

MIT OpenCourseWare  
<http://ocw.mit.edu>

6.047 / 6.878 Computational Biology: Genomes, Networks, Evolution  
Fall 2008

For information about citing these materials or our Terms of Use, visit: <http://ocw.mit.edu/terms>.

# 6.047/6.878 Lecture 10

## Molecular Evolution and Phylogenetics

October 8, 2008

### Notes from Lecture

- Error in slide 13:  $N_{substitutions} < N_{mutations}$
- Next 2 lectures given by guest speakers
- The section on Maximum Likelihood Methods is left for recitation

## 1 Motivations

Evolution by natural selection is responsible for the divergence of species populations through three primary mechanisms: populations being altered over evolutionary time and speciating into separate branches, hybridization of two previously distinct species into one, or termination by extinction. Given the vastness of time elapsed since life first emerged on this planet, many distinct species have evolved which are all related to one another; phylogenetics is the study of evolutionary relatedness among species and populations. Traditional phylogeny asks how do species evolve and before the advent of genomic data mostly relied on physiological data (bone structure from fossils, etc). We are interested in tackling phylogenetics from a different perspective; analyzing DNA sequence data in order to determine relationships between and among species. At the core, we would like to detect evidence of natural selection in populations. This is an increasingly important area of research in computational biology and is starting to find commercial applications in the realm of personal genomics: it was recently announced that a joint MIT & Harvard affiliated company was established to sequence individual genomes for \$5000 (other private companies including “23 and me,” “deCODEme,” are already doing this.

We will formulate this biological problem in computational terms by studying two probabilistic models of divergence: Jukes-Cantor & Kimura. Two purely algorithmic approaches (UPGMA & Neighbor-Joining) will be introduced to build species or gene trees from these relatedness data (the distinction between the species & gene trees is explained below).

Among the many open problems in phylogenetics that we can currently address with genomics are how similar two species are, what migration paths early humans took when they first left the African continent by studying variations in identical genes of a number of local tribes around the world (The National Geographic Project is one such example), and determining our closest living cousins (chimpanzees or gorillas?), among many others. Many open questions in evolutionary biology have already been answered by genomic phylogenetics (a major recent one being the revelation that the closest living relative of the whale is the hippopotamus).

Information in phylogenetics is best represented using trees that succinctly show relationships among species or genes. There are a number of important issues regarding inferring phylogenies when modeling evolution with trees that one should be aware of:

- nodes linking branches (exact type of common ancestors)
- meaning of branch lengths (time scaled or not)
- type of splitting event at branches (usually binary)

As an aside relating to the last bulleted point, Professor Pavel Pevzner of UCSD (author of our class textbook on Bioinformatic Algorithms) mentioned in his recent talk at MIT that the order of convergence (are humans closer to dogs or mouse?) may require a trifurcation model (split in three ways) instead.

It is important to note that gene & species divergence are two distinct events. The same gene (or slight variations thereof) can be found in different species (organisms that cannot interbreed). To think about it another way, a species tree is a special case of a gene tree, consisting of an orthologous sequence of the same common gene. Moreover, in a species tree, we can have gene flow between the different branches of the tree. If every “leaf” is an organism, it is a species tree. A gene tree captures both speciation & duplication events with the length between the root and the various leaves of the tree being a measure of the number of mutations between the two. The level of tree complexity (branching lengths & number) dictates what types of algorithms to use. In this class we focus on sequence comparison to develop gene & species trees.

There are many advantages of using genomes in phylogenetics. A major one being the vast amount of information we have access to: consider for a moment that for every position in the genome, particularly individual amino acid positions within the protein sequence, lies a corresponding trait. A small number of traits is usually used to assemble species trees, for instance in traditionally phylogenetics (before the advent of genomic data), one could compare the panda skeletal structure with that of bears & raccoons. The underlying premise in creating trees from traits is the parsimony principle: find a tree that best explains the set of traits in a *minimum* number of changes. Unfortunately, there are also a number of complications having to do with the fact that traits are typically ill behaved: back mutations are frequent (nails becoming short, then long and finally back to short), inaccessibility to ancestral sequences and difficulties in correlating substitution rate with time. With the regards to back mutations, even though evolution is traditionally thought of as divergent always acting to increase entropy, there are rare instances of convergent evolution (or homoplasy). Homoplasy is the phenomenon where two separate lineages, completely independent of one another, undergo the same changes that lead to their convergence. It is a strictly random process that has not infrequently been observed in nature.

To take the human population as an example, the mutation rate within the human genome since our species left Africa is comparable to a phylogenetic event. The mutations are infrequent, roughly 1000 mutations (single nucleotide polymorphisms or SNPs) in the total 3 billion nucleotide genome. This is why producing a map of the human family tree can even be undertaken given the manageability of such complexity.

How are genes produced? There are two main mechanisms: 1) duplication: new versions of old genes (most frequent) and 2) de novo genes: intronic (non-coding) segments or regulatory connection of sequences become coding (hi-jacking of nonfunctional nucleotides does happen but at a much lower rate).

We will study three types of trees: cladogram, phylogram and ultrametric. The three kinds of trees associate different meanings to the branch lengths, in order they are: no meaning, genetic change and elapsed time. Species with much shorter lifespans and higher reproduction rates will tend to show much larger genetic changes (cf. mouse and human genes). Tree building methods can be constructed by decoupling the notion of the algorithm used and the definition of the optimality condition. We will need to think carefully about what we choose as an optimality condition and how well an algorithm reaches the optimality condition. As an example, clustering algorithms do not have an optimality criteria (they don't maximize/minimize an objective function). In computational biology, we are faced with a dilemma: whether to design an algorithm that does the right thing or whether to design a more suitable model (e.g., how frequently changes observed in a sequence can be used to infer a distance to another sequence).

## 2 Modeling Evolution

We need a method to measure evolutionary rates so that a distance matrix can be constructed before we can develop a tree. A distance matrix will allow us to turn a set of sequences into a set of pairwise distances between sequences. Let's focus on two types of single nucleotide mutations: transitions (A  $\leftrightarrow$  G, C  $\leftrightarrow$  T) or transversions (A  $\leftrightarrow$  T, G  $\leftrightarrow$  C) happening one at a time. We consider two stationary Markov models represented by a nucleotide substitute matrix that assume nucleotides evolve independently of one another.

The Jukes-Cantor approach assumes a constant rate of evolution assigning one rate to self-mutation (A  $\rightarrow$  A, etc) and another to cross mutation (A  $\rightarrow$  one of C,G,T). The Jukes-Cantor AGCT substitution matrix:

$$S = \begin{pmatrix} r & s & s & s \\ s & r & s & s \\ s & s & r & s \\ s & s & s & r \end{pmatrix}$$

For short times, the evolutionary rate is constant:  $r = 1 - 3\alpha\epsilon$  and  $s = \alpha\epsilon$ . For longer times, the rate is a function of time:  $r = 0.25(1 + 3e^{-4\alpha t})$  and  $s = 0.25(1 - e^{-4\alpha t})$ . The Kimura model goes one further and takes into account that transitions are more frequent than transversions. The Kimura AGCT substitution matrix:

$$S = \begin{pmatrix} r & s & u & u \\ s & r & u & u \\ u & u & r & s \\ u & u & s & r \end{pmatrix}$$

where  $s = 0.25(1 - e^{-4\beta t})$ ,  $u = 0.25(1 + e^{-4\beta t} - e^{-2(\alpha+\beta)t})$ , and  $r = 1 - 2s - u$ .

## 3 From distances to trees

Given our generative Markov models and corresponding (time-dependent) substitution matrices, we are now ready to determine a distance matrix. The elements of this distance matrix  $d_{ij}$  are a measure of the distance of two well aligned sequences. One possibility is to define the distance

matrix  $(d_{ij})$  as the fraction of sites ( $f$ ) where two sequences  $x_i$  and  $x_j$  are mismatched:  $d_{ij} = -\frac{3}{4}\log(1 - 4f/3)$ . This model blows up when  $f = 0.75$  which places a threshold on the amount of mismatch between two sequences.

To then use this distance matrix to measure actual distances between any pair of sequences (ie. to construct a tree), there are two standard approaches: 1) ultrametric distances infer equidistant paths from any leaf to the root while 2) additive distances dictate that all pairwise distances are obtained by traversing a tree. Ultrametric trees are rarely valid since it assumes uniform rate of evolution while additive distances are a less restrictive model. In practice, distance matrices is neither ultrametric nor additive.

The distance matrix and tree duality also means that distances can be inferred by traversing tree. If ultrametric distances are used, we have a guarantee of finding the correct tree by minimizing the discrepancy between observed distances and tree-based distances. On the other hand, if additive distances are used, neighbors are guaranteed to find the correct tree. The tree building algorithm can then be thought of fitting data to constraints.

## 4 Tree building algorithms

### 4.1 UPGMA

UPGMA (Unweighted Pair Group Method with Arithmetic Mean) is the simplest example of a tree-building algorithm. It involves a hierarchical clustering algorithm that begins at the leaves of the tree and works it way up to the root. As input, it takes a distance matrix and produces an ultrametric tree (ie. in accordance with the molecular clock hypothesis of equal evolutionary rates among species). Only when the input distance matrix is ultrametric is the UPGMA algorithm guaranteed to form the correct tree. If, on the other hand, the distance matrix is additive, there is no guarantee that the pairwise branch distances equal those specified in the distance matrix.

ALGORITHM:

1. Begin by initializing all sequences each to its own cluster. These will form the leaves of the tree.
2. Find the pair of sequences with minimum distance from the distance matrix  $D$ . This pair forms the first cluster, and we can draw the first part of the tree by linking the pair. (example: from  $D$ , we find  $seqA$  and  $seqB$  have the minimum distance of 10. We draw the tree by linking  $seqA$  and  $seqB$ , with branch length of 5 (hence a total distance of 10 between them)).
3. Update matrix  $D$ : A new row and column representing  $seq_{AB}$  is added to  $D$ . The distance between  $seq_{AB}$  and another  $seq_C$  would be  $\frac{1}{2}(d_{AC} + d_{BC})$ . Remove the rows and columns associated with  $seqA$  and  $seqB$ . Note, overall, the matrix has shrunk by 1 row and 1 column. From this point on, we can completely forget about  $seqA$ ,  $seqB$ , and assume we just have  $seq_{AB}$ .
4. Repeat steps 2 and 2 until matrix  $D$  is empty

## 4.2 Neighbor-Joining

The more sophisticated N-J algorithm is used to generate phylogram trees in which branch lengths are proportional to evolutionary rates. The N-J algorithm is guaranteed to produce the correct tree if the input distance matrix is additive and may still produce a good tree even when the distance matrix is not additive.

ALGORITHM:

1. Make a new matrix  $M$  from the distance matrix  $D$  with the same dimensions:

$$M_{ij} = D_{ij} - \frac{\sum_k D_{ik} + D_{jk}}{N - 2}$$

where  $N$  is the number of sequences. This is the adjusted distance metric, which ensures  $M_{ij}$  is minimal iff  $i$  and  $j$  are neighbors (see proof in Durbin, p.189).

2. (similar to UPGMA Step 2) Find the pair of sequences with minimum distance from the new distance matrix  $M$ . This pair forms the first cluster, and we can draw the first part of the tree by linking the pair. (example: from  $M$ , we find  $seqA$  and  $seqB$  have the minimum distance. We draw the tree by linking  $seqA$  and  $seqB$ , via new node  $U$ . The branch length from A to U is calculated as:  $D_{AU} = \frac{1}{2} \left( D_{AB} + \frac{\sum_k D_{Ak} + D_{Bk}}{N-2} \right)$ . Also,  $D_{BU} = D_{AB} - D_{AU}$ ).
3. (similar to UPGMA Step 3) Update matrix  $D$ . A new row and column representing node  $U$  is added to  $D$ . The distance between  $U$  and another  $seqC$  would be  $\frac{1}{2} (d_{AC} + d_{BC} - d_{AB})$ . Remove the rows and columns associated with  $seqA$  and  $seqB$ . Note, overall, the matrix has shrunk by 1 row and 1 column. From this point on, we can completely forget about  $seqA$  and  $seqB$ , and assume we just have node  $U$ .
4. Repeat steps 1 thru 3 until matrix D is empty.

## 4.3 Parsimony

Another approach to building trees is parsimony; a method that doesn't rely on distance matrices but instead on sequence alignments. Parsimony will find the tree that explains the observed sequences with the minimal number of substitutions. The algorithm entails two computational sub-problems: 1) finding the parsimony cost of a given tree and 2) searching through all tree topologies. The first sub-problem is straightforward while the second is computationally very challenging and best approached with Monte Carlo methods. As there is no closed form solution and hence no optimality criteria while searching through all topologies, a heuristic search is appropriate. Parsimony uses dynamic programming in scoring and traceback to determine ancestral nucleotides.

### 4.3.1 Scoring

The parsimony scoring algorithm is akin to dynamic programming as it makes local assignments at each step: penalizing sequence mismatches while assigning no score for sequence matches.

- Initialization: Set cost  $C = 0$ ;  $k = 2N - 1$

- Iteration:

If  $k$  is a leaf, set  $R_k = \{x^k[u]\}$

If  $k$  is not a leaf,

Let  $i, j$  be the daughter nodes;

Set  $R_k = R_i \cap R_j$  if intersection is nonempty

Set  $R_k = R_i \cup R_j$ , and  $C += 1$ , if intersection is empty

- Termination: Minimal cost of tree for column  $u = C$

### 4.3.2 Traceback

The traceback routine to find ancestral nucleotides involves finding a path through the tree from the leaf nodes to an ancestral nucleotide (that may or may not lead to the root of the tree). It can be summarized as follows:

- If the intersection of two sets (A and B) is empty, ancestor is A or B with equal cost
- If the intersection of two sets is non-empty, ancestor is intersection with minimal cost

This method determines a non-unique path with minimum assignment of substitutions for a given tree topology given hidden internal/intermediate nodes (that may correspond to extinct species).

### 4.3.3 Bootstrapping

A shortcut to building trees based on parsimony is to build trees using only one column of the multiple sequence alignment matrix. If repeated many times and a histogram tabulated of the particular trees constructed, the most likely tree constructed can be easily found. The advantage to such an approach is obvious: reduced complexity at the expense of accuracy.

## 4.4 Maximum Likelihood

Maximum likelihood methods combine statistical models with known evolutionary observed data. They can be used to predict a host of interesting and realistic features - character and rate analysis, sequences of (hypothetical) extinct species - but do so at the expense of increased complexity.