\theta) = \prod_{i=1}^{|\mathcal{D}_v|} \Pr(d_i|\theta)^{c_i} \tag{12}$$

$$\tag{13}$$

Here $c_i$ is the number of characters in the matrix that display pattern $d_i$. There is not widely-used notion for the count of someting, so we show this mathematically by making it a sum across all characters (that is the $\sum_{j=1}^{M}$ notation) of and indicator function. An indicator function is a common notational convenience that means "a value of 1 if the condition is true, and 0 if the condition is false" So:

$$I(x_j = d_i) = \begin{cases} 1 & \text{if character } x_j \text{ displays pattern } d_i \\ 0 & \text{otherwise} \end{cases} \tag{14}$$

## The likelihood of a character

Under our 2-state model of homoplasy-free variable patterns, each unrooted tree can generate 5 data patterns and each patterns are generated with a probability of $0.2 = \frac{1}{5}$. The patterns correspond to what you would get if you label the leaf A with state 0, and then consider putting one transition on tree. There are 5 edges in the unrooted trees, such as the one shown in Figure 1 Table 1 shows the likelihood for the 7 types of data pattern on each of the three trees. Note that the red rows correspond to the data patterns that are informative in a Hennigian analysis.

Our simple model can be seen as a justification of the Hennigian approach in that it behaves in an identical fashion:

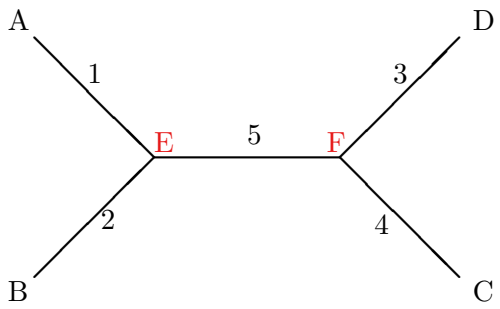
- If the matrix has at least one 0011 pattern (C and D with state 1 and A and B with state 0), then the B+C tree and the B+D tree will have a likelihood of 0.
- If the matrix has at least one 0101 pattern, then the B+C tree and the C+D tree will have a likelihood of 0.
- If the matrix has at least one 0110 pattern, then the B+D tree and the C+D tree will have a likelihood of 0.
- patterns that correspond to autapomorphies do not provide any phylogenetic information because they contribute the same character likelihood (0.2) to every tree.

So the presence of only one type of synapomorphy leads to the corresponding tree to be the maximum likelihood estimate of the tree under our model.

Table 1: Pattern likelihoods for variable perfect data model

| | $T_1$ (C+D) | $T_2$ (B+D) | $T_3$ (B+C) |
|---|---|---|---|
| 0001 | 0.2 | 0.2 | 0.2 |
| 0010 | 0.2 | 0.2 | 0.2 |
| 0011 | 0.2 | 0.0 | 0.0 |
| 0100 | 0.2 | 0.2 | 0.2 |
| 0101 | 0.0 | 0.2 | 0.0 |
| 0110 | 0.0 | 0.0 | 0.2 |
| 0111 | 0.2 | 0.2 | 0.2 |

Figure 1: The unrooted tree $AB|CD$ with edges labelled. Internal nodes are labelled in red.




## A model that can explain character conflict

The chief difficulty with the model that we just formulated is that it predicts *no* character conflict. So if a data matrix displays any character conflict, then all trees will have a likelihood of 0 – we will not be able to infer a tree and it will be obvious that our model is wrong. Given that almost all data sets display character conflict, the model is clearly inappropriate for real data.

Why do we get character conflict? What types of characters are likely to end up in our matrix? We must be more specific in order to construct a full probability that will allow us to infer trees

using ML

One possibility is that some characters that we have scored in our matrix are simply random noise mistaken as biologically significant. Imagine a trait that is plastic in all species such that the variability that we see phenotypically is the result of environmental variation, and that this environmental variability is patchy and essentially random with respect to the phylogeny. This is not a terribly likely situation, I'll admit.

If this were the case, then we can think of our matrix being composed of two classes of characters: the first being the informative characters that we are trying to collect, and a second class of "noise" characters that contaminate the matrix.

We have described a type of model is normally called a "mixture model." There are two classes of random processes acting and our data is a mixture of columns drawn from each of the process. To make things clear, lets use $\Pr(d_i|T_1, p)$ to represent the probability that the our homoplasy-free model (from above) would produce the pattern $d_i$ under tree 1; the $p$ will remind us that this is the perfect data model. We'll use $\Pr(d_i|T_1, n)$ to indicate the probability of the same pattern under the "noise" model (hence the $n$).

When we consider a character we do not know whether it is coming from the perfect model class or the noise model. We can deal with this by summing over each category while weighting by the probability that the site belongs to that category.

Basic probability states that:

$$\Pr(A) = \Pr(B)\Pr(A|B) + (1 - \Pr(B))\Pr(A| - B) \tag{15}$$

where $-B$ means "not B" (borrowing the notation from the complement meaning "anything other than $B$" in the set context). For our mixture model, we can construct:

$$\Pr(d_i|T_j) = \Pr(\text{perfect model})\Pr(d_i|T_j, p) + \Pr(\text{noise})\Pr(d_i|T_j, n) \tag{16}$$

We do not know the probability that a character that we score will be from the perfect model category. So we will introduce a parameter into our model that represents the proportion of time that we accidentally include noise in our matrix. We do not know the correct value for this parameter. Let's call this parameter $\epsilon$ (the Greek letter epsilon), it will represent the *a priori* probability that a character is random, "white" noise. You can also think of $\epsilon$ as the proportion of characters that are noise if we had an infinite number of characters in our matrix – or think of it as the expectation of the proportion of noise.

$$\Pr(d_i|T_j) = (1 - \epsilon)\Pr(d_i|T_j, p) + \epsilon\Pr(d_i|T_j, n) \tag{17}$$

We already discussed how to calculate $\Pr(d_i|T_j, p)$ (these probabilities were shown in table 1). How do we calculate $\Pr(d_i|T_j, n)$, the probability of a data pattern under the assumption that the character is pure noise? If we assume that each of the $k$ characters states are equally probable, then the probability of a pattern is simply $\left(\frac{1}{k}\right)^N$. Of course, we are conditioning our data on being variable and having state 0 in the outgroup. Recall that early we calculated the number of such patterns (equation 8). If they occur at the same frequency, then we can see that:

$$\Pr(d_i|T_j, n) = \frac{1}{|\mathcal{D}_v|} = \frac{1}{7} \tag{18}$$

Because the "noise" model reveals no phylogenetic information:

$$\Pr(d_i|T_1, n) = \Pr(d_i|T_2, n) = \Pr(d_i|T_3, n) = \frac{1}{7} \tag{19}$$

We can now see that if a pattern, $d$, is compatible with a tree, $T$, then the likelihood will be:

$$\Pr(d|T) = \frac{(1-\epsilon)}{5} + \frac{\epsilon}{7} \tag{20}$$

and if a pattern is incompatible with a tree, then the probability that we would see that pattern under the perfect model is 0 so we have:

$$\Pr(d|T) = \frac{\epsilon}{7} \tag{21}$$

To be more formal we can denote the set of patterns that are compatible with a tree as $D(T)$. Thus,

$$\Pr(d_i|T_j) = \begin{cases} \frac{(1-\epsilon)}{5} + \frac{\epsilon}{7} & \text{if } d_i \in D(T_j) \\ \frac{\epsilon}{7} & \text{if } d_i \notin D(T_j) \end{cases} \tag{22}$$

Note that:

$$\frac{(1-\epsilon)}{5} + \frac{\epsilon}{7} = (7 - 2\epsilon)/35$$

Table 2 shows the pattern likelihood under our "perfect+noise" mixture model. We can see that if $\epsilon = 0$, then the model collapses to the perfect-data model. If $\epsilon = 1$, then all characters are from the noise model; every pattern has the probability of $\frac{1}{7}$ regardless of tree, and we can no longer infer trees (from any dataset).

Table 2: Pattern likelihoods for under the perfect+noise mixture model

| | $T_1$ (C+D) | $T_2$ (B+D) | $T_3$ (B+C) |
|---|---|---|---|
| 0001 | $(7-2\epsilon)/35$ | $(7-2\epsilon)/35$ | $(7-2\epsilon)/35$ |
| 0010 | $(7-2\epsilon)/35$ | $(7-2\epsilon)/35$ | $(7-2\epsilon)/35$ |
| 0011 | $(7-2\epsilon)/35$ | $\epsilon/7$ | $\epsilon/7$ |
| 0100 | $(7-2\epsilon)/35$ | $(7-2\epsilon)/35$ | $(7-2\epsilon)/35$ |
| 0101 | $\epsilon/7$ | $(7-2\epsilon)/35$ | $\epsilon/7$ |
| 0110 | $\epsilon/7$ | $\epsilon/7$ | $(7-2\epsilon)/35$ |
| 0111 | $(7-2\epsilon)/35$ | $(7-2\epsilon)/35$ | $(7-2\epsilon)/35$ |

## General formulae for the perfect+noise mixture model

Above (equation 22) we presented the probability of a pattern on a four taxon tree with variable, 2-state characters scored. The more general formulae for probabilities of the patterns depend on

the number of states and the number of branches in the tree. In our "perfect data" (homoplasy-free) model a state changes occur across each branch with equal frequency, so the probability of each resulting pattern is just 1 over the number of edges in the tree. Fortunately there is a simple formula for the number of edges in a fully-resolved tree (or the size of the set of edges, $E$):

$$|E| = \#\text{ edges} = 2N - 3 \tag{23}$$

(I always remember this formula by remembering that every time we add a taxon we add to edges to the tree - one new terminal edge, and one edge in the previous tree is split into two edges in the new tree. This implies that the coefficient for $N$ will be 2. Then I remember that an unrooted two-taxon tree has one edges, so we must subtract three.)

The most general form of the perfect+noise mixture model would work for any number of states, but that gets complicated (because we have to consider multiple transition for a character across the tree even though there is no homoplasy). We will simply note that for two states we have:

$$\Pr(d_i|T_j) = \begin{cases} \frac{(1-\epsilon)}{2N-3} + \frac{\epsilon}{2^{(N-1)}-1} & \text{if } d_i \in D(T_j) \\ \frac{\epsilon}{2^{(N-1)}-1} & \text{if } d_i \notin D(T_j) \end{cases} \tag{24}$$

## The ML tree

### ML under the perfect+noise model is the max compatibility tree

The first problem that we face if we want to calculate the likelihood is the fact that we do not know the correct value for $\epsilon$. One of the most popular (and reliable) ways to deal with this issue is too estimate both the tree and $\epsilon$ using ML. Interestingly, in this case we will find that we do not even need to estimate the correct value of $\epsilon$ to infer the tree.

We can see that this is the case by "reparameterizing" the model. Let:

$$\phi = \frac{(7-2\epsilon)/35}{\epsilon/7} \tag{25}$$

$$= \frac{(7-2\epsilon)/5}{\epsilon} \tag{26}$$

$$= \frac{(7-2\epsilon)}{5\epsilon} \tag{27}$$

$$= \frac{1.4}{\epsilon} - 0.4 \tag{28}$$

Here $\phi$ is simple function of $\epsilon$, and $\phi > 1$ for range $0 < \epsilon < 1$.

When we express the likelihood for the patterns in the new parameterization we see that every probability is either $\epsilon/7$ (for all patterns that conflict with the tree), or $\phi\epsilon/7$ (for all patterns that

are compatible with the tree. This means that we can rephrase equation 11 as:

$$\Pr(X|T_j) = \prod_{i=1}^{|\mathcal{D}_v|} \Pr(d_i|\theta)^{c_i} \quad (29)$$

$$= \left( \prod_{d_i \in D(T_j)} \Pr(d_i|\theta)^{c_i} \right) \left( \prod_{d_i \notin D(T_j)} \Pr(d_i|\theta)^{c_i} \right) \quad (30)$$

$$= \left( \prod_{d_i \in D(T_j)} [\phi\epsilon/7] \right) \left( \prod_{d_i \notin D(T_j)} [\epsilon/7] \right) \quad (31)$$

$$= \phi^{C(T_j)} \left( \frac{\epsilon}{7} \right)^{|\mathcal{D}_v|} \quad (32)$$

where $C(T_j)$ is the sum of the counts of the characters that are compatible with tree $T_j$:

$$C(T_j) = \sum_{d_i \in D(T_j)} c_i \quad (33)$$

Note that our likelihood in equation 32 has a two factors: the first depends on $\epsilon$ and the count of the number of characters in the data matrix that are compatible with the tree that we are scoring; the second factor does not depend on the data set or the tree we are scoring. Because the second factor appears in the likelihood equation for every tree, we can simply ignore it when comparing trees. The tree with the largest $\phi^{C(T_j)}$ will have the highest likelihood. Because $\phi$ is always greater than one (for the restrictions that we have put on it), the tree with for which $C(T_j)$ is greatest will have the highest likelihood.

If we want to know exactly how much higher the likelihood would be, we would have to estimate a value for $\epsilon$ (which would then give us a value for $\phi$). But if we just want to find the ML tree, we don't need to even calculate the likelihood!

The preferred tree is the tree preferred by the maximum compatibility criterion for phylogenies. This result holds for trees with any number of taxa, and any number of states, though the formulae are more complex. See chapter 8 of Felsenstein's book for a nice description of the compatibility approach to tree inference.

### Finding the ML tree

We could find the tree with the highest likelihood by examining every tree. This is not practical for even moderate (10-20) numbers of taxa because there are far too many trees to consider (we'll discuss this later).

In the case of finding the maximum compatibility tree, though, we can take a different approach. We can construct a compatibility graph. Consider a graph with nodes representing every pattern that occurs in the character matrix. Each node has a weight that is equal to the number of times that pattern was seen in the data matrix. If two patterns are compatible with each other, then we introduce an edge in our graph between the corresponding nodes.

We can also view the nodes in the graph as representing splits in the tree (because of the one-to-one mapping between splits and binary characters that we talked about last time).

**The pairwise compatibility theorem** says:
We have a collection of splits, the entire collection of splits can be fit onto one tree if and only if every possible pair of splits in the set are pairwise compatible.

So if we look in the compatibility graph and find a subgraph for which there is an edge between every pair of nodes, then all of the splits represented by that subgraph can be put onto the same tree. Such subgraph is called a clique. If we consider the weight of a clique to be the sum of the counts associated with the nodes in the clique, then finding the clique with the highest weight corresponds to finding the largest set of compatible characters that can be mapped onto a single tree. This is tree will be the maximum compatibility tree. See Felsenstein (2004) (Chapter 8), (Buneman, 1971) and (Estabrook and McMorris, 1980) for details and proofs.

As discussed by Felsenstein, the maximal compatibility problem is NP-hard, but for phylogenetic problems, heuristic approaches often work well (the compatibility graphs in many phylogenetic problems are simple enough to find the maximum compatibility tree quickly).

Table 3: Data matrix with some character conflict

| | Character # | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Taxon | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| A | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| B | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| C | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| D | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |

Table 4: Patterns and counts for the data shown in table 3

| | Character | | | | | |
|---|---|---|---|---|---|---|
| Taxon | 1 | 2-4 | 5 | 6-9 | 10-11 | 12 |
| A | 0 | 0 | 0 | 0 | 0 | 0 |
| B | 1 | 0 | 0 | 1 | 1 | 1 |
| C | 0 | 1 | 0 | 1 | 1 | 0 |
| D | 0 | 0 | 1 | 1 | 0 | 1 |
| count | 1 | 3 | 1 | 4 | 2 | 1 |

Figure 2: Compatibility graph for the patterns in table 4. Character numbers are inside node circles. Pattern weights are shown next to the nodes.

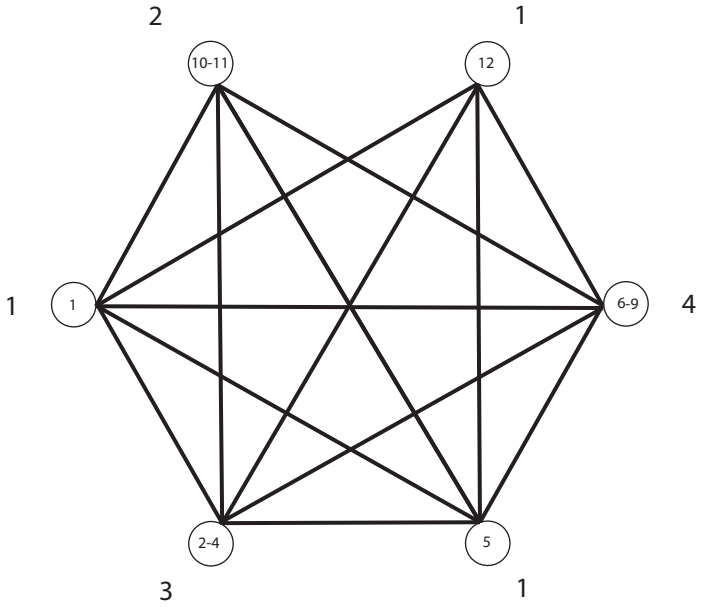
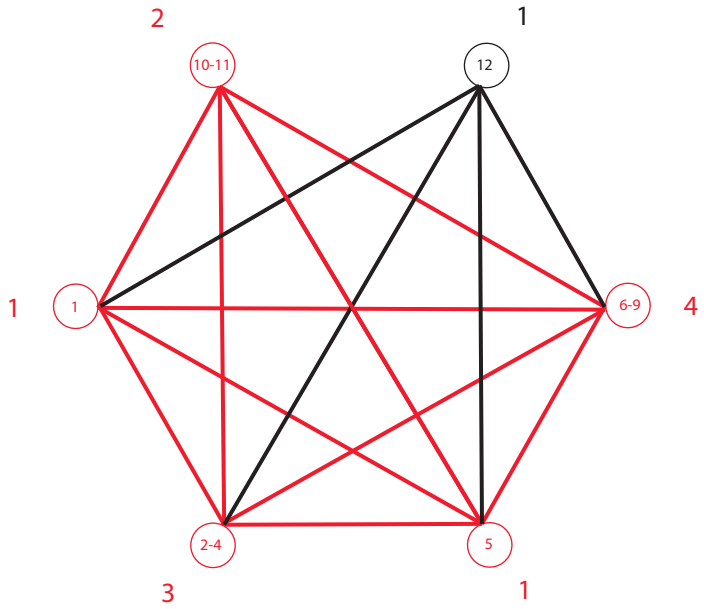


Figure 3: Compatibility graph for the patterns in table 4, with maximal weighted clique shown in red (total weight of 11).




As Felsenstein (2004) discusses, the pairwise compatibility theorem does not apply to characters with more than 2 states or in matrices with missing data – see the examples in table 5 from Felsenstein (2004) and Fitch (1975). This does not mean that you cannot use maximum compatibility as a criterion for evaluating trees. Nor does it mean that the correspondence between ML under the "perfect + noise" model and maximum compatibility will be disrupted. It merely means that we cannot (necessarily) find the maximum compatibility tree by constructing a compatibility graph of the character patterns and finding the largest weight clique.

Table 5: Two data matrices for which the pairwise compatibility theorem does not apply

| Taxon | | | |
|---|---|---|---|
| A | 0 | 0 | 0 |
| B | ? | 0 | 1 |
| C | 1 | ? | 0 |
| D | 0 | 1 | ? |
| E | 1 | 1 | 1 |

| Taxon | | | |
|---|---|---|---|
| A | 0 | 0 | 0 |
| B | 0 | 1 | 2 |
| C | 1 | 1 | 1 |
| D | 1 | 2 | 0 |
| E | 2 | 2 | 2 |

## References

Buneman, P. (1971). The recovery of trees from measures of dissimilarity. In Hodson, F. R., Kendall, D. G., and Tautu, P., editors, *Mathematics in the Archaeological and Historical Sciences*, Edinburgh. The Royal Society of London and the Academy of the Socialist Republic of Romania, Edinburgh University Press.

Estabrook, G. F. and McMorris, F. R. (1980). When is one estimate of evolutionary relationships a refinement of another? *Journal of mathematical biology*, 10:367–373.

Felsenstein, J. (2004). *Inferring Phylogenies*. Sinauer Associates, Inc, Sunderland, Massachusetts, 1 edition.

Fitch, W. M. (1975). toward finding the tree of maximum parsimony. In Estabrook, G. F., editor, *Proceedings of the Eighth International Conference on Numerical Taxonomy*, San Francisco, CA. W. H. Freeman.