

Tutorial 2: Tree Inference

Jonathan Palko and Willow Livengood

A dark blue diagonal gradient bar that starts from the bottom left and extends towards the top right, covering the lower half of the slide.

Phylogenetic Tree Background

Phylogenetic trees: Branching diagrams that show the evolutionary relationships among various biological species based on similarities and differences in their physical and genetic characteristics

One common form of phylogeny estimation is using maximum likelihood estimators

PhyML, IQ-Tree, FastTree and RAxML are all examples of maximum likelihood (ML) phylogenetic programs

16s rRNA H. influenzae bacteria was discovered in 1892 by Richard Pfeiffer. Argued to be the cause of influenza and was one of the first free-living organism to have its genome sequenced.

Sequence Being Aligned

Unaligned Haemophilus Influenzae 16S rRNA sequence

```
>Haemophilus_influenzae_86_028NP_uid58093_1
AATTGAAGAGTTTGATCATGGCTCAGATTGAACGCTGGCGGCAGGCTTAACACATGCAAGTCGAACGGTAGCAGGAGGAAAGCTTGCTTTCTTGCTGACGAGTGGCGGACGGGTG
>Haemophilus_influenzae_86_028NP_uid58093_2
AATTGAAGAGTTTGATCATGGCTCAGATTGAACGCTGGCGGCAGGCTTAACACATGCAAGTCGAACGGTAGCAGGAGGAAAGCTTGCTTTCTTGCTGACGAGTGGCGGACGGGTG
>Haemophilus_influenzae_86_028NP_uid58093_3
AATTGAAGAGTTTGATCATGGCTCAGATTGAACGCTGGCGGCAGGCTTAACACATGCAAGTCGAACGGTAGCAGGAGGAAAGCTTGCTTTCTTGCTGACGAGTGGCGGACGGGTG
>Haemophilus_influenzae_86_028NP_uid58093_4
AATTGAAGAGTTTGATCATGGCTCAGATTGAACGCTGGCGGCAGGCTTAACACATGCAAGTCGAACGGTAGCAGGAGGAAAGCTTGCTTTCTTGCTGACGAGTGGCGGACGGGTG
>Haemophilus_influenzae_86_028NP_uid58093_5
AATTGAAGAGTTTGATCATGGCTCAGATTGAACGCTGGCGGCAGGCTTAACACATGCAAGTCGAACGGTAGCAGGAGGAAAGCTTGCTTTCTTGCTGACGAGTGGCGGACGGGTG
>Haemophilus_influenzae_86_028NP_uid58093_6
AATTGAAGAGTTTGATCATGGCTCAGATTGAACGCTGGCGGCAGGCTTAACACATGCAAGTCGAACGGTAGCAGGAGGAAAGCTTGCTTTCTTGCTGACGAGTGGCGGACGGGTG
>Haemophilus_influenzae_F3031_uid62123_1
ACTTGAATTGAAGAGTTTGATCATGGCTCAGATTGAACGCTGGCGGCAGGCTTAACACATGCAAGTCGAACGGTAGCAGGAGGAAAGCTTGCTTTCTTGCTGACGAGTGGCGGAC
>Haemophilus_influenzae_F3031_uid62123_2
ACTTGAATTGAAGAGTTTGATCATGGCTCAGATTGAACGCTGGCGGCAGGCTTAACACATGCAAGTCGAACGGTAGCAGGAGGAAAGCTTGCTTTCTTGCTGACGAGTGGCGGAC
>Haemophilus_influenzae_F3031_uid62123_3
ACTTGAATTGAAGAGTTTGATCATGGCTCAGATTGAACGCTGGCGGCAGGCTTAACACATGCAAGTCGAACGGTAGCAGGAGGAAAGCTTGCTTTCTTGCTGACGAGTGGCGGAC
>Haemophilus_influenzae_F3031_uid62123_4
ACTTGAATTGAAGAGTTTGATCATGGCTCAGATTGAACGCTGGCGGCAGGCTTAACACATGCAAGTCGAACGGTAGCAGGAGGAAAGCTTGCTTTCTTGCTGACGAGTGGCGGAC
>Haemophilus_influenzae_F3031_uid62123_5
ACTTGAATTGAAGAGTTTGATCATGGCTCAGATTGAACGCTGGCGGCAGGCTTAACACATGCAAGTCGAACGGTAGCAGGAGGAAAGCTTGCTTTCTTGCTGACGAGTGGCGGAC
>Haemophilus_influenzae_F3031_uid62123_6
ACTTGAATTGAAGAGTTTGATCATGGCTCAGATTGAACGCTGGCGGCAGGCTTAACACATGCAAGTCGAACGGTAGCAGGAGGAAAGCTTGCTTTCTTGCTGACGAGTGGCGGAC
>Haemophilus_influenzae_F3047_uid62097_1
ACTTGAATTGAAGAGTTTGATCATGGCTCAGATTGAACGCTGGCGGCAGGCTTAACACATGCAAGTCGAACGGTAGCAGGAGGAAAGCTTGCTTTCTTGCTGACGAGTGGCGGAC
>Haemophilus_influenzae_F3047_uid62097_2
ACTTGAATTGAAGAGTTTGATCATGGCTCAGATTGAACGCTGGCGGCAGGCTTAACACATGCAAGTCGAACGGTAGCAGGAGGAAAGCTTGCTTTCTTGCTGACGAGTGGCGGAC
ACTTGAATTGAAGAGTTTGATCATGGCTCAGATTGAACGCTGGCGGCAGGCTTAACACATGCAAGTCGAACGGTAGCAGGAGGAAAGCTTGCTTTCTTGCTGACGAGTGGCGGAC
```

MAFFT Aligner Background

Multiple Alignment Using Fast Fourier Transform
(MAFFT) is a multiple sequence alignment program.

Organizes DNA or RNA into a matrix where each nucleotide sequence is represented as a row, columns represent the nucleobases, and gaps are inserted such that identical characters are aligned.

Used to identify regions of similarity and produce an analysis on evolutionary relationships and constructing phylogenetic trees.

Aligned Sequence

MAFFT Aligned Haemophilus Influenzae 16S rRNA sequence

```
>Haemophilus_influenzae_86_028NP_uid58093_1
-----aattgaag-----agt
ttgatcatggctcagat----tgacgctggcggcaggcttaacacatgcaagtcgaac
ggtagcaggaggaagcttgcttcttgctgacgagtgccggacgg-gtgagtaatgcttg
ggaatctggcttatgga-gggggataacgacgggaaactgtcgctaataccg----cgta
ttatcgga----agatgaaagtgcgggactgagaggccgcatgccataggatgagccaa
gtggg-----att
aggtagttggtggggtaaatgcctaccaagcctgc-----gatctctagctggctgga
ggagatgaccagccacactgggaactgagacacggctccagact-cctacgggaggcagcag
tggggaatatgtgcgaatggggggaaacctgacgcagccatgccgctgaatgaagaagg
ccttcgggttgtaaaagtcttctcggtattgaggaaaggtgatgtgtaatagacacatcaa
a---tgacgttaaatacagaagaagcaccggct-aactccgtgccagcagccggttaa
tacggagggtgagcgttaatcggaataactgggcgtaaagggcacgcaggcgggttatt
taagtgggtgtgaaagccccgggcttaacctgggaattgcatttcagactgggtaacta
gagtacttttagggaggggtagaattccacgtgtagcggtgaaat--gcgtagagatgtgg
aggaataccgaaggcgaaggcagcccttgggaatgtactgacgctcatgtgcgaagcgc
tggggagcaaacaggattagataccctggtagtccacgctgtaaacgctgt-----c
gatttgggggttggggttaactctggcgccttagctaacgtgataaatgcagccctg
gggagtacggcgg----caagggttaaaactcaaa----tgaattgacggggggcccgca
caagcgggtggagcatgtggtttaaattcgatgcaacgcgaagaaccttacctactcttgac
atcctaa----gaagagctcagagatgagcttgtgccttcgggaacttagagacaggtgc
tgcatggctgtcgtcag---ctcgtgtgtgaaatgtt----gggttaagtcccgcaa
-----cgagcgcaacccttatccttgttgccagcagactgtgtcgggaactcaaaaggag
actgccagtgataaaactggaggaaggtggggatgacgtcaagtcacatgccccttacga
gtagggtcacacacgtgctacaatggcggtatacagaggggaagcgaagctgcgaggtggag
cgaatctcataaagtacg---tctaagtcggattggagctgcaactcgactccatga
agtcggaatcgctagtaatcgcaatcagaatgtcgggtgaa---tacgttccggggcc
ttgtacacaccgccctcacaccatgggagtggttgtagcagaagtagatagcttaacc
ttttggaggcggttaccacggtatgattcatgactggggtgaagtcgtaacaaggtaac
cgtagggg---aacctgcgggttgatcacctccta-----
>Haemophilus_influenzae_86_028NP_uid58093_2
-----aattgaag-----aet
```

RaxML Tree Details

```
RaxML(tree t, int rStart, int rMax)
{
    int rL, rU;
    boolean a = TRUE;
    boolean impr = TRUE;

    while(TRUE)
    {
        if(impr)
        {
            rL = 1;
            rU = rStart;
            rearr(t, rL, rU, a);
        }
        else
        {
            if(!a)
            {
                a = FALSE;
                rL = 1;
                rU = rStart;
            }
            else
            {
                rL += rStart;
                rU += rStart;
            }
            if(rU < rMax)
                rearr(t, rL, rU, a);
            else
                goto end;
        }
        impr = optimizeList20();
    }
    end:
}
```

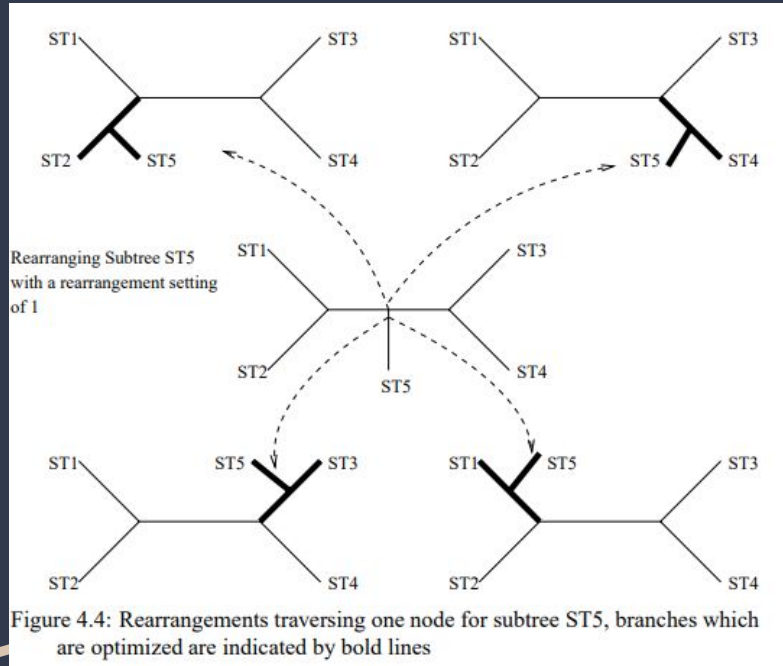
Randomized Axelerated Maximum Likelihood (RaxML) uses sequential and parallel Maximum Likelihood inference for large phylogenetic trees, with its major strength being a fast maximum likelihood tree search algorithm that returns good likelihood scores

Created by Alexandros Stamatakis, Heidelberg Institute for Theoretical Studies

Originally derived from fastDNaml and Joe Felsenstein's dnaml from the PHYLIP package

RAXML-HPC v.8 on XSEDE specifically has four different bootstrapping methods for confidence values, support for more input data types, fine-grain parallelization, and additional post-analysis methods for statistical significance tests, and optimizations to computational power use and memory

RaxML Heuristics



1. Build initial parsimony tree with dnaphars (Felstein's PHYLIP package)
 - a. Parsimony is related to maximum likelihood under simple evolutionary models
 - b. Dnaphars use stepwise addition for tree building and are relatively fast
 - c. Strong starting tree increases chances of good likelihood value compared to random or neighbor starting trees
 - d. Enables the construction of distinct starting trees using a randomized sequence order
2. Tree optimization process
 - a. Standard subtree rearrangements by removing all possible subtrees from current best tree
 - b. Reinsert subtrees into neighboring branches at specified distance nodes
 - c. Inherited from fastDNAm1, but RaxML only optimizes three local branches adjacent to subtree or Newton-Raphson method
 - d. Compute maximum likelihood estimation
 - e. Store 20 best trees obtained during one rearrangement step

RaxML Heuristics

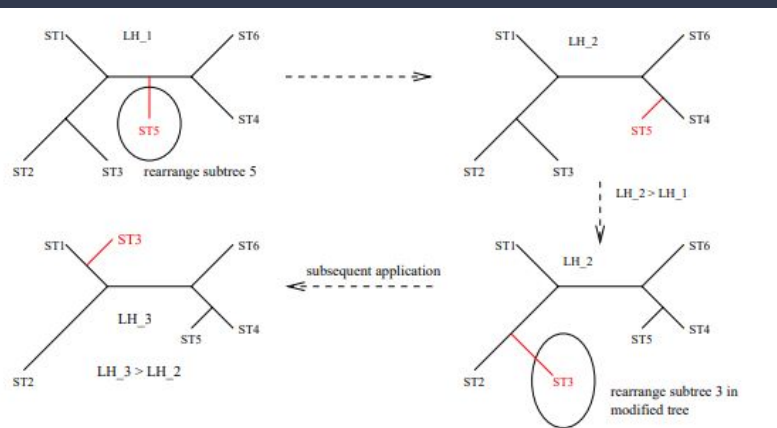


Figure 4.6: Example for subsequent application of topological improvements during one rearrangement step

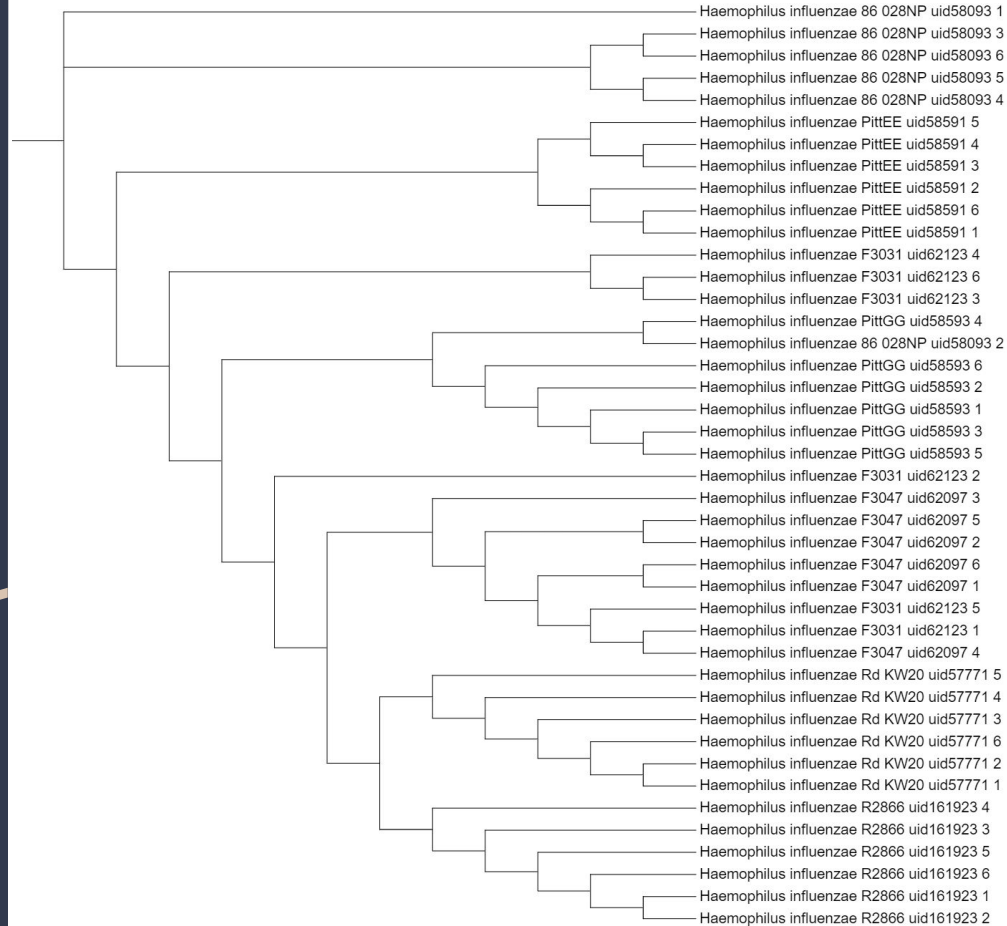
1. Global Branch Length Optimizations
 - a. Only on 20 best topologies
 - b. Capable of rapidly analyzing more alternative and diverse topologies due to higher rearrangement setting of 20 (compared to 5 or 10) leads to significantly higher trees
2. Subsequent application of topological improvements may occur during one rearrangement step
3. If a subtree inserted into an alternate branch has a higher likelihood than the best tree is encountered this tree is kept and all subsequent rearrangements are performed on this topology
 - a. Enables rapid optimization for random starting trees
4. Leverages PThreads for parallelization of likelihood calculations

RAxML-HPC v.8 Generated Tree For MAFFT Aligned Data

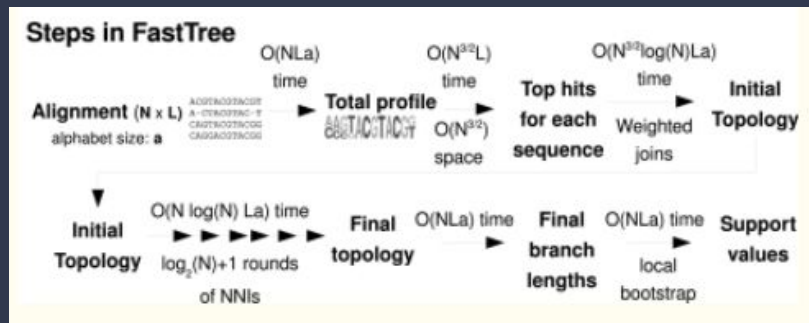
(Uses iOT, Interactive Tree of Life)

Branch lengths were ignored, due
to outlier branch distance of
“Haemophilus influenzae F3047
uid62097 4”
distorting the generated tree

The distances **should** represent
the evolutionary differences
between species



General FastTree Details



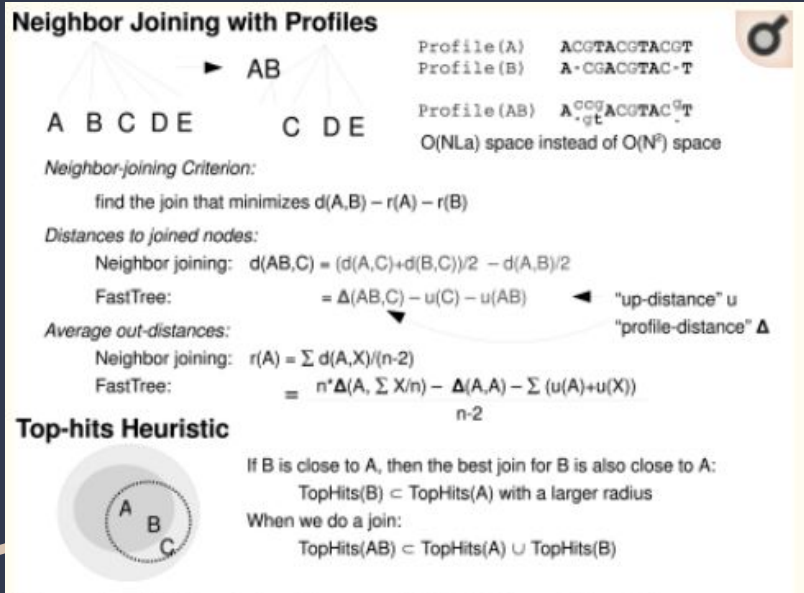
FastTree has several different stages to operation [7]

1. Heuristic Neighbor-Joining to obtain a rough topology
2. Minimum Evolution to reduce the length of the tree
3. ML-based NNI rearrangement methods to improve the topology and branch lengths with ML rearrangements
4. Reliability estimation using local support values

FastTree is many times faster than RAxML and PhyML when dealing with larger alignments

FastTree – Heuristic Neighbor-Joining

Neighbor Joining with Profiles



Profile(A) **ACGTCGTACGT**
 Profile(B) **A·CGACGTAC·T**
 Profile(AB) **A^{CGT}·ACGTAC^{GT}·T**
 O(NLa) space instead of O(N²) space

Neighbor-joining Criterion:
 find the join that minimizes $d(A,B) - r(A) - r(B)$

Distances to joined nodes:
 Neighbor joining: $d(AB,C) = (d(A,C) + d(B,C))/2 - d(A,B)/2$
 FastTree: $= \Delta(AB,C) - u(C) - u(AB)$ (where u is "up-distance" and Δ is "profile-distance")

Average out-distances:
 Neighbor joining: $r(A) = \sum d(A,X)/(n-2)$
 FastTree: $= n \cdot \Delta(A, \sum X/n) - \Delta(A,A) - \sum (u(A) + u(X)) / (n-2)$

Top-hits Heuristic

If B is close to A, then the best join for B is also close to A:
 $\text{TopHits}(B) \subset \text{TopHits}(A)$ with a larger radius
 When we do a join:
 $\text{TopHits}(AB) \subset \text{TopHits}(A) \cup \text{TopHits}(B)$

Neighbor-Joining: A clustering algorithm that clusters haplotypes based on genetic distance. Quick, but not very reliable, especially with deeper divergence times. [6]

1. FastTree stores profiles of internal nodes instead of a distance matrix, during this step, which reduces required memory
2. FastTree then uses three heuristics to speed up this phase
 - a. Remembers best join of each node
 - b. Does hill-climbing searches for better joins
 - c. Uses "top hits" heuristic to avoid computing pairwise distances and to avoid considering all possible joins at every step

FastTree updates the best join for a node as they appear, in order to reduce hill-climbing

FastTree's neighbor-joining phase does not correct the distances for multiple substitutions, which makes long-branch attraction worse

FastTree– Minimum Evolution

Nearest-neighbor interchanges (NNIs) & Local bootstrap

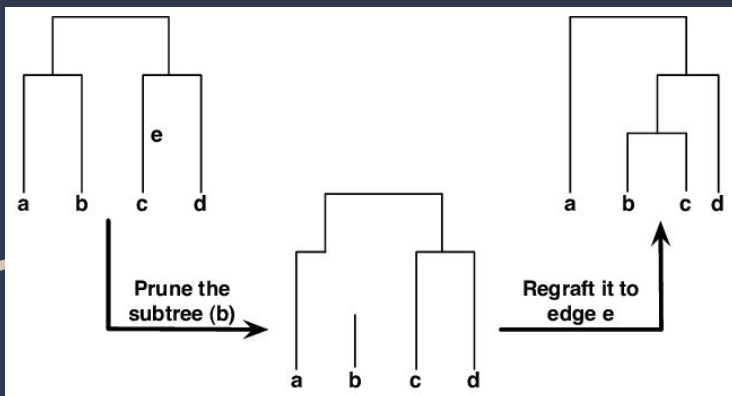
Consider three
local topologies:



Minimum evolution criterion:

prefer AB|CD if $d_{AB} + d_{CD} < \min(d_{AC} + d_{BD}, d_{AD} + d_{BC})$

where $d_{AB} = \text{LogCorrection}(\Delta(A,B))$



Minimum Evolution: An algorithm that tries to minimize branch lengths by minimizing distance (minimum evolution) or minimizing the number of mutations (maximum parsimony) [6]

Meant to address the aggravation of long-branch attraction, resulting from the previous FastTree stage and reduce the length of the tree using the following:

- Nearest-neighbor interchanges (NNIs)
 - Default setting for FastTree is $4\log_2(N)$ rounds of NNIs
- Subtree-prune-regraft moves (SPRs)
 - Default setting for FastTree is 2 rounds of SPRs

Each round every possible NNI in the tree is considered.

FastTree treats SPR moves as chains of NNIs and only extends the best choice in the chain for chain lengths of two or more. This is because there are too many possible SPR moves, ($O(N^2)$).

As long as the distances are not too noisy and the NNI and SPR moves will be able to reach optimal trees within a minimum-evolution framework

FastTree – Distances

The BLOSUM45 Matrix

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
A	5	-2	-1	-2	-1	-1	-1	0	-2	-1	-1	-1	-1	-2	-1	1	0	-2	-2	0
R	-2	7	0	-1	-3	1	0	-2	0	-3	-2	3	-1	-2	-2	-1	-1	-2	-1	-2
N	-1	0	6	2	-2	0	0	0	1	-2	-3	0	-2	-2	-2	1	0	-4	-2	-3
D	-2	-1	2	7	-3	0	2	-1	0	-4	-3	0	-3	-4	-1	0	-1	-4	-2	-3
C	-1	-3	-2	-3	12	-3	-3	-3	-3	-3	-2	-3	-2	-2	-4	-1	-1	-5	-3	-1
Q	-1	1	0	0	-3	6	2	-2	1	-2	-2	1	0	-4	-1	0	-1	-2	-1	-3
E	-1	0	0	2	-3	2	6	-2	0	-3	-2	1	-2	-3	0	0	-1	-3	-2	-3
G	0	-2	0	-1	-3	-2	-2	7	-2	-4	-3	-2	-2	-3	-2	0	-2	-2	-3	-3
H	-2	0	1	0	-3	1	0	-2	10	-3	-2	-1	0	-2	-2	-1	-2	-3	2	-3
I	-1	-3	-2	-4	-3	-2	-3	-4	-3	5	2	-3	2	0	-2	-2	-1	-2	0	3
L	-1	-2	-3	-3	-2	-2	-2	-3	-2	2	5	-3	2	1	-3	-3	-1	-2	0	1
K	-1	3	0	0	-3	1	1	-2	-1	-3	-3	5	-1	-3	-1	-1	-1	-2	-1	-2
M	-1	-1	-2	-3	-2	0	-2	-2	0	2	2	-1	6	0	-2	-2	-1	-2	0	1
F	-2	-2	-2	-4	-2	-4	-3	-3	-2	0	1	-3	0	8	-3	-2	-1	1	3	0
P	-1	-2	-2	-1	-4	-1	0	-2	-2	-2	-3	-1	-2	-3	9	-1	-1	-3	-3	-3
S	1	-1	1	0	-1	0	0	0	-1	-2	-3	-1	-2	-2	-1	4	2	-4	-2	-1
T	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-1	-1	2	5	-3	-1	0
W	-2	-2	-4	-4	-5	-2	-3	-2	-3	-2	-2	-2	-2	1	-3	-4	-3	15	3	-3
Y	-2	-1	-2	-2	-3	-1	-2	-3	2	0	0	-1	0	3	-3	-2	-1	3	8	-1
V	0	-2	-3	-3	-1	-3	-3	-3	-3	3	1	-2	1	0	-3	-1	0	-3	-1	5

In previous minimum evolution step FastTree had to estimate distances between sequences/profiles

FastTree estimates the distances for amino acid sequences as follows:

1. Uses the BLOSUM45 amino acid similarity matrix,
2. Corrects for multiple substitutions using $-1.3\log(1-d)$ where d is weighted so random sequences have an average value of 1

FastTree estimates the distances for nucleotide sequences as follows:

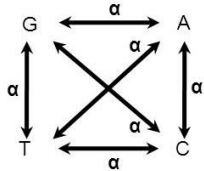
1. Uses the Jukes-Cantor distance formula, $-0.75\log(1-d*(4/3))$, where d is the proportion of positions that differ

Positions and gaps are ignored when comparing sequences, while positions are weighted by their proportions of non-gaps when comparing profiles

FastTree – ML Based Rearrangements

Jukes-Cantor Model

- Assumes that each nucleotide is equally likely to change into any other nucleotide with *probability* α per time step



	A	T	C	G
A	φ	α	α	α
T	α	φ	α	α
C	α	α	φ	α
G	α	α	α	φ

$$\varphi = 1 - 3\alpha$$

Maximum-likelihood rearrangements meant to improve the topology and branch lengths of the previously generated tree

The following ML-rearrangement methods are used by FastTree

- Jukes-Cantor model of nucleotide evolution [8]
- Generalized time-reversible model of nucleotide evolution
- Jones-Taylor-Thorton (JTT) model of amino acid evolution
- Whelan Goldman (WAG) model of amino acid evolution

FastTree takes into account the variable rates of evolution across sites, by default, through the assignment of each site to one of 20 categories that have rates geometrically spaced from 0.05 to 20 and by CAT approximation

FastTree sets each site to its most likely category via a Bayesian approach with a gamma prior, which prevents overfitting on small alignments

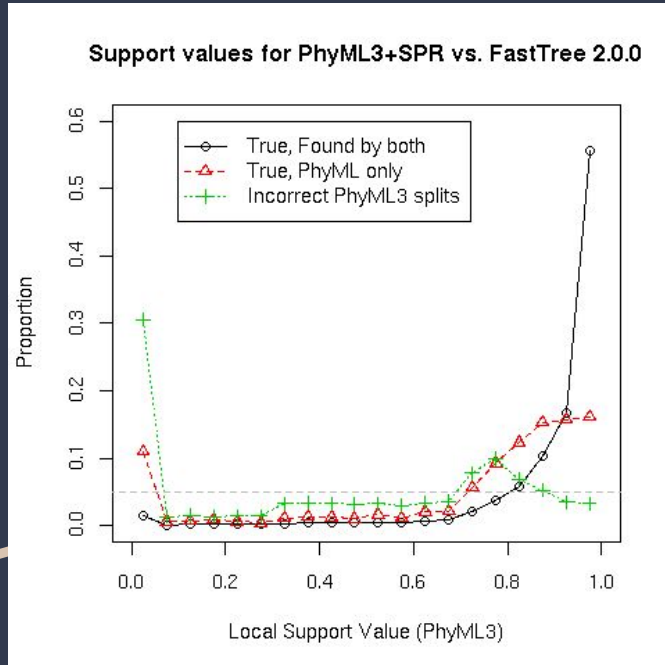
FastTree – Local Support Values

Local support values are given as names for internal nodes and range from 0 to 1

Meant to quickly estimate reliability of each split in the tree, and verify that they are correct splits

FastTree uses the Shimodaira-Hasegawa (SH) test on the NNIs around each split

Despite their differences, the resulting support values for the FastTree SH local supports are similar to local support estimates in other estimators like PhyML 3



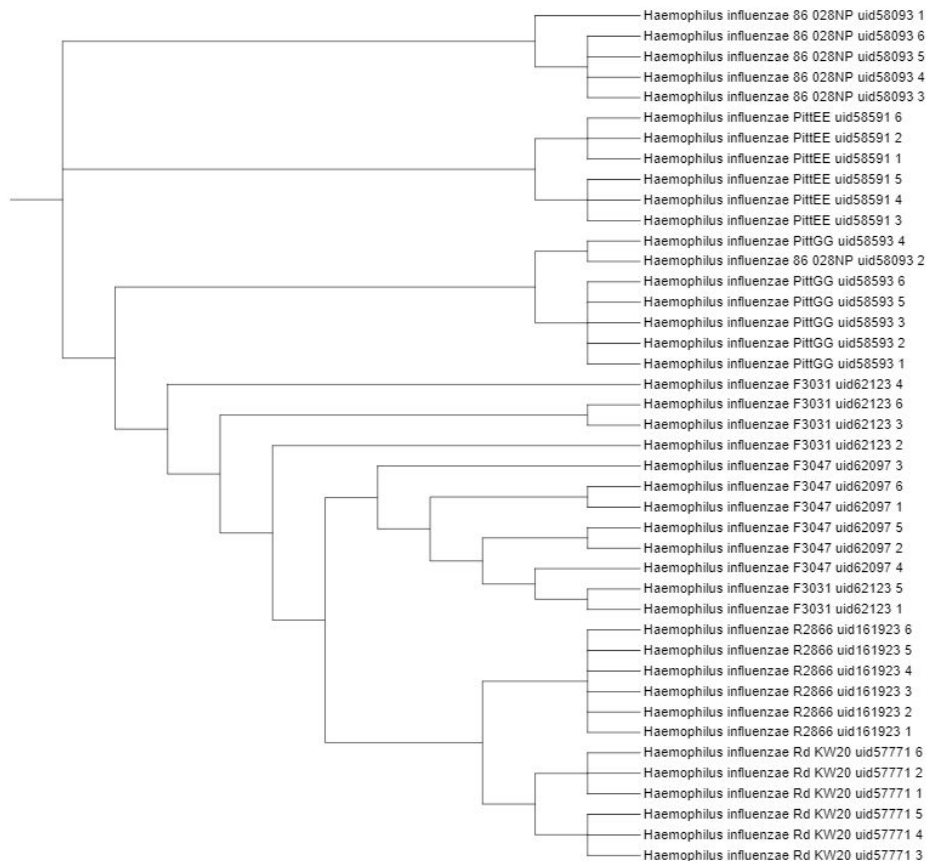
FastTree Generated Tree For MAFFT Aligned Data

(Uses iOT, Interactive Tree of Life)

Branch lengths were ignored, due to outlier branch distances greatly distorting the generated tree

The distances **should** represent the evolutionary differences between two species

FastTree Generated Rectangular Phylogenetic Tree



Testing Comparisons For Both Methods

Criteria	FastTree	RAxML
16S ribosomal RNAs, computing ML trees distinct families hours (15,011 distinct sequences)	0.56 hours	99 hours
Topological Accuracy of 250 aligned sequences	86.9%	90.5%
Topological Accuracy of 5000 aligned sequences	83.7%	88.4%
Avg log-likelihood for random subsets of 500 16S ribosomal RNAs	-168,577	-168,104

RaxML and FastTree Comparisons

FastTree	RAxML
Uses a combination of Neighbor-Joining, Min Evolution, and ML-based NNI rearrangement	Uses standard SPR-based hill-climbing algorithm
Can handle alignments up to 1 million sequences	Commonly used for large-scale ML phylogeny estimation
Orders of magnitude faster than RAxML and PhyML when dealing with large alignments	Has limited number of sequences and sites due to computational requirements
Relatively similar accuracy, but RAxML tends to be slightly more accurate on average	

Questions?

References

- [1] [https://en.wikipedia.org/wiki/Phylogenetic_tree#:~:text=A%20phylogenetic%20tree%20\(also%20phylogeny,their%20physical%20or%20genetic%20characteristics.](https://en.wikipedia.org/wiki/Phylogenetic_tree#:~:text=A%20phylogenetic%20tree%20(also%20phylogeny,their%20physical%20or%20genetic%20characteristics.)
- [2] <https://tandy.cs.illinois.edu/mia-project-presentation.pdf>
- [3] <https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0027731#:~:text=RAxML%20produced%20more%20accurate%20trees%20than%20RAxML%20Limited%20on%20all,PartTree%20alignment%20of%20the%2016S.>
- [4] <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3221724/>
- [5] <https://academic.oup.com/mbe/article/35/2/486/4644721>
- [6] <https://www.researchgate.net/post/Whats-the-difference-between-neighbor-joining-maximum-likelihood-maximum-parsimony-and-Bayesian-inference>
- [7] <http://www.microbesonline.org/fasttree/>
- [8] <https://slideplayer.com/slide/6997730/>
- [9] <https://www.ebi.ac.uk/goldman-srv/WAG/#:~:text=Goldman%20Group%20Projects-The%20'WAG'%20matrix%3A%20a%20new%20amino%20acid,replacement%20matrix%20for%20globular%20proteins&text=The%20method%20has%20been%20used,kindly%20provided%20by%20David%20Jones.>
- [10] <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2693737/>