Computing a tree

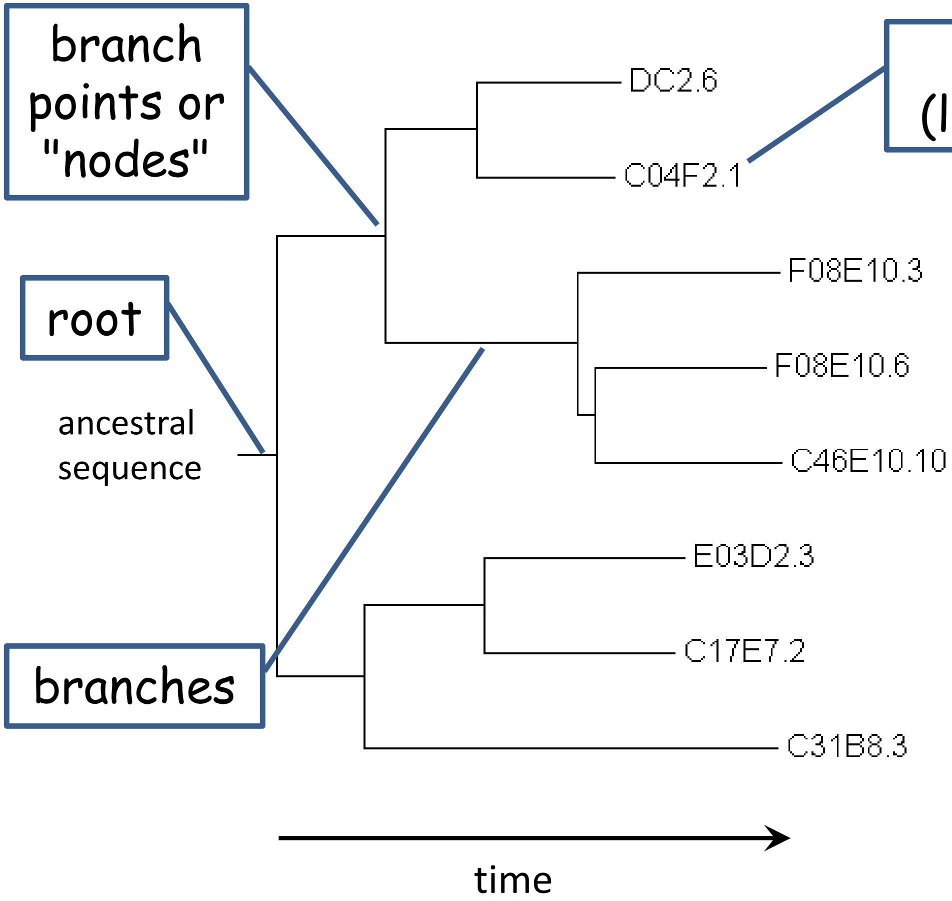
http://faculty.washington.edu/jht/GS559_2017/

Genome 559: Introduction to Statistical and Computational Genomics

Prof. James H. Thomas

Defining what a “tree” means

rooted tree (all real trees are rooted):



branch points or "nodes"

root

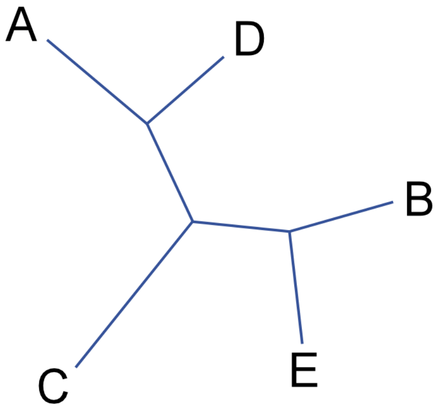
ancestral sequence

branches

sequences (leaves or tips)

time

unrooted tree (used when the root isn’t known):

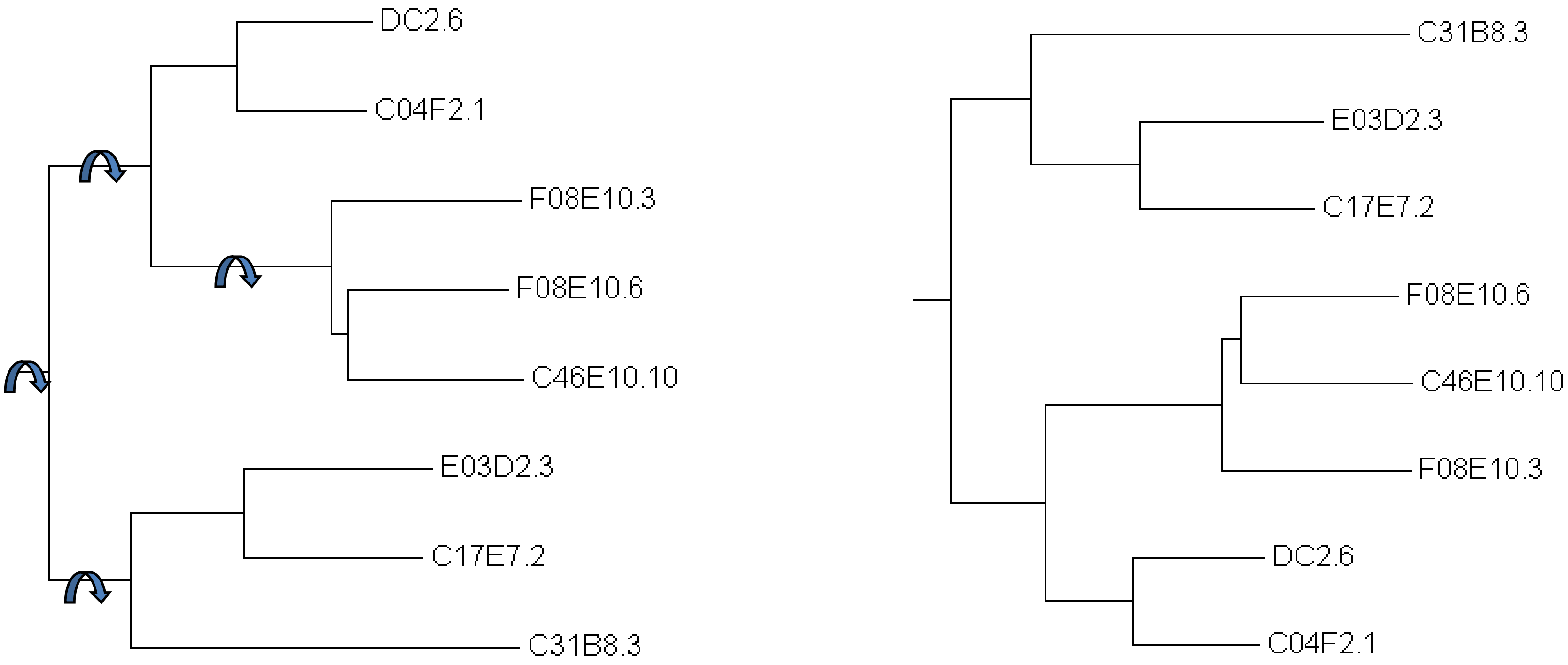


time radiates out from somewhere, usually near the center

…divergence time is the sum of (horizontal) branch lengths

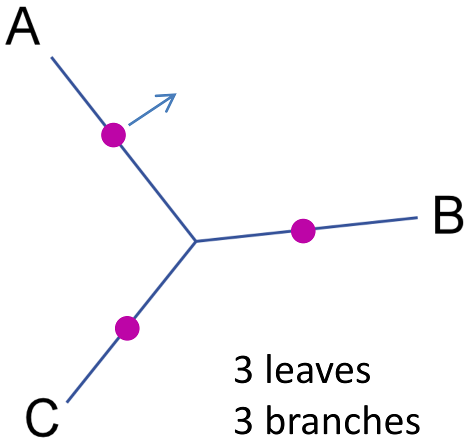
A tree has topology and distances

Are these different trees?

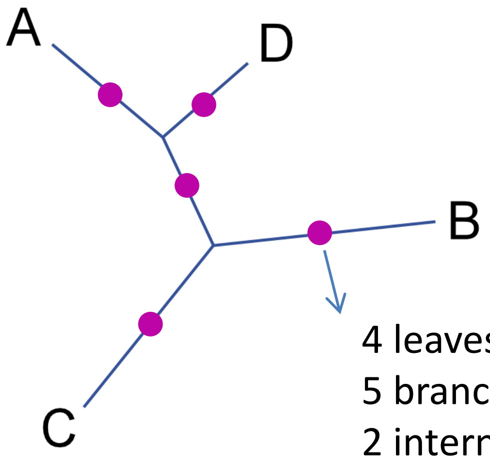


Topologically, these are the SAME tree. In general, two trees are the same if they can be inter-converted by branch rotations.

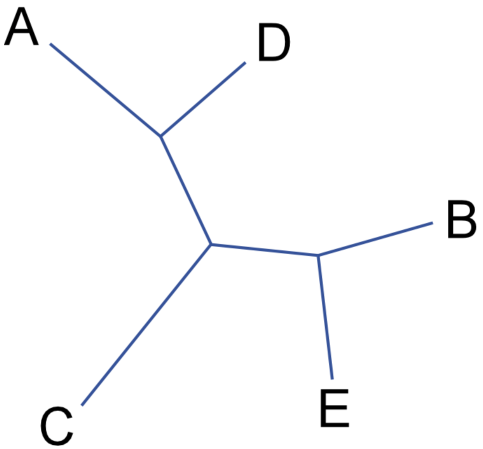
The number of tree topologies grows extremely fast



3 leaves
3 branches
1 internal node
1 topology
(3 insertions)



4 leaves
5 branches
2 internal nodes
3 topologies (x3)
(5 insertions)

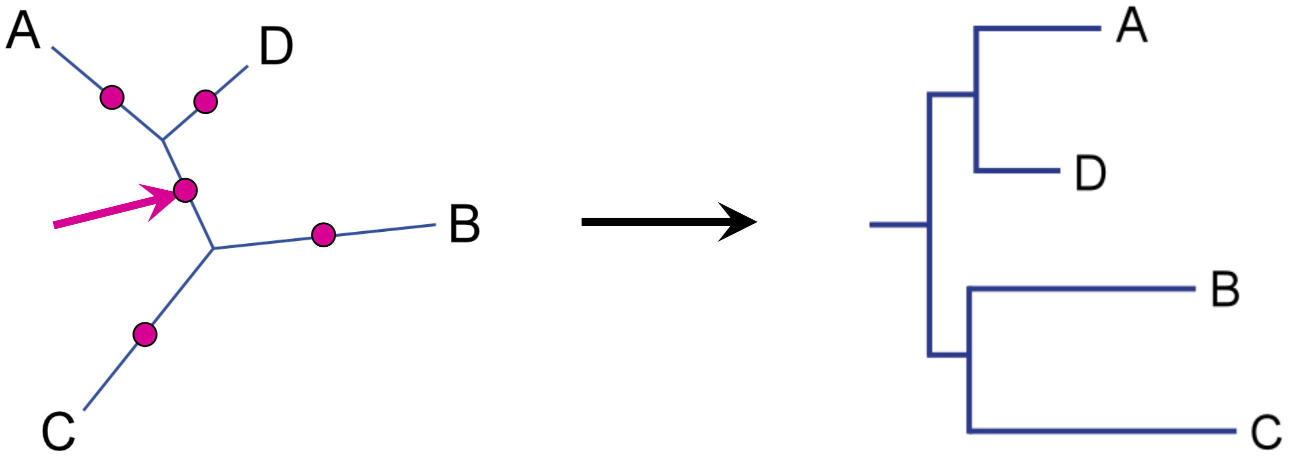


5 leaves
7 branches
3 internal nodes
15 topologies (x5)
(7 insertions)

In general, an unrooted tree with N leaves has:
2N – 3 branches
N – 2 internal nodes
~ O(N!) topologies $3*5*7*...*(2N-5)$

There are many rooted trees for each unrooted tree

For each unrooted tree, there are 2N - 3 times as many rooted trees, where N is the number of leaves (# internal branches = 2N – 3).



20 leaves - 564,480,989,588,730,591,336,960,000,000 topologies

How can you compute a tree?

Many methods available, we will talk about:

Distance trees
Parsimony trees

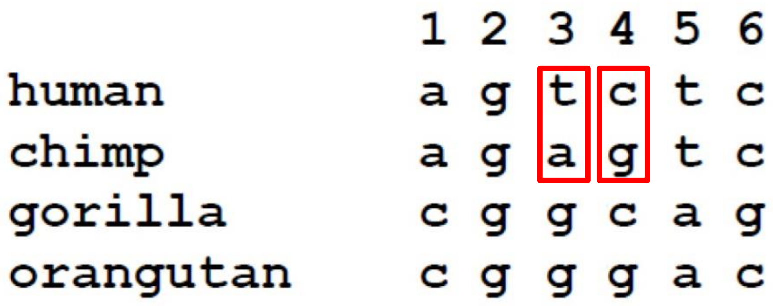
Others include:

Maximum-likelihood trees
Bayesian trees

Distance tree methods

- Measure pairwise 'distance' between each pair of sequences.

- Use a clustering method to build up a tree, starting with the closest pair.



	1	2	3	4	5	6
human	a	g	t	c	t	c
chimp	a	g	a	g	t	c
gorilla	c	g	g	c	a	g
orangutan	c	g	g	g	a	c

human - chimp has 2 changes out of 6 sites □
human - orang has 4 changes of out 6 sites
etc.

Distance matrix from alignment

	human	chimp	gorilla	orang
human	0	2/6	4/6	4/6
chimp		0	5/6	3/6
gorilla			0	2/6
orang				0

(symmetrical, lower left not filled in)

Distance matrix methods

- Methods based on a set of pairwise sequence distances, typically from a multiple alignment.

- Try to build the tree that best matches the distances.

- Usual standard for "best match" is the least squares of the tree distances compared to the real pairwise distances:

Let $\boldsymbol{D_m}$ be the real (matrix) distances and $\boldsymbol{D_t}$ be the tree distances. Find the tree (with associated $\boldsymbol{D_t}$) that minimizes:

$$\sum_{i=1}^{N} \left(D_t - D_m\right)^2$$

Enumerate and score all trees

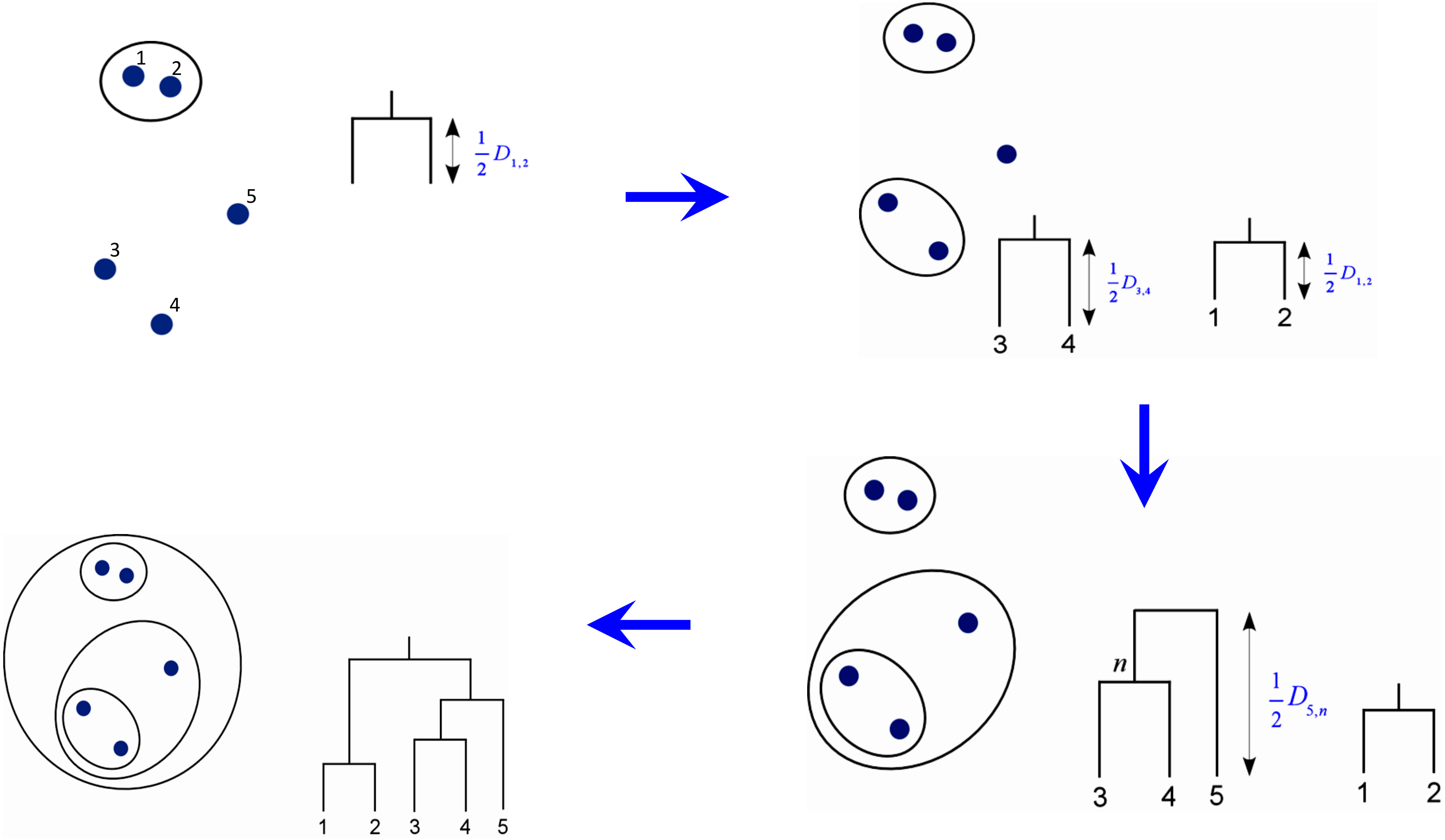
- Enumerate every tree topology, fit least-squares best distances for each topology, keep best.

- Not used for distance trees - there is a much faster way to get very close to correct.

- Called Neighbor-Joining algorithm, one of a general class called hierarchical clustering algorithms.

- I will show a slightly simpler algorithm called UPGMA (**U**nweighted **P**air **G**roup **M**ethod with **A**rithmetic Mean).

Sequential clustering approach (UPGMA)



Sequential clustering algorithm

1) generate a table of pairwise sequence distances and assign each sequence to a list of N tree nodes.

2) look through the current list of nodes (initially these will all be leaf nodes) for the pair with the smallest distance.

3) merge the closest pair, remove the pair of nodes from the list and add back the merged node to the list.

4) repeat until there is only one node left - it is the root.

$$D_{n1,n2} = \frac{1}{N}\sum_{i}\sum_{j} d_{ij}$$

distance when there are multiple sequences in a node

where i is each leaf of $n1$ (node1), j is each leaf of $n2$ (node2),

and N is the number of distances summed $(= i * j)$

(in words, this is the arithmetic average of the distances between all the leaves in one node and all the leaves in the other node)

Neighbor-Joining Algorithm (side note)

Essentially as for UPGMA, but correction for distance to other leaves is made.

Specifically, for sets of leaves ***i*** and ***j***, we denote the set of all other leaves as L, and the size of that set as $|L|$, and we compute the corrected distance D_{ij} as:

$$D_{ij} = d_{ij} - (r_i + r_j)$$

(d_{ij} is calculated as before)

where

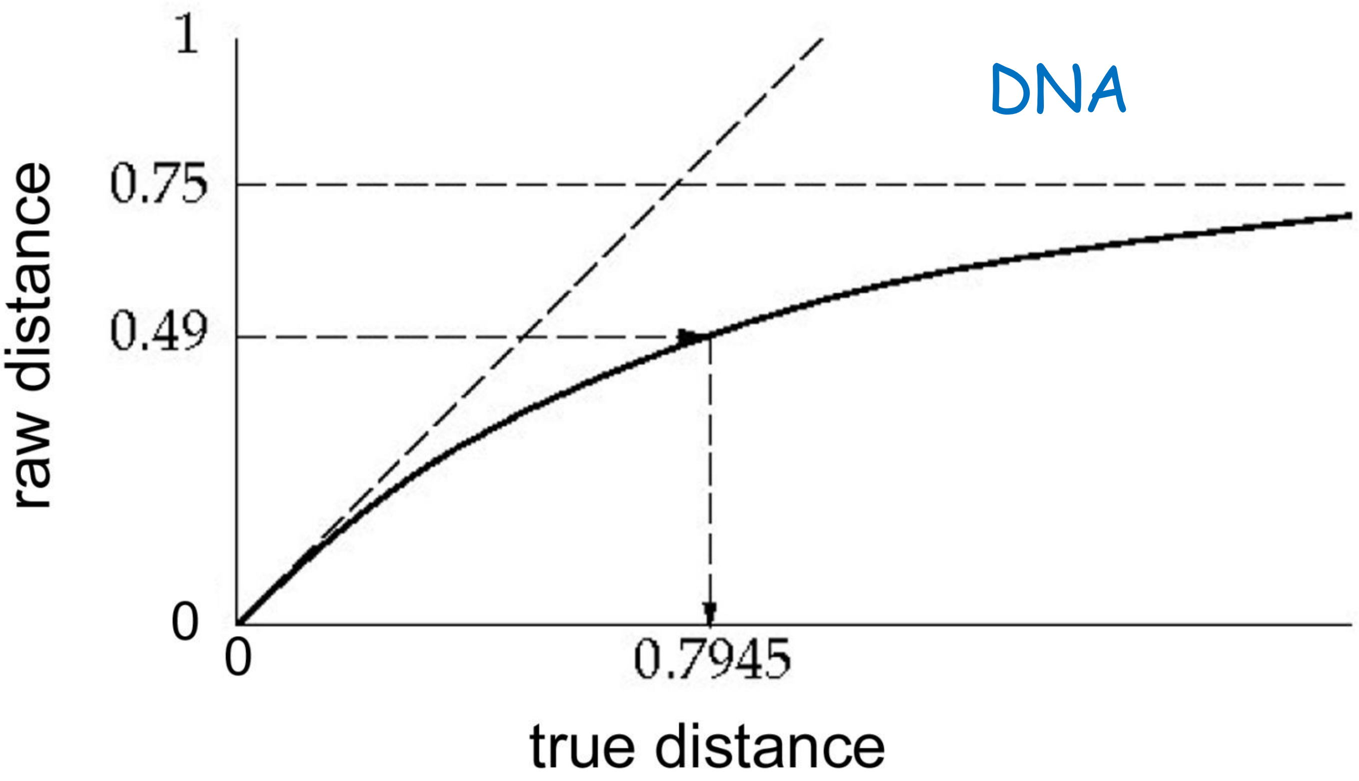
$$r_i = \frac{1}{|L|}\sum_{k \in L} d_{ik} \text{ and } r_j = \frac{1}{|L|}\sum_{k \in L} d_{jk}$$

(the mean distance from
i to all 'other' leaves)

Raw distance correction

- As two DNA sequences diverge, it is easy to see that their maximum raw distance is ~0.75 (assuming equal nt frequencies).

- This graph shows evolutionary distance related to raw distance:



Mutational models for DNA

- Jukes-Cantor (JC) - all mutations occur at the same rate.

- Kimura 2-parameter (K2P) - transitions and transversions have separate rates.

- Generalized Time Reversible (GTR) - all changes may have separate rates.

(Models similar to GTR are also available for protein)

Jukes-Cantor

	G	A	T	C
G	1-3α	α	α	α
A	α	1-3α	α	α
T	α	α	1-3α	α
C	α	α	α	1-3α

Kimura 2-parameter

transition rate

	purines		pyrimidines	
	G	A	T	C
G	1-α-2β	α	β	β
A	α	1-α-2β	β	β
T	β	β	1-α-2β	α
C	β	β	α	1-α-2β

transversion rate

Jukes-Cantor model - distance correction

Jukes-Cantor model:

$$D = -\frac{3}{4}\ln(1 - \frac{4}{3}D_{raw})$$

D_{raw} is the raw distance (what we directly measure)

D is the corrected distance (what we want)

ln is natural log

Note - similar calculations can be made for the other models,
in particular K2P and GTR are often used (but more complex).

Distance trees - summary

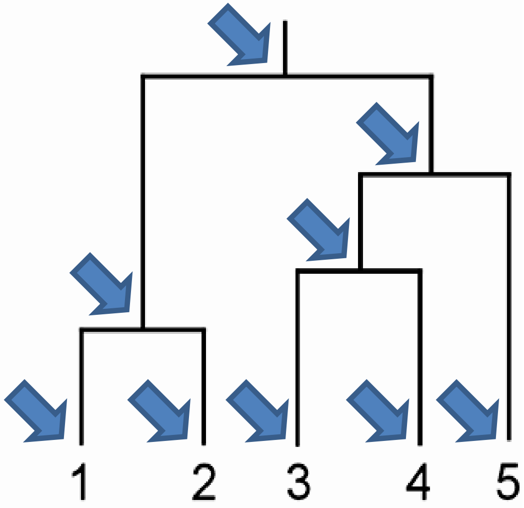
- Compute pairwise distances and convert each to a corrected distance.

- Build tree (UPGMA or neighbor-joining).

- Notice that these methods do not consider all tree topologies - they are very fast, even for large trees.

Data structure for a tree

```
class TreeNode:
    <parent node>
    <left-child node>
    <right-child node>
    <distance to parent>
```



The tree itself is made up of `TreeNode` objects, each of which is connected to other `TreeNode` objects based on its three attributes.

How do we know a node is a leaf? A root?

A leaf (or tip) has no child nodes. A root has no parent node. All the rest have all three.