Lecture 7
Phylogenetic Analysis

Additional Reference

Molecular Evolution: A Phylogenetic Approach
Roderic D. M. Page and Edward C. Holmes

Uses of Phylogentic Analysis

- Evolutionary trees
- Multiple sequence alignment

Evolutionary Problems

i) The fossil record suggests that modern man diverged from apes about 5-6 million years ago. Modern Homo sapiens emerged between 100,000-60,000 years ago

ii) DNA and sequence alignment by Paabo support this.

iii) Work based on mitochondrial DNA by Wilson et al suggest the modern man emerged only 200,000 years ago with the divergence into different races 50,000 years ago

1. mitochondrial DNA circular
2. maternal inheritance
3. 10x faster mutation rate than nuclear DNA

Algorithms

Types of Data

Tree-building Method		Distances	Nucleotide sites
	Clustering Algorithm	UPGMA Neighbor Joining	
	Optimality Criterion	Minimum Evolution	Maximum Parsimony Maximum Likelihood

From Page and Holmes
Molecular Evolution: A Phylogenetic Approach

Preliminaries

Taxon (*taxa* plural) or *operation taxon unit* is a entity whose distance from other entities can be measures (ie species, amino acid sequence, language, etc.)

Comparisons are made on measurements or assumptions concerning rates of evolutionary change. This is complicated by *back mutations*, *parallel mutations*, and *variations in mutation rate*. We will only consider *substitutions*.

Amino Acid Sequences

i) For example, the amino acid substitution rate per site per year is 5.3×10^{-9} for guinea pig but only 0.33×10^{-9} for other organisms.

ii) The evolutionary time is the average time to produce one substitution per 100 amino acids

$$T_u = \frac{1}{100\lambda}$$

Amino Acid Sequences

Example – There are 2 differences in a sequence of 100 amino acids when comparing calf and pea histone H4. Since plants and animals diverged 1 billion years ago, T_u= 0.5 billion years

$$\lambda = \frac{1}{100T_u}$$

$$\lambda \approx 10^{-11}$$

iii) probability of substitution – several way to calculate it. The best way is using the PAM matrices.

Nucleotide Sequences

i) Different from amino acid sequences due to redundancy in the genetic code (ie several codons can code for a particular amino acid.

ii) Most substitutions in the 3rd position are synonomous (UC* is the RNA coding for serine – the corresponding DNA would be AG*). Since evolution should depend on function and this is conferred by the amino acid sequence, it has been suggested that the "molecular clock" should be based on the substitution rate in the third position of the codon. In fact, in the fibrinopeptides, this is as high as the amino acid substitution rate.

Nucleotide Sequences

iii) In the definition of PAM matrices, one assumes a discrete Markov Chain, with the PAM matrix being the transition matrix for the Markov Chain.

Markov Chains

Assume that we have a process that has discrete observable states x_1, x_2, … . When we monitor this over time we get a sequence of the states occupied q_1, q_2, …. where q_i = any of x_1, x_2, ….

This sequence is a Markov Chain. Note that while there can be an infinite number of states, the Markov chain has a countable number of elements.

Markov Chains

Another property of a Markov process is that "history does not matter". This means that the state assumed at time t+1 depends on the state assumed on t (not on any other previous state). This is called the Markov property. Let $X = \{X_n, n = 1, 2, \ldots\}$ be a discrete time random process with state space S whose elements are s_1, s_2, … X is a Markov chain if for any n 0, the probability that X_{n+1} takes on any value s_k S is conditional on the value of X_n but does not depend on the values of X_{n-1}, X_{n-2}, … . The one-time-step transition probabilities

$$p_{jk}(n) = \Pr\{X_n = s_k \mid X_{n-1} = s_j\} \; j,k=1,2,\ldots \; n = 1,2, \ldots$$

Since X_0 is a random variable called the initial condition,

$$p_j(0) = \Pr\{X_0 = s_j\} \quad j=1,2,\ldots$$

Markov Chains

- Transition matrix – put the p_{jk} into a matrix P.
- A sequence of amino acids can be thought of as a Markov chain.
- Stationary Markov process – the probabilities $p_{jk}(n)$ do not depend on n, that is they are constant. Another way of saying this is an initial distribution π is said to be **stationary** if $\pi P(t)=\pi$.
- Irreducible – every state can be reached from every other state

Application of Markov processes to evolutionary models

i) The PAM matrix has its substitution probabilities determined from closely related amino acid sequences, it assumes that the substitutions have occurred through one application of the transition matrix (i.e. no multiple substitutions and a given site) and assumes that evolutionary distance results from repeated application of the same PAM matrix.

ii) A better evolutionary model is needed. (text p 140-144) This requires the use of a continuous Markov process rather than a discrete Markov chain. This still has the Markov property.

Application of Markov processes to evolutionary models

A time homogenous Markov process for the stochastic function X(t) consists of a set of states Q={1,2,…,n}, a set of initial state distributions $\pi=(\pi_1,\ldots,\pi_n)$, and transition probability functions

$$\mathbf{P}(t)=\begin{pmatrix} \mathbf{p_{1,,1}(t)} \ldots \mathbf{p_{1,n}(t)} \\ \cdot \quad\quad \cdot \\ \cdot \quad\quad \cdot \\ \mathbf{p_{n,1}(t)} \ldots \mathbf{p_{n,n}(t)} \end{pmatrix}$$

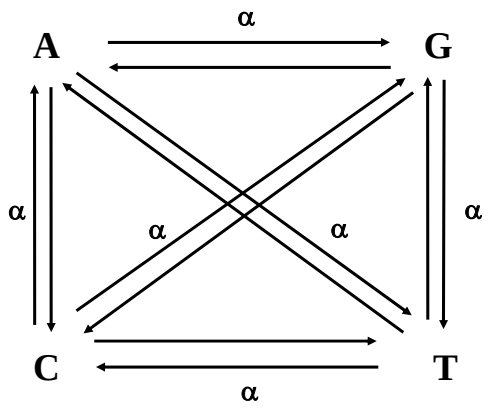
Application of Markov processes to evolutionary models

We can apply this to nucleotide sequences. Let Q={1,2,3,4} correspond to {A,C,G,T}.

$$\mathbf{P}(t)=\begin{pmatrix} \mathbf{p_{1,,1}(t)} \ldots \mathbf{p_{1,4}(t)} \\ \cdot \quad\quad \cdot \\ \cdot \quad\quad \cdot \\ \mathbf{p_{n,4}(t)} \ldots \mathbf{p_{4,4}(t)} \end{pmatrix}$$

$$\begin{pmatrix} \mathbf{P[A|A,t]} & \mathbf{P[C|A,t]} & P[G|A,t] & P[T|A,t] \\ P[A|C,t] & P[C|C,t] & P[G|C,t] & P[T|C,t] \\ P[A|G,t] & P[C|G,t] & P[G|G,t] & P[T|G,t] \\ P[A|T,t] & P[C|T,t] & P[G|T,t] & P[T|T,t] \end{pmatrix}$$

Jukes-Cantor Model



Transitions = Transversions

Rates of Nucleic Acid Change

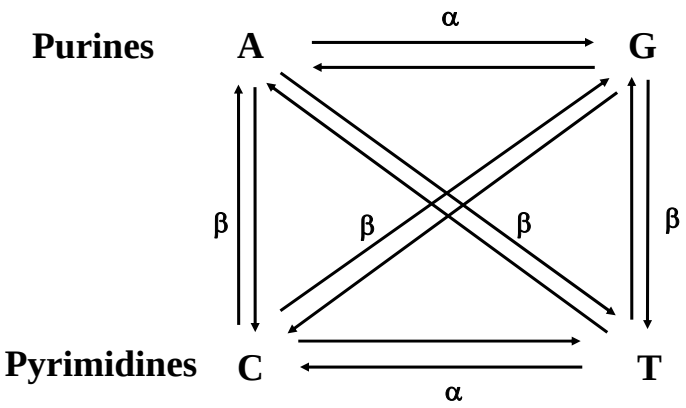
The Jukes Cantor model assumes that $u_1=u_2=u_3=u_4=a$, yielding the rate matrix.

$$\Lambda = \begin{pmatrix} -3\alpha & \alpha & \alpha & \alpha \\ \alpha & -3\alpha & \alpha & \alpha \\ \alpha & \alpha & -3\alpha & \alpha \\ \alpha & \alpha & \alpha & -3\alpha \end{pmatrix}$$

Then $p_1=p_2=p_3=p_4=a$

Use in Maimum Likelihood Calculation

HKY Model



Transitions > Transversions

$\alpha > \beta$

Definitions

-*taxa* – entities whose distance from other entities can be measured

-A *directed graph* G(V, E) consists of a set V of *nodes* or *vertices* and a set E(V) of *directed edges*. Then (i,j) E means that there is a directed edge from i to j.

-A graph is *undirected* if the edge relation is *symmetric*, that is, (i,j) E iff (j,i) E.

-A directed graph is *connected* if there is a directed path between any two nodes.

Definitions

-A directed graph is *acyclic* if it does not contain a cycle. (i.e. (i,j), (j,k), and (k,i) all belong to E.

-A *tree* is a undirected, connected, acyclic graph.

-A *rooted tree* has a starting node called a *root*.

-The *parent node* is immediately before a node on the path from the root.

-The *child node* is a node that is follows a node.

Definitions

-An *ancestor* is any node that came before a node on the path from a root.

-A *leaf* or *external node* is a node that had no children.

-Non-leaf nodes are called *internal nodes*.

-The *depth* of a tree is one less than the maximal number of nodes on a path from the root to a leaf.

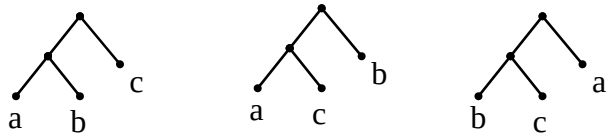
Definitions

-An *ordered tree* is a tree where the children of internal nodes are numbered.

-A *binary tree* is a tree where each node has at most two children. Otherwise it is *multifurcating*.

Trees

Question: Draw all binary trees on 1, 2, and 3 taxa.



A *phylogenetic tree* on n taxa is a tree with leaves labeled by 1,…,n.

How do you tell if two trees are the same?

If you can convert one tree into another without breaking any branches they are topologically equivalent.

Phylogenetic Trees

Phylogenetic trees or *evolutionary trees* are *binary* trees that describe the "relations" between species.

Trees consist of *nodes* or *vertices* and *taxa* or *leaves.*

Phylogenetic Trees

To understand the data, we must understand some of the methods behind phylogenetic trees or evolutionary trees

i) Clustering methods

ii) Maximum likelihood methods

iii) Quartet puzzling

What do we do with phylogenetic trees?

- measuring evolutionary change on a tree

  If the leaves of a tree each signify a sequence, the sum of the weights of the edges gives the evolutionary distance between the two sequences.

- molecular phylogenetics

  Convert information in sequences into an evolutionary tree for those sequences.

Cluster methods vs. search methods

There are two basic methods for constructing trees.

*Cluster method*s use an algorithm (set of steps) to generate a tree. These methods are very easy to implement and hence can be computationally efficient. They also typically produce a single tree. A big disadvantage to this method is that it depends upon the order in which we add sequences to the tree. Hence, there could be a different tree that explains the data just as well.

*Search method*s use some sort of optimality criteria to choose among the set of all possible trees. The *optimality criteria* gives each tree a score that is based on the comparison of the tree to data. The advantage of search methods is that they use an explicit function relating the trees to the data (for example, a model of how the sequences evolve). The disadvantage is that they are computationally very expensive (NP complete problem).

How do we compare different tree methods?

- Efficiency – How fast is the method?
- power –How much data does the method require?
- consistency – Will the tree converge on the right answer give enough data?
- robustness – Will minor violations of the method's assumptions result in poor estimates of phylogeny?
- falsifiability – Will the method tell us when its assumptions are violated?

How do assign weights for the edges of our trees?

- *Distance methods* first convert aligned sequences into a pairwise distance matrix then input that matrix into a tree building method. The major objections to distance methods are that summarizing a set of sequences by distance data loses information and branch lengths estimated by some distance methods might not be evolutionarily determinable.
- *Discrete methods* consider each nucleotide site (of some function of each site) directly.

Distance Methods

•Two distance methods are **neighbor joining** and **minimum evolution.**

• *Minimum evolution* finds the tree that minimizes the sum of the branch lengths where the lengths are calculated from the pairwise distances between the sequences. Linear programming or least squares methods can be used to do this.

•*Neighbor joining* is a clustering method that is computationally fast and gives a unique result. This can use something like the *four-point condition* and clusters the closest elements.

Discrete Methods

The two major discrete methods are **maximum parsimony** and **maximum likelihood**. Both these are search methods.

i) With *maximum parsimony* we try to reconstruct the evolution at a particular site with the fewest possible evolutionary changes. The **advantages** of parsimony are that it makes relatively few assumptions about the evolutionary process, it has been studied extensively mathematically, and some very powerful software implementations are available. The major **disadvantage** to using parsimony is that under some models of evolution, it is inconsistent , that is if more data is added the wrong result might occur.

Discrete Methods

ii) The *maximum likelihood approach* looks for the tree that makes the data the most probable evolutionary outcome. This approach requires a explicit model of evolution which is both a strength and weakness because the results depend on the model used. This method can also be very computationally expensive.

Types of metrics

For the *four point condition* or *additive metric*, given the leaves i, j, k, and l

$d(i,j) + d(k,l) \leq d(i,k) + d(j,l) = d(i,l) + d(j,k)$

For an ultrametric metric the *ultrametric* or *3-point condition* holds

That is given the leaves i, j, and k

$d(i,j) \leq d(i,k) = d(j,k)$

Ultrametric trees

- Clustering methods attempt to repeated cluster the data by grouping the closest elements together. They are used for phylogeny and gene expression microarray analysis.
- The *pair group method* (PGM) is a technique where the pairs are repeatedly amalgamated.
- The *unweighted paired group method with arithmetic mean* (UPGMA) is used to cluster molecular data where sequence alignment distance between sequences has been determined in a distance matrix.

UPGMA

Input an n x n distance matrix D

1. Initialize a set C to consist of n singleton clusters
2. Initialize dist(c,d) on C by defining for all {i} and {j} in C

   dist({i},{j}) = D(i,j)
3. Repeat the following n-1 times

   a) determine a pair c,d of clusters in C such that dist (c,d) in minimal; define d_{min}= dist(c,d)

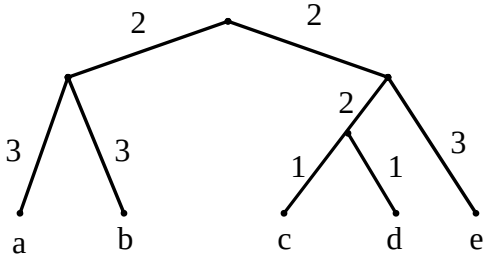
   b) define a new cluster e = c U d; define C = C – {c,d} U {e}

   c) define a node with label e and daughters c and d, where the e has distance dmin/2 to its leaves

   d) define for all f in C with f different from e

   dist(e,f) = dist(f,e) = [dist(c,f) + dist(d,f)]/2

UPGMA Example



	a	b	c	d	e
a	0	6	10	10	10
b	6	0	10	10	10
c	10	10	0	2	6
d	10	10	2	0	6
e	10	10	6	6	0

Ultrametric Topology Distance Table

$d(i,j) \leq d(i,k) = d(j,k)$

from Clote and Backofen *Computational Molecular Biology*

UPGMA Example

Given the distance table

	a	b	c	d	e
a	0	6	10	10	10
b	6	0	10	10	10
c	10	10	0	2	6
d	10	10	2	0	6
e	10	10	6	6	0

1. We have five singleton clusters {a}, {b}, {c}, {d}, and {e} from the set C = {a,b,c,d,e}
2. Get the distances from the distance table (left)
3. a) Find the closest two clusters, namely, clusters {c} and {d} with $d_{min} = 2$

   b) f = {c,d} and C= {a,b,e,f}

   c) f is the root for c and d

   d) Define new distance table

Repeat 3

UPGMA Example

The old distance table

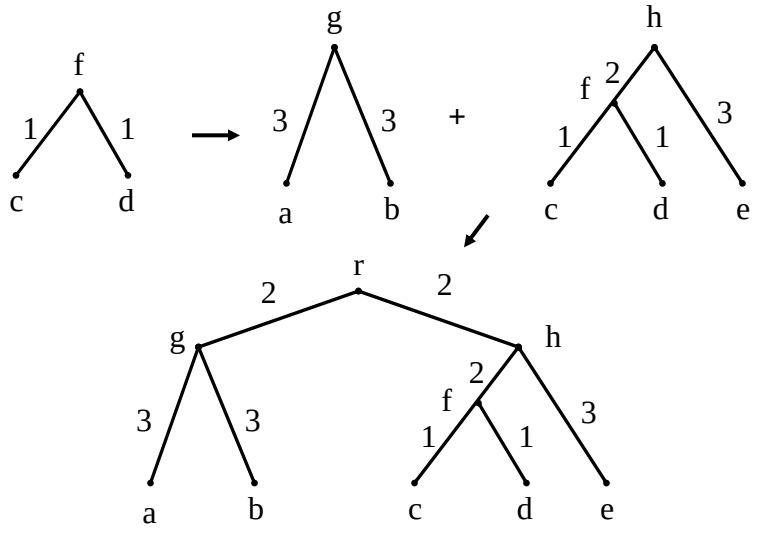
	a	b	c	d	e
a	0	6	10	10	10
b	6	0	10	10	10
c	10	10	0	2	6
d	10	10	2	0	6
e	10	10	6	6	0

The new distance table

	a	b	e	f
a	0	6	10	10
b	6	0	10	10
e	10	10	0	6
f	10	10	6	0

UPGMA example

Tree formation



WPGMA

Input an n x n distance matrix D

1. Initialize a set C to consist of n singleton clusters
2. Initialize dist(c,d) on C by defining for all {i} and {j} in C

   dist({i},{j}) = D(i,j)
3. Repeat the following n-1 times

   a) determine a pair c,d of clusters in C such that dist (c,d) in minimal; define d_{min}= dist(c,d)

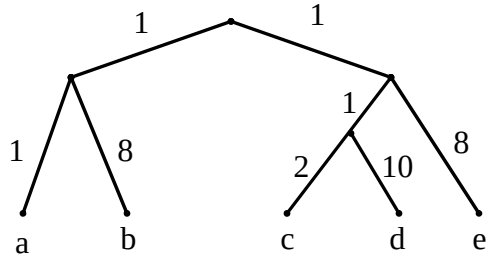
   b) define a new cluster e = c U d; define C = C – {c,d} U {e}

   c) define a node with label e and daughters c and d, where the e has distance dmin/2 to its leaves

   d) define for all f in C with f different from e

   dist(e,f) = dist(f,e) = [|c|dist(c,f) + |d|dist(d,f)]/[|c|+|d|]

Farris Transform - Example



Additive, non-ultrametric topology

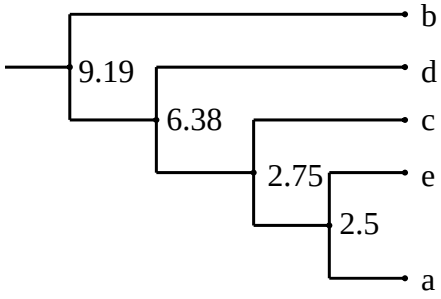
	a	b	c	d	e
a	0	9	6	14	11
b	9	0	13	21	18
c	6	13	0	12	11
d	14	21	12	0	19
e	11	18	11	19	0

Distance Table

$$d(i,j) + d(k,l) \leq d(i,k) + d(j,l) = d(i,l) + d(j,k)$$

from Clote and Backofen *Computational Molecular Biology*

Farris Transform - Example



UPGMA incorrectly reconstructed topology

	a	b	c	d	e
a	0	9	6	14	11
b	9	0	13	21	18
c	6	13	0	12	11
d	14	21	12	0	19
e	11	18	11	19	0

Distance Table

from Clote and Backofen *Computational Molecular Biology*

Farris Transform

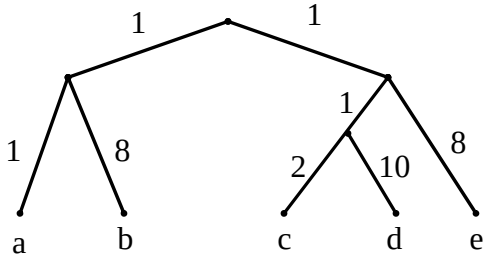
- Sometimes the data will satisfy an additive metric and not a ultrametric. This will yield a tree with the incorrect topology if UPGMA or WPGMA is used.
- The *Farris Transformed Distance Method* converts the data for an additive, non-ultrametric metric so that it satisfies the ultrametric. Then UPGMA or WPGMA can be used to yield a tree with the correct topology

Farris Transform

If we have a phylogenetic tree with root r and leaves (taxa) 1,…,n and $d_{i,j}$ is the distance between two nodes, then we have the transformed distance

You must assume a root r. This can be the leaf that is farthest from all the others. Unfortunately, depending on the root selected the method might not give the right topology.

Farris Transform - Example



Additive, non-ultrametric topology

What is the distance to the root?

	a	b	c	d	e
a	0	9	6	14	11
b	9	0	13	21	18
c	6	13	0	12	11
d	14	21	12	0	19
e	11	18	11	19	0

Distance Table

	a	b	c	d	e
r	2	9	4	12	9

Farris Transform - Example

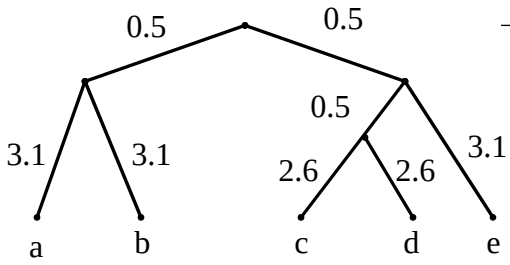
Original Distance Table with assumed root

	a	b	c	d	e	r
a	0	9	6	14	11	2
b	9	0	13	21	18	9
c	6	13	0	12	11	4
d	14	21	12	0	19	12
e	11	18	11	19	0	9

Transformed Distance Table

	a	b	c	d	e
a	0	6.2	7.2	7.2	7.2
b	6.2	0	7.2	7.2	7.2
c	7.2	7.2	0	5.2	6.2
d	7.2	7.2	5.2	0	6.2
e	7.2	7.2	6.2	6.2	0

Farris Transform - Example



Farris transformed tree topology

	a	b	c	d	e
a	0	6.2	7.2	7.2	7.2
b	6.2	0	7.2	7.2	7.2
c	7.2	7.2	0	5.2	6.2
d	7.2	7.2	5.2	0	6.2
e	7.2	7.2	6.2	6.2	0

Distance Table

$d(i,j) + d(k,l) \leq d(i,k) + d(j,l) = d(i,l) + d(j,k)$

Farris Transform – Pick d as root

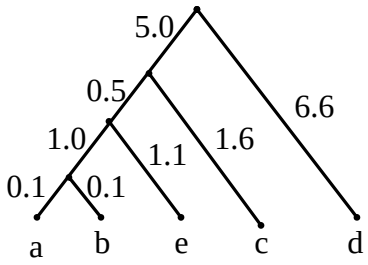
Original Distance Table with assumed root

	a	b	c	d	e
a	0	9	6	14	11
b	9	0	13	21	18
c	6	13	0	12	11
d	14	21	12	0	19
e	11	18	11	19	0

Transformed Distance Table

	a	b	c	d	e
a	0	0.2	3.2	13.2	2.2
b	0.2	0	3.2	13.2	2.2
c	3.2	3.2	0	13.2	3.2
d	13.2	13.2	13.2	0	13.2
e	2.2	2.2	3.2	13.2	0

Farris Transform – Correct Topology!



Farris transformed tree topology

	a	b	c	d	e
a	0	0.2	3.2	13.2	2.2
b	0.2	0	3.2	13.2	2.2
c	3.2	3.2	0	13.2	3.2
d	13.2	13.2	13.2	0	13.2
e	2.2	2.2	3.2	13.2	0

Distance Table

$$d(i,j) + d(k,l) \leq d(i,k) + d(j,l) = d(i,l) + d(j,k)$$

Algorithms

Types of Data

Tree-building Method		Distances	Nucleotide sites
Clustering Algorithm		UPGMA Neighbor Joining	
Optimality Criterion		Minimum Evolution	Maximum Parsimony Maximum Likelihood

From Page and Holmes
Molecular Evolution: A Phylogenetic Approach

Phylogeny: Distance Methods

- **Parsimony**
- **Maximum Likelihood**

- **Look at changes in each column of alignment**
- **Metric to estimate Population Drift**
- **Computationally more expensive**

Neighbor Joining

- Combines computational speed with uniqueness of result
- Clustering method – hence has no optimality criteria.
- Often used in conjunction with Minimum Evolution to estimate the minimum evolution tree

Neighbor Joining and Minimum Evolution

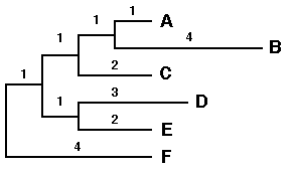
- Compute the Neighbor Joining Tree and see if any local rearrangement produces a shorter tree.
- Not guaranteed to give the minimum evolution tree.

Neighbor Joining Algorithm

- Related to cluster analysis but removes the assumption of ultrametric data
- Does not assume data comes close to fitting an additive tree (need to use an appropriate model of evolution).
- Keeps track of nodes on tree
- Considers only closest pairs and not all possible pairs in each step of star decomposition.

Nieghbor Joining

- FROM: http://www.icp.ucl.ac.be/~opperd/private/neighbor.html
- Author: Fred Opperdoes
- Suppose we have the following tree:
- Since B and D have accumulated mutations at a higher rate than A. The Three-point criterion is violated and the UPGMA method cannot be used since this would group together A and C rather than A and B. In such a case the neighbor-joining method is one of the recommended methods.



Neighbor Joining

The raw data of the tree are represented by the following distance matrix:

	A	B	C	D	E
B	5				
C	4	7			
D	7	10	7		
E	6	9	6	5	
F	8	11	8	9	8

We have in total 6 OTUs (N=6).

Neighbor Joining

Step 1: We calculate the net divergence r (i) for each OTU from all other OTUs

r(A) = 5+4+7+6+8=30
r(B) = 42
r(C) = 32
r(D) = 38
r(E) = 34
r(F) = 44

Neighbor Joining

Step 2: Now we calculate a new distance matrix using for each pair of OTUs the formula:

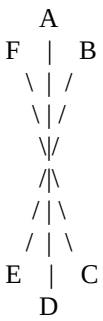
M(I,j)=d(i,j) - [r(i) + r(j)]/(N-2) or in the case of the pair A,B:

M(AB)=d(AB) -[(r(A) + r(B)]/(N-2) = -13

	A	B	C	D	E
B	-13				
C	-11.5	-11.5			
D	-10	-10	-10.5		
E	-10	-10	-10.5	-13	
F	-10.5	-10.5	-11	-11.5	-11.5

Neighbor Joining

Now we start with a star tree:



Neighbor Joining

- **Step 3**: Now we choose as neighbors those two OTUs for which M_{ij} is the smallest. These are A and B and D and E. Let's take A and B as neighbors and we form a new node called U. Now we calculate the branch length from the internal node U to the external OTUs A and B.
- S(AU) =d(AB) / 2 + [r(A)-r(B)] / 2(N-2) = 1
  S(BU) =d(AB) -S(AU) = 4

Neighbor Joining

- **Step 4**: Now we define new distances from U to each other terminal node:
- d(CU) = d(AC) + d(BC) - d(AB) / 2 = 3
  d(DU) = d(AD) + d(BD) - d(AB) / 2 = 6
  d(EU) = d(AE) + d(BE) - d(AB) / 2 = 5
  d(FU) = d(AF) + d(BF) - d(AB) / 2 = 7 and we create a new matrix:

Neighbor Joining

U C D E
C 3
D 6 7
E 5 6 5
F 7 8 9 8

The resulting tree will be the following:

N= N-1 = 5

The entire prodcedure is repeated starting at step 1

Quartet Puzzling

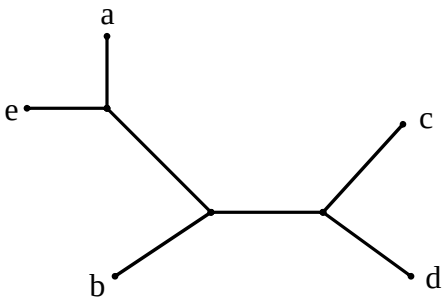
Quartet puzzling is a less computationally expensive method than maximum likelihood to determine the phylogentic tree.

Procedure:

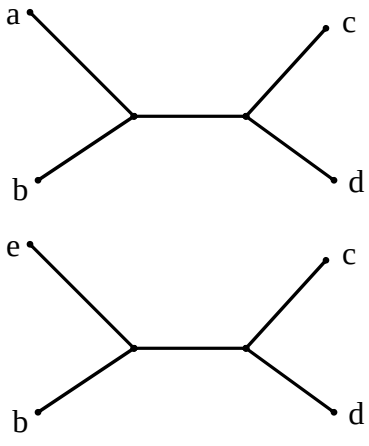
1. Compute the $\binom{n}{4}$ maximum likelihood trees for all possible quartets
2. (Quartet Puzzling step) Combine the quartet trees into a n-taxon tree that tries to conform to all the neighbor relations of all the quartet trees.
3. Repeat steps 1. and 2. many times and use the majority consensus tree.

Quartet Puzzling

Given the original tree topology for 5 taxa

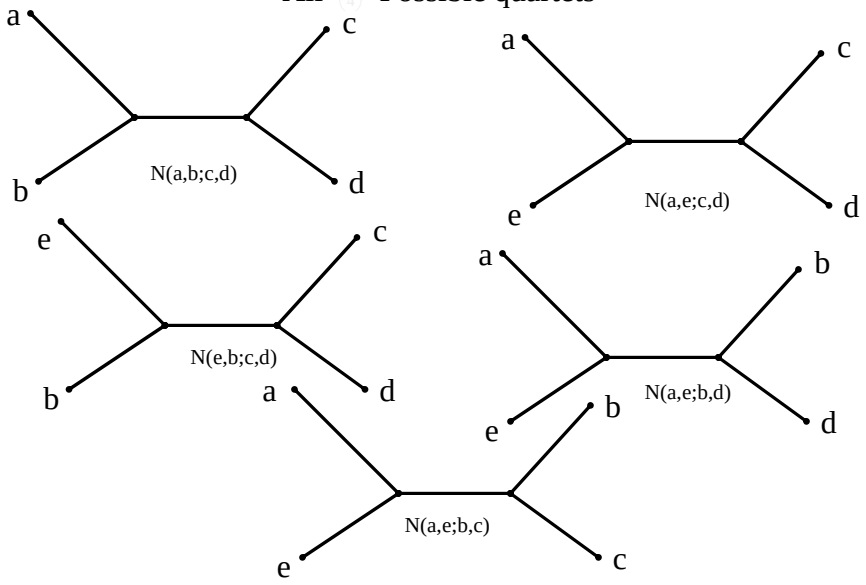


Two possible quartets



Quartet Puzzling

All $\binom{5}{4}$ Possible quartets



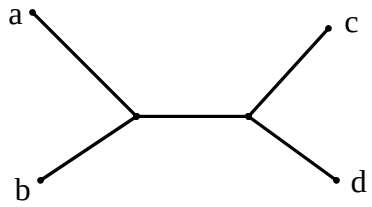
Quartet Puzzling

Quartet puzzling step procedure:

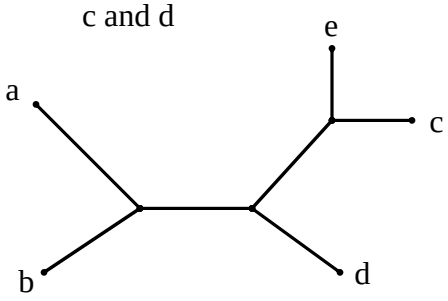
1. Take one of the quartet with the neighbor relation N(a,b;c,d)
2. Add a penalty of 1 to every edge such that the addition of the new taxa e will yield the incorrect topology.
3. Repeat for all the neighbor relations
4. The branch with the lowest weight is the branch where the taxa e show be added

Quartet Puzzling

Quartet with neighbor relation N(a,b;c,d)

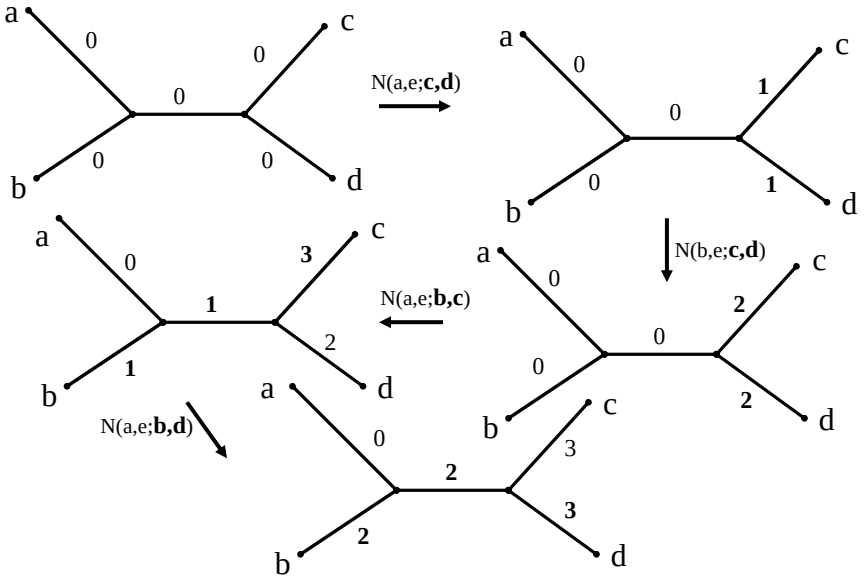


Adding taxa e between c and d



Yields wrong topology!

Quartet Puzzling



Quartet Puzzling

- Chose one quartet tree.
- Pick the taxa to add
- Use all neighbor relations (other than the one deciding the quartet tree used) to find weights on branches
- Add the taxa to the branch with the lowest penalty.

Minimum Evolution

- Given an unrooted metric tree for n sequences, there are (2n-3) branches each with branch length e_i.
- The sum of these branch lengths is the length L of the tree.
- The minimum evolution tree is the tree which minimizes L

Minimum Evolution

- similar to parsimony
- But length comes from pairwise distances between the sequences (not from fit of nucleotide sites)
- Use linear programming or least squares to find optimal solution.

Minimum Evolution

Table 6.2 Observed pairwise distances (p) between hominoid sequences (above diagonal) and tree distances computed by linear programming (below diagonal). Tree distances that differ from the observed are marked in bold. The corresponding tree is shown in Fig. 6.10.

	Human	Chimp	Gorilla	Orang-utan	Gibbon
Human	–	79	92	144	162
Chimp	79	–	95	154	169
Gorilla	92	**102**	–	150	169
Orang-utan	144	154	150	–	169
Gibbon	**163**	**173**	169	169	–

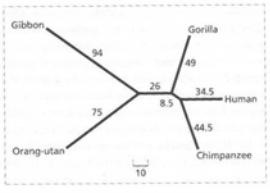


Fig. 6.10 Minimum evolution tree for the hominoid sequences with the branch lengths computed from the observed pairwise distances using linear programming. The total length of this tree is 331.5.

Phylogeny: Character State Methods

- **Parsimony**
- **Maxmum Likelihood**

- **Look at changes in each column of alignment**
- **Metric to estimate Population Drift**
- **Computationally more expensive**

PHYLOGENY: Character States

Taxa 1	**ATT-GCCATT**
Taxa 2	**ATG-GC-ATT**
Taxa 3	**ATC-TATCTT**
Taxa 4	**ATCAAATCTT**
Taxa 5	**ACT-G--ACC**

Informative characters (columns)
Look at all possible trees
For each column, calculate cost
Minimum score = best tree

Maximum Parsimony

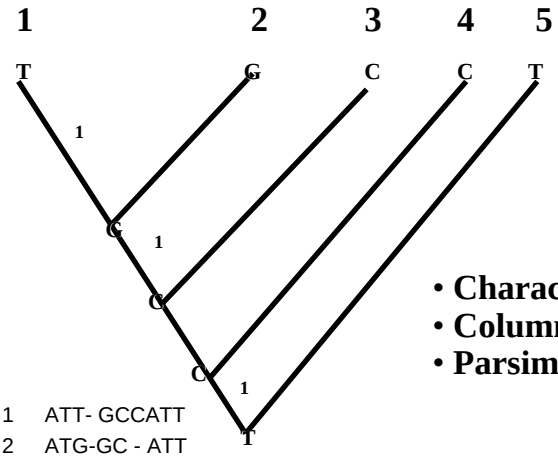
Taxa 1 ATT-GCCATT
Taxa 2 ATG-GC-ATT
Taxa 3 ATC-TATCTT
Taxa 4 ATCAAATCTT
Taxa 5 ACT-G--ACC

Informative characters
Minimum number of changes
Multiple substitutions = homoplasy

Maximum Parsimony

- Smallest number of evolutionary changes
- First used on protein data (Eck & Dayhoff, 1966)
- Applied to Nucleotide data (Fitch, 1977)
- Brute force search of tree space

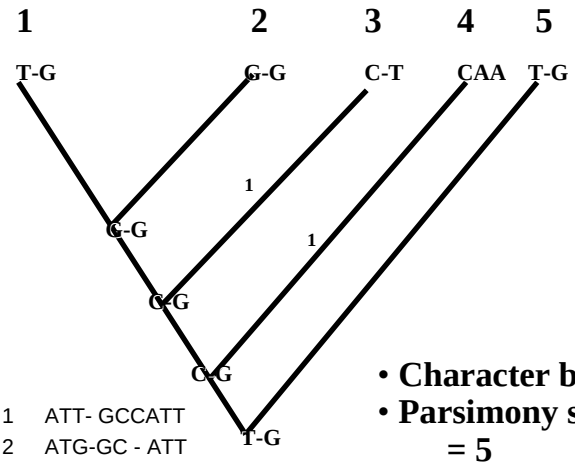
Cost of Parsimony Tree



- **Character by character**
- **Column 3**
- **Parsimony score = 0+0+3**

Taxa 1	ATT- GCCATT
Taxa 2	ATG-GC - ATT
Taxa 3	ATC- TAT CTT
Taxa 4	ATCAAAT CTT
Taxa 5	ACT- G - - ACC

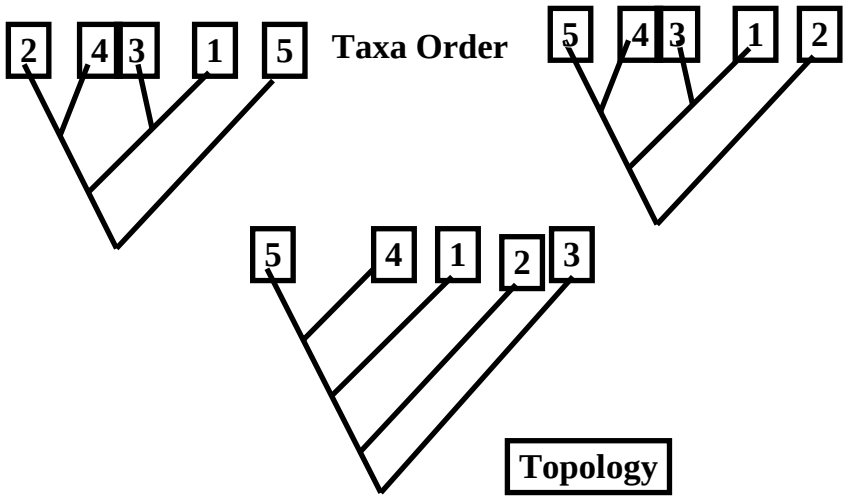
Cost of Parsimony Tree



- **Character by character**
- **Parsimony score = 0+0+3+0+2 = 5**

Taxa 1	ATT- GCCATT
Taxa 2	ATG-GC - ATT
Taxa 3	ATC- TAT CTT
Taxa 4	ATCAAAT CTT
Taxa 5	ACT- G - - ACC

Tree Space



Search Tree Space

- **Exhaustive Search (Brute Force)**
- **Branch and Bound (Efficient?)**
- **Heuristic Methods (Hill Climbing)**
- **Genetic Algorithms (GAML)**

Maximum Likelihood

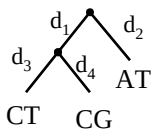
- Goal: Construct a phylogenetic tree from DNA sequences whose likelihood is a maximum. (Felsenstein 1981)
- Procedure
  - Start with a given topology and use the maximum likelihood method to optimize branch lengths
  - Make local modifications to the topology and re-optimize the branch lengths
  - New taxa are added one by one, optimizing branch lengths and topologies each time
  - Assumes an evolutionary process that is a reversible Markov process
  - Very computationally expensive to use

Likelihood of a Tree

We want to find L(tree) = Pr[data|tree]

Given the data a_1=CT, a_2=CG and a_3=AT

Consider the tree

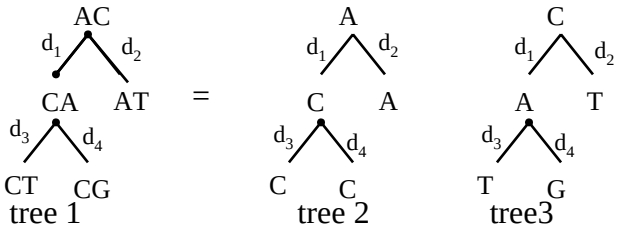


We can calculate the likelihood of this tree if we fill in the internal nodes

Likelihood of a Tree

Since this is a Markov process, we can consider each site separately from the other which reduces the complexity of the calculation.

Example



Pr[data|tree 1] = Pr[data|tree 2] + Pr[data|tree 3]

Likelihood of a site specific tree

We can calculate from the transition matrix and the distances on each branch the probability of each change. The product of these multiplied by the probability of the original base gives the likelihood of a site specific tree.

Since there are two unknown nodes the double sum of all possible values for each (ACTG) gives the likelihood for the original tree.

Maximum Likelihood

- **Statistical model for changes in nucleotides**
- **Likelihood that that change occurred**
- **Much more computational intensive than parsimony**
- **Hypothesis Testing**

- **Transitions/Transversions**
- **HKY (Kimura 2 parameter model)**
- **Jukes Cantor (1 parameter)**

Maximum Likelihood

Taxa 1	ATT-GCCATT
Taxa 2	ATG-GC-ATT
Taxa 3	ATC-TATCTT
Taxa 4	ATCAAATCTT
Taxa 5	ACT-G--ACC

- **Statistical model for changes in nucleotides**
- **Transitions/Transversions**
- **HKY (Kimura 2 parameter model)**
- **Jukes Cantor (1 parameter)**
- **Likelihood that that change occurred**
- **Much more computational intensive than parsimony**

Likelihood of a Tree

L(tree) = Pr [data|tree]

- **Multiply likelihood for each character position**
- **Recursive definition of Likelihood**
- **Saves computational time**

Likelihood of a Tree

L(tree) = Pr [data|tree]

Multiply likelihood for each character position

Recursive definition of Likelihood

Likelihood of a Tree