Assignment 7

In this assignment, the task is to obtain the maximum likelihood of a tree (topology) and (optionally) to search for the maximum likelihood tree (topology). We will examine the simplest possible case, the Jukes Cantor model, in which there is no substitution model parameters to optimize. Thus, we need only optimize branch lengths to get the maximum likelihood of a tree (topology).

To optimize the branch lengths, we cycle through the branches in the tree and optimize one branch at a time. When we have optimized all branches once, we start over from the first branch, and continue cycling through the tree until the likelihood score does not improve significantly.

Optimizing one branch length is easy for the Jukes Cantor model. First, the conditional likelihoods are pulled towards the two end points of the branch from the two subtrees on each side of the branch. Then the maximum likelihood branch length, $\hat{v}$ conditional on the likelihoods at the two ends of the branch, can be obtained by a simple formula,

$$\hat{v} = -\frac{3}{4}\ln\left(1 - \frac{4}{3}\bar{d}\right)$$

where $\bar{d}$ is the average distance, per site, over the branch. The average distance is calculated from the conditional likelihoods at the two end points, i and j, of the branch:

$$\bar{d} = \frac{1}{N}\sum_{c}\left(1 - \sum_{k} L_i(c,k)L_j(c,k)\right)$$

In this formula, $L_i(c,k)$ is the conditional likelihood at branch endpoint i of state k of character c; $L_j(c,k)$ denotes the corresponding conditional likelihood at the other endpoint of the branch. The total number of characters is assumed to be N.

The only thing we need to know now is the method for obtaining the conditional likelihoods. Perhaps the simplest solution is to use the likelihood downpass algorithm on each of the two subtrees, as if they were rooted on the nodes connected to the endpoints of the branch we are interested in. An alternative is to calculate the final pass conditional likelihoods and then subtract from the final likelihoods at the bottom node the conditional likelihoods coming into that node from the branch we are interested in. A third solution is to calculate each of the three possible views from each node in the tree, as described by Felsenstein on page 17 of his book.

If you want to use the likelihood uppass algorithm, it is given here. It is analogous to the Sankoff uppass algorithm (Algorithm 1). We use $f_i^{(p)}$ to denote the final conditional likelihood of state i for node p, and $g_i^{(p)}$ to denote the downpass conditional likelihood of that node. Furthermore, $h_i^{(p)}$

is the conditional likelihood of state i at the bottom end of the branch ending in node p. We are focusing on a node p and its ancestral node a.

Algorithm 1 Likelihood uppass algorithm

for all j **do**

$f_j^{(p)} \leftarrow \sum_i ((f_i^{(a)}/h_i^{(p)}) c_{ij} g_j^{(p)})$

end for

This is your assignment in summary:

1. Write new code to optimize the likelihood of a given tree by changing one branch length at a time in several cycles until the likelihood of the entire tree does not improve more than a certain pre-specified value. Start the procedure with some arbitrarily set branch lengths, for instance, branch lengths all set to 0.1.

2. (Optional) Combine your likelihood optimizer algorithms with code that you wrote previously to search for trees using stepwise addition.