Phylogenetic Trees

COMPSCI 260 – Spring 2016
Phylogenetics

- **Phylogenetics** is the study of evolutionary relationships among organisms or genes.
- In general, we are interested in the phylogeny of organisms or species.
- But oftentimes phylogenies are constructed from genes.
- **Phylogenetic trees** are used to describe phylogenies.

The purpose of **phylogenetic studies**:
- reconstruct evolutionary ties between species
- estimate the time of divergence between species since they last shared a common ancestor
Binomial nomenclature is a formal system of naming species by giving each a name composed of two parts, both of which use Latin grammatical forms, although they can be based on words from other languages.

- The first part of the name identifies the genus to which the species belongs; the second part identifies the species within the genus.

- Introduced by Carl Linnaeus in 1753

- Also called ‘scientific name’ or ‘Latin name’
What is a phylogenetic tree?

- **Binary tree** (every node has <=3 neighbors)
- **Rooted or unrooted**
- **Nodes**
  - Leaves: current species
  - Internal nodes: (hypothetical) ancestral species
- **Edges**
  - Amount of change (mutation rate) or
  - Evolutionary time
What is a phylogenetic tree

- **Binary tree** (every node has <=3 neighbors)
- **Rooted or unrooted**
- **Nodes**
  - Leaves: current species
  - Internal nodes: (hypothetical) ancestral species
- **Edges**
  - Amount of change (mutation rate) or
  - Evolutionary time
Data used to build phylogenetic trees

- Traditionally, phylogenetic trees were built from *morphological features* (e.g., beak shapes, presence of feathers, number of legs, etc).
- Today, we use mostly *molecular data* like DNA sequences and protein sequences.
- Data can be classified into 2 categories:
  - **Discrete characters**
    - Each character has a finite number of states. For example, discrete characters include the number of legs of an organism, or a column in an alignment of DNA sequences.
  - **Comparative numerical data**
    - These data encode the *distances* between objects and are usually derived from sequence data. For example, we could hypothetically say distance(man,mouse) = 500 and distance(man,chimp) = 100.
Phylogenetic trees

- NOTE: in general, different genes/proteins may give slightly different phylogenetic trees (because different genes/proteins may evolve at different rates)
- Averaging over large sets of genes/proteins does demonstrate a broad correspondence between lengths of branches and evolutionary time

- NOTE: Topology vs. (Phylogenetic) Tree

Which nodes are connected?
Speciation vs. duplication events

- Another thing to keep in mind: in general, we assume that the sequences in a phylogenetic tree have descended from an *ancestral gene A in an ancestral species*
- In other words, we assume they arose through a *speciation event*
- Another mechanism by which two sequences can diverge from a common ancestor is through a *duplication event* in the same species

![Example phylogenetic trees]

We need to make sure we are using orthologs when building phylogenetic trees!!!
Homology example: evolution of globins

- Human α-globin and human β-globin are paralogs or orthologs?
- Paralogs
- Human α-globin and mouse α-globin are homologs or orthologs?
- Both
Building a phylogenetic tree

• **Distance methods**
  – Evolutionary distances are computed for all leaf nodes, and these are used to construct trees

• **Maximum parsimony methods**
  – The tree is chosen to minimize the number of changes required to explain the data

• **Maximum likelihood methods**
  – Under a model of sequence evolution, we search for the tree which gives the highest likelihood of the data

• Bootstrapping
Building a phylogenetic tree

• We will discuss two algorithms: UPGMA and NJ
• Both algorithms require a metric that describes the distance between any 2 leaf nodes (i.e., any 2 sequences)
• How can we obtain such distances?
  – Align the 2 sequences and take the fraction of nucleotides/amino acids that are different
  – Use models of residue/nucleotide substitution (for example, the Jukes-Cantor model for DNA sequences)

Assume we have 5 sequences. We need to define a metric:

\[
d = \begin{array}{cccccc}
1 & 2 & 3 & 4 & 5 \\
1 & 0 & * & & & \\
2 & * & 0 & & & \\
3 & & & 0 & & \\
4 & & & & 0 & \\
5 & & & & & 0
\end{array}
\]
Building **rooted** phylogenetic trees

- **UPGMA** = unweighted pair group method using arithmetic averages [the name is actually more complicated than the method]
- It is basically a *hierarchical clustering algorithm*

![Diagram of phylogenetic tree construction]
Building **rooted** phylogenetic trees

- **UPGMA** = unweighted pair group method using arithmetic averages [the name is actually more complicated than the method]
- It is basically a *hierarchical clustering algorithm*
• Distance between 2 clusters (groups)?

\[
C_p \times C_q
\]

\[
d_{pq} = \frac{1}{|C_p| \times |C_q|} \sum_{i \in C_p, j \in C_q} d_{ij}
\]

(average linkage clustering)
UPGMA algorithm

- **Initialization**
  - For each sequence $i$, create cluster $C_i$
  - For each sequence $i$, create a leaf node at height 0

- **Iterate**
  - Find $i,j$ such that $d_{ij}$ is minimal
  - Define new cluster $C_k = C_i \cup C_j$ and compute $d_{kl}$ for all other clusters $l$
  - Create node $k$ (parent of $i$ and $j$) at height $d_{ij}/2$
  - Remove clusters $i$ and $j$

- **Terminate**
  - When only 2 clusters remain
  - Create root at height $d_{ij}/2$
UPGMA algorithm

Computing the distance between two clusters:

$$d_{kl} = \frac{1}{|C_k| \times |C_l|} \sum_{x \in C_k, y \in C_l} d_{xy}$$

$$C_k = C_i \cup C_j$$

How many pairs?

A more efficient way to compute the distance:

$$d_{kl} = \frac{d_{il}|C_i| + d_{jl}|C_j|}{|C_i| + |C_j|}$$

Time complexity:
Naïve implementation: $n$ iterations, $O(n^2)$ time for each iteration (to find a closest pair) => $O(n^3)$ total.
Optimal implementation: $O(n^2)$
UPGMA algorithm

- **Rooted** tree
- Leaves are at the same level
- Assumes there exists a “molecular clock” with a constant rate
UPGMA

• *Rooted* tree
• Leaves are at the same level
• Assumes there exists a “molecular clock” with a constant rate (assumes the rates of evolution are the same among different lineages)
• Can we always build such a tree?
• **NO**

![Diagram](image)

• What is the problem?
• The closest leaves are not neighbors!!!
UPGMA returns the correct tree if...

• The distance \( d \) is **ultrametric**:

\[
\forall i, j, k \quad \text{one can shuffle them to achieve} \quad d_{ij} = d_{jk} \geq d_{ik}
\]

• Is this true for the distance that characterizes this tree?

NO
Another property of UPGMA trees...

- Another property of UPGMA trees is **additivity**

\[ \forall i, j, k, l \text{ one can shuffle them to achieve } \]

\[ d_{ij} + d_{kl} = d_{ik} + d_{jl} \geq d_{il} + d_{jk} \]

Pairs of distances

Is this tree additive?

**YES**

\[ d_{13} + d_{24} = d_{14} + d_{23} \geq d_{12} + d_{34} \]
The Neighbor Joining (NJ) algorithm

- If \( d \) is **additive** but not **ultrametric**, we can use the Neighbor Joining algorithm (NJ) to build an **unrooted** phylogenetic tree.

- Idea:
  - Find two neighbors \( i \) and \( j \)
  - Join them and create a new node \( k \)
  - Recompute distances: \( d_{km} = \frac{1}{2} (d_{im} + d_{jm} - d_{ij}) \)
  - Remove nodes \( i \) and \( j \)
  - Until only two nodes remain
The Neighbor Joining (NJ) algorithm

• How do we find two neighbors?
• Find the minimum $d_{ij}$?
• NO

• We need to compensate for long edges

• We redefine the distances:

$$D_{ij} = d_{ij} - (r_i + r_j)$$
$$r_i = \frac{1}{|L| - 2} \sum_{k \in L} d_{ik}$$

where $L = \text{the set of leaves}$

• Trick: subtract the average distance to all other nodes

• Now we can choose $i,j$ that minimize $D_{ij}$
• Nodes $i$ and $j$ are guaranteed to be neighbors
Example – finding neighboring nodes

\[ \min_{i,j} d_{ij} \quad \text{vs.} \quad \min_{i,j} D_{ij} \]

\[ D_{ij} = d_{ij} - (r_i + r_j) \]

\[ r_i = \frac{1}{|L| - 2} \sum_{k \in L} d_{ik} \]
NJ algorithm

- **Initialization:**
  - $L =$ the set of leaf nodes (the set of all sequences)
- **Iteration:**
  - Choose $i, j$ in $L$ such that $D_{ij}$ is minimal
  - Define new node $k$
  - Recompute distances to all other nodes $m$: $d_{km} = \frac{1}{2}(d_{im} + d_{jm} - d_{ij})$
  - Add node $k$ to the tree
  - Add edges to $i$ and $j$ with lengths $d_{ik} = \frac{1}{2}(d_{ij} + r_i - r_j)$, $d_{jk} = d_{ij} - d_{ik}$
  - Remove $i$ and $j$ from $L$; add $k$ to $L$
- **Termination:**
  - When $|L| = 2$; add edge $(i, j)$ with length $d_{ij}$
NJ algorithm

- If the distance measure is *additive*, NJ is guaranteed to construct the correct *unrooted* tree

\[ \forall i, j, k, l \text{ one can shuffle them to achieve } d_{ij} + d_{kl} = d_{ik} + d_{jl} \geq d_{il} + d_{jk} \]

- If the distance measure is NOT additive, the NJ may not be correct
NJ algorithm

- **Time complexity:**
  - NJ on a set of \( n \) sequences requires \( O(n) \) iterations
  - At each step one has to build and search a D matrix
  - Initially the D matrix is size \( n \times n \), then at the next step it is \( (n-1) \times (n-1) \), etc.
  - This leads to an algorithm with a time complexity of \( O(n^3) \)
- What type of algorithm is this? (dynamic programming? EM? etc.)
- NJ is a *greedy algorithm*
  - At each step, NJ greedily joins that pair of sequences that will give the greatest decrease in the estimated tree length
- The main advantage of NJ: it is *fast* (it is polynomial-time)
- This makes it practical for analyzing large data sets and for bootstrapping (compared to maximum parsimony or maximum likelihood approaches, which may be computationally prohibitive)
UPGMA and NJ

Pairwise distance metric \((d)\) between sequences must be: ultrametric

The tree will be: rooted

If \(d\) is ultrametric, why don’t we build a NJ tree?

UPGMA algorithm

Neighbor Joining algorithm

ultrametric

additive

rooted

unrooted
NJ when $d$ is ultrametric

- Assume the correct tree (characterized by an ultrametric distance $d$) is:

```
   0.2
  /   \
0.2   0.2
   1    2
   |
0.3
```

- Let's apply the NJ algorithm on this tree.

Let's apply the NJ algorithm on this tree.

We can root an unrooted tree by finding an outgroup.

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0.4</td>
<td>0.8</td>
<td>0.8</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0.8</td>
<td>0.8</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0.6</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
NJ algorithm

- **Initialization:**
  - $L$ = the set of leaf nodes (the set of all sequences)

- **Iteration:**
  - Choose $i,j$ in $L$ such that $D_{ij}$ is minimal
  - Define new node $k$
  - Recompute distances to all other nodes $m$:
    $$d_{km} = \frac{1}{2}(d_{im} + d_{jm} - d_{ij})$$
  - Add node $k$ to the tree
  - Add edges to $i$ and $j$ with lengths
    $$d_{ik} = \frac{1}{2}(d_{ij} + r_i - r_j)$$
    $$d_{jk} = d_{ij} - d_{ik}$$
  - Remove $i$ and $j$ from $L$; add $k$ to $L$

- **Termination:**
  - When $|L| = 2$; add edge $(i,j)$ with length $d_{ij}$
NJ when d is ultrametric

• Assume the correct tree (characterized by an ultrametric distance d) is:

We can root an unrooted tree by finding an outgroup.

• Let’s apply the NJ algorithm on this tree.

• We will construct a “correct” but unrooted tree.

• Unrooted => rooted. How many topologies? (for n leaves)

• Unrooted => rooted. How many trees?
Building phylogenetic trees

• *Distance methods*
  – Evolutionary distances are computed for all leaf nodes, and these are used to construct trees
  – UPGMA, NJ

• *Maximum parsimony methods*
  – The tree is chosen to minimize the number of changes required to explain the data

• *Maximum likelihood methods*
  – Under a model of sequence evolution, we search for the tree which gives the highest likelihood of the data

• Bootstrapping
Bootstrapping: confidence in reconstructed trees

- **Bootstrapping** is a technique commonly used for estimating statistics or parameters when the distribution is difficult to derive analytically.
- **Idea:** sample columns from the multiple aligned group of sequences, and create many new alignments (with replacement).
- Repeat the process many times (at least 100 times).
- Generate a tree each time.
- Compute the number of times each branching point occurred (out of all the trees that were built).
- The higher the number, the more valid the branching point.
Bootstrapping: confidence in reconstructed trees

Of the 100 generated trees, we see:

41/100

Chimpanzee

41

Human

Gibbon

100

Gorilla

Orang-utan

28/100

Chimpanzee

Human

Gibbon

Orang-utan

31/100

Chimpanzee

Human

Gibbon

Orang-utan

Gorilla

In 100 of the 100 trees, gibbon and orang-utan are split from the rest.

In 41 of the 100 trees, chimp and gorilla are split from the rest.
Maximum parsimony methods

- Briefly: find the tree that can explain the observed sequences with a *minimal number of substitutions*
- For example, given the following DNA data, which tree is most parsimonious?

```
AAG  AAA  GGA  AGA
AAA  AAA  GGA  AGA
```

- Instead of building a tree, it assigns a *cost* to a given tree, and it is necessary to search through all topologies (or use a more efficient strategy) to identify the “best” tree
- Fitch’s algorithm
- Phylip package
Maximum likelihood methods

- Another method commonly used for reconstructing trees: maximum likelihood (ML)

- Idea: given a probabilistic model for nucleotide substitution (e.g., the Jukes and Cantor model for DNA sequences), pick the tree that has the highest probability of generating the observed data

- In other words, given character data $D$ and a model $M$, we want to find the tree $T$ that maximizes the likelihood $P(D | T, M)$

- Use dynamic programming and EM

- ML methods are more time intensive compared to maximum parsimony or distance-based methods

- ML methods are more sensitive at large evolutionary distances, and are considered to be the best for reconstructing phylogenetic trees
More information: