



Melbourne Bioinformatics

BIOINFORMATICS + DATA SERVICES + INFRASTRUCTURE, FOR LIFE SCIENCES TODAY



COMP90014

Algorithms for Bioinformatics

Week 5A - Evolutionary Trees I

生物学中的一种图形表示，用于展示不同物种或生物体之间的进化关系和演化历史

Evolutionary Trees I

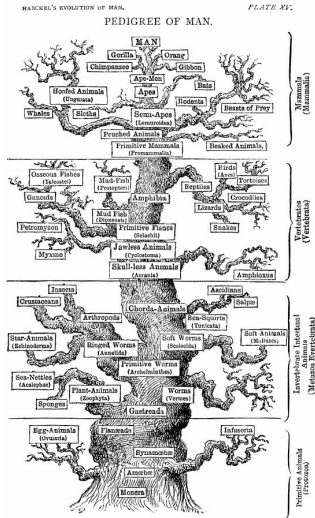
Phylogenetics

How do we build trees?

Building trees (distance methods)

- UPGMA algorithm
- Neighbor Joining algorithm

Phylogenetic trees

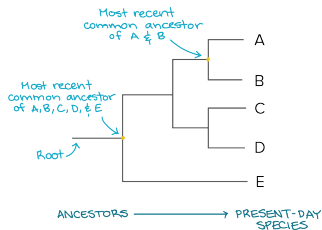


- 🌲 approaches for phylogenetic reconstruction
 - distance-based methods
 - character-based methods
- 🌲 assessing reliability/robustness

系统发育学)是研究生物种类之间的演化关系和分类学的科学领域。系统发育学的主要目标是通过分析不同物种的遗传信息、形态特征以及其他演化信息,来构建进化树,从而揭示生物种类的进化历程和亲缘关系。系统发育学在研究生物分类学、进化过程以及物种多样性的形成等方面发挥着重要作用。

Phylogenetics

分类学
系统发育学
物种形成



Taxonomy: the science of classifying organisms

Phylogenetics: describes the evolutionary relationship between species

Speciation: A population of organisms becomes separated.

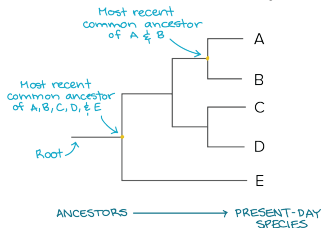
Over time, these evolve into separate species that do not cross-breed.

Phylogenetic tree

图的右侧说明了构建系统发生树的两种主要方法：

经典系统发生分析：基于形态学特征，如动物的腿的数量、鸟的喙形状等。

分子系统发生学：基于DNA、RNA或蛋白质序列的比较，特别是在不同物种中相同（同源）的序列。



Classical phylogenetic analysis:

🌲 number of legs, beak shape, etc.

Molecular phylogenetics:

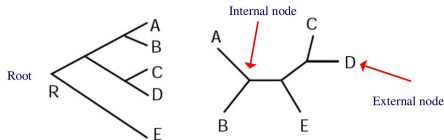
🌲 sequences

🌲 homologous sequences in different species

同源序列 不同物种

What is a tree, mathematically?

- 🌲 connected, acyclic graph 无环图
- 🌲 graph: a pair $G = (V, E)$, consisting of:
 - a set V of vertices (or nodes)
 - and a set E of edges (or branches) that connect nodes
- 🌲 acyclic: there is no path where the first and last vertices are the same
- 🌲 unrooted
- 🌲 rooted



What information is encoded on the tree?

External nodes (leaves): Taxonomic unit, e.g. different current day species.

外部节点：代表分类单元，例如不同的现代物种。每个外部节点仅与树上的一个其他节点相连。

Only connected to one other node.

Internal nodes: **Hypothetical** most recent common ancestors (MRCA).

代表假设的最近共同祖先 (MRCA, Most Recent Common Ancestor)

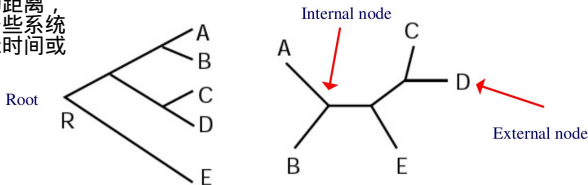
Node degree? 假设的最近共同祖先 (MRCA)

树的分支点，代表了一次分化事件，即两个或多个物种分开的地方

Edge length: Distance between leaves, e.g. time since divergence.

Topology: Relationship between leaves and nodes.

边长：表示叶子（物种）之间的距离，例如自从分歧以来的时间。在一些系统发生树中，分支的长度可以表示时间或遗传变异的量。



Evolutionary Trees I

Phylogenetics

How do we build trees?

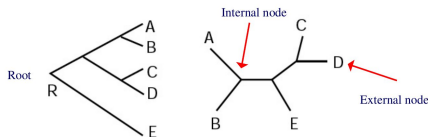
Building trees (distance methods)

- UPGMA algorithm
- Neighbor Joining algorithm

How many different trees can we construct with n sequences?

Sequences	Unrooted trees
3	1
4	3
5	15
10	> 2 000 000

Unrooted: $\prod_{i=3}^n (2i - 5)$



Rooted: (more or less?)

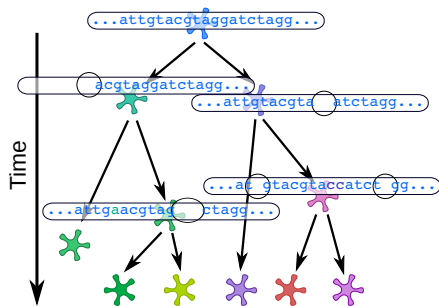
列举所有可能的树来找到最好的树不是可行的优化方法：哪棵树最小化了解释数据所需的变化数量（简约性）??是减少类群之间距离的？

🌲 enumerating all possible trees to find the best one is not feasible

🌲 optimisation approach:

- which tree minimises number of changes needed to explain data (parsimony)?
- which minimises the distance between taxa?

Inferring a phylogenetic tree: workflow

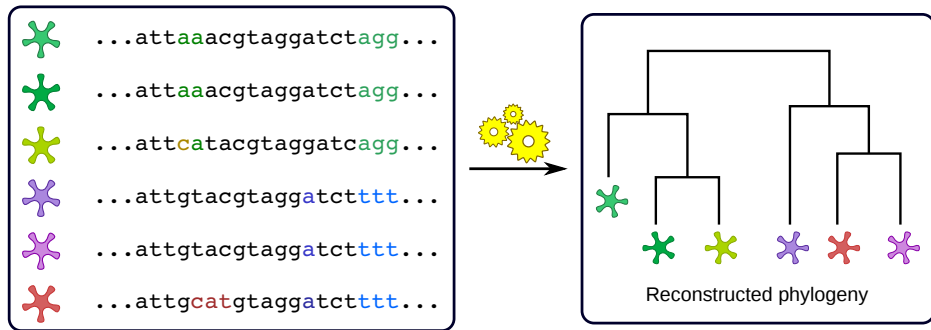


1. **data preparation:**
multiple sequence alignment
2. **tree inference/reconstruction methods:**
different algorithmic approaches
3. **tree analysis:**
assess robustness/reliability

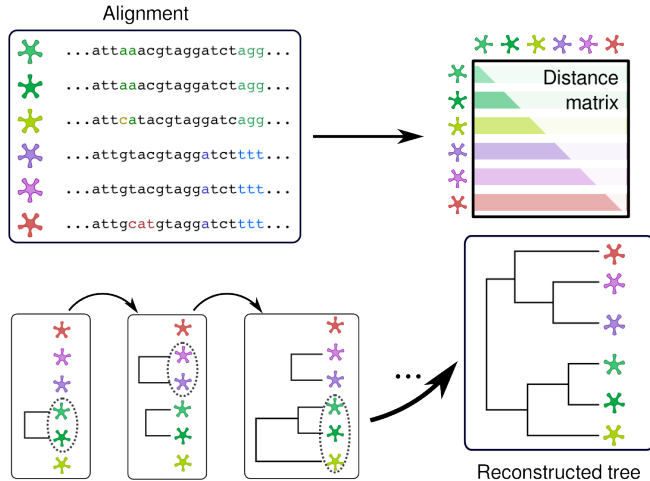
🌲 Substitutions accumulated over time tell us about evolution/genealogy

评估稳健性/可靠性

Inferring a phylogenetic tree



Agglomerative clustering



Phylogeny reconstruction algorithms

系统发育重建算法

Two types of reconstruction:

Distance-based

- 🌲 A tree is built based on the distance between items
- 🌲 Closer taxa should be more evolutionarily related

更近的分类群应该在进化上更相关

- 🌲 UPGMA
- 🌲 neighbour-joining

Character-based

- 🌲 Every taxon is described by a number of characters
e.g. number of fingers, protein sequence.
- 🌲 each has a finite number of states.
- 🌲 Goal: build the tree that best explains the character matrix
 - Optimise a objective function

- 🌲 Maximum Parsimony
- 🌲 Maximum Likelihood

Evolutionary Trees I

Phylogenetics

How do we build trees?

Building trees (distance methods)

- UPGMA algorithm
- Neighbor Joining algorithm

Distance metrics

用于比较基因序列的两种主要的距离度量方法：基于多序列比对（MSA）的方法和无需比对的方法。

Scores from multiple sequence alignments (MSA)

- 🌲 progressive (ClustalW)
- 🌲 iterative (MAFFT, MUSCLE)
- 🌲 probabilistic (Hidden Markov Models)

Alignment-free alternatives

- 🌲 k-mer count
(substring/word of length k)
- 🌲 usually much faster than alignment
but much less sensitive

k-mer计数 (k-mer count, 即长度为 k 的子串/单词) : 这种方法通过计算序列中所有可能的 k 长子串的出现频率来估算序列之间的差异, 不需要进行序列比对。通常比比对方法要快得多 (usually much faster than alignment), 但是灵敏度要低得多 (but much less sensitive), 这意味着k-mer方法可能无法检测到微小的差异, 因此在需要高精确度的情况下可能不太适用。

分
子
近
似

k-mers

Query sequences	x	ATGTGTG	y	CATGTG
Word size: 3	W_3^x	ATG TGT GTG TGT GTG	W_3^y	CAT ATG TGT GTG
Union of two sets	$W_3 = W_3^x \cup W_3^y$	CAT ATG TGT GTG		
Word counts	c_3^x	0 1 2 2	c_3^y	1 1 1 1
Euclidean distance	$\ c_3^x - c_3^y\ $	$\sqrt{(0-1)^2 + (1-1)^2 + (2-1)^2 + (2-1)^2} = \sqrt{3} = 1.73$		

🌲 all possible substrings of length k

• e.g. $k = 3$

🌲 sliding window

1. generate the set of unique k -mers
2. count k -mer frequency in each sequence
3. calculate distance

🌲 time complexity?

- how many k -mers of size k are there in a sequence of length L ?
- how many times do we move the sliding window for a sequence of length L ?

Distance-based methods

Start with each data point as a single cluster, then iteratively join clusters into bigger clusters until we reach a single cluster with all the data points

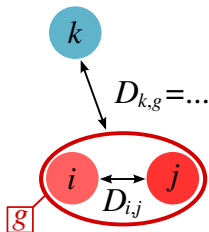
🌲 which are the next clusters to merge?

- e.g. single linkage,
complete linkage,
average linkage (UPGMA),
neighbour-joining

Agglomerative clustering:

1. Compute pairwise distances into a distance matrix D ;
2. Find the two clusters i and j with the smallest distance d_{ij} ;
3. Create a new cluster u that joins clusters i and j ;
4. Define the height (i.e. distance from leaves) of u to be $\frac{d_{ij}}{2}$;
5. Update D (d_{ku} for each $k \neq \{i, j\}$, replace i and j by new cluster u);
6. Go back to 2 until all items are grouped.

Which are the next clusters to merge?



- 🌲 Cluster $g = (i, j)$
- 🌲 Recalculate the distance matrix ($D_{k,g}$, where k are items still in individual clusters)

Single linkage: $D_{k,g} = \min(D_{k,i}, D_{k,j})$

Complete linkage: $D_{k,g} = \max(D_{k,i}, D_{k,j})$

Average linkage: $D_{k,g} = \frac{D_{k,i} + D_{k,j}}{2}$

UPGMA: Unweighted Pair Group Method with Arithmetic Mean

Branches have the same evolutionary rate (molecular clocks)

Neighbour joining: Transforms original distances to account for heterogeneous rates of evolution

Evolutionary Trees I

Phylogenetics

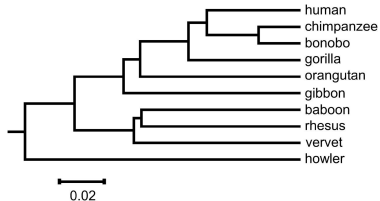
How do we build trees?

Building trees (distance methods)

- UPGMA algorithm
- Neighbor Joining algorithm

UPGMA

Unweighted Pair Group Method with Arithmetic Mean



- average linkage: mean distance between elements of each group
- generates **rooted** trees
- generates **ultrametric** trees:
 - distances from the root to every branch tip are equal
- $O(n^3)$ unoptimized

$$\frac{1}{|\mathbb{A}| \cdot |\mathbb{B}|} \sum_{x \in \mathbb{A}} \sum_{y \in \mathbb{B}} d(x, y)$$

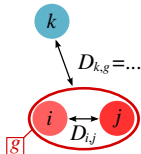
UPGMA assumes a molecular clock

ultrametric: distances from the root to every branch tip are equal

- 🌲 for this to be true, mutation rate along each branch would have to be the same
- 🌲 this is called a **molecular clock**
- 🌲 the rate of the molecular clock is **definitely not constant** in nature!
- 🌲 UPGMA still has some applications, but not in phylogenetics



UPGMA algorithm



Consider a distance matrix D , and n groups containing one item/leaf each:

1. Choose the i and j with the smallest D_{ij}
2. Create a new group ij
3. Connect i and j to a new node in the tree that correspond to the new group
4. Set the branch length to $\frac{D_{ij}}{2}$ (ultrametric)
5. Calculate the distance between the group and all existing groups (n_i = number of elements):

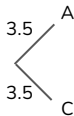
$$D_{(ij),k} = \left(\frac{n_i}{n_i + n_j}\right)D_{ik} + \left(\frac{n_j}{n_i + n_j}\right)D_{jk}$$

6. Replace the i and j columns with the new group
7. If there is only one item left stop, otherwise go to 1



Example

	<u>A</u>	B	C	D
A	0			
B	8	0		
<u>C</u>	7	9	0	
D	12	14	11	0



UPGMA algorithm

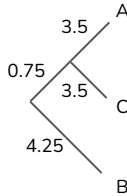
1. Choose smallest D_{ij}
2. Create a new group ij
3. Set the branch length to $\frac{D_{ij}}{2}$
4. Update distance matrix
5. Replace the i and j columns with the new group

$$M_{B(AC)} = (M_{BA} + M_{BC}) / 2 = (8 + 9) / 2 = 8.5$$

$$M_{D(AC)} = (M_{DA} + M_{DC}) / 2 = (12 + 11) / 2 = 11.5$$

Example

	<u>AC</u>	B	D
AC	0		
<u>B</u>	8.5	0	
D	11.5	14	0



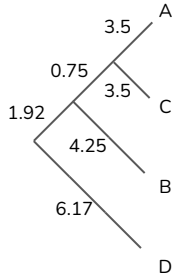
UPGMA algorithm

1. Choose smallest D_{ij}
2. Create a new group ij
3. Set the branch length to $\frac{D_{ij}}{2}$
4. Update distance matrix
5. Replace the i and j columns with the new group

$$M_{D(ABC)} = (M_{AD} + M_{BD} + M_{CD}) / 3 = (12 + 14 + 11) / 3 = 12.33$$

Example

	ABC	D
ABC	0	
D	12.33	0

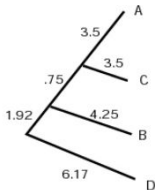


UPGMA algorithm

1. Choose smallest D_{ij}
2. Create a new group ij
3. Set the branch length to $\frac{D_{ij}}{2}$
4. Update distance matrix
5. Replace the i and j columns with the new group

Example

	ABC	D
ABC	0	
D	12.33	0



- 🌲 UPGMA assumes that the rates of evolution are the same among different items
- 🌲 We don't use this method for phylogenetic tree reconstruction (unless we believe the assumption...!)

UPGMA algorithm

1. Choose smallest D_{ij}
2. Create a new group ij
3. Set the branch length to $\frac{D_{ij}}{2}$
4. Update distance matrix
5. Replace the i and j columns with the new group

Evolutionary Trees I

Phylogenetics

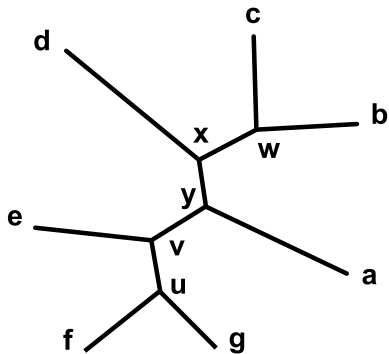
How do we build trees?

Building trees (distance methods)

- UPGMA algorithm
- Neighbor Joining algorithm

Neighbour joining

即不假设不同物种的进化速率一致。
超度树是指树中所有叶节点到根节点的路径长度相等。



- most widely-used distance based method for phylogenetic reconstruction
- trees are unrooted
- does not assume a molecular clock
- does not produce ultrametric trees
- UPGMA: constructs a larger cluster C by merging two nearest clusters A and B
- neighbour joining: distance from A and B to other clusters should be as large as possible
 - look for nodes that are **close to each other** and **far from everything else**
 - subtract the averaged distances to all other leaves
 - compensate for long edges
- $O(n^3)$ unoptimized

在选择合并的节点时，寻找彼此距离最近且与其他所有节点的距离都较远的节点对。

在这个过程中，会从每个节点到其他所有节点的平均距离中减去这个平均值，以此来补偿边的长度。

Neighbour joining algorithm

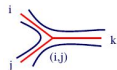
Consider a distance matrix D :

$$u_i = \sum_{j:j \neq i}^n \frac{D_{ij}}{n-2}$$



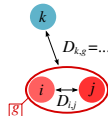
$$v_i = \frac{1}{2}D_{ij} + \frac{1}{2}(u_i - u_j)$$

$$v_j = \frac{1}{2}D_{ij} + \frac{1}{2}(u_j - u_i)$$

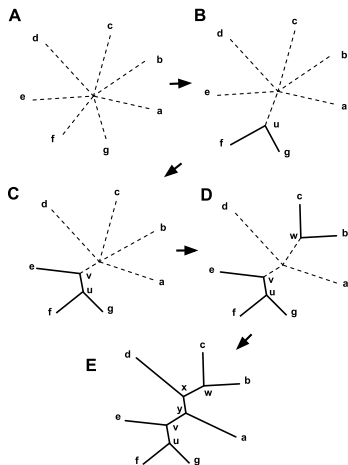


$$D_{(ij)k} = \frac{D_{ik} + D_{jk} - D_{ij}}{2}$$

1. Calculate the “average” distance to other nodes/clusters for each leaf
2. Choose i and j to minimize $D_{ij} - u_i - u_j$
(Nodes that are close to each other, and far from everything else)
3. Join i and j to create a new node (i,j) and calculate the new branch lengths
4. Compute distance between leaves and the new group
5. Replace the i and j leaves with the new node (i,j)
6. Continue until two nodes remain



Neighbour joining, graphically



1. begin with a star tree
2. minimise $D_{ij} - u_i - u_j$
3. resolve pairs
4. update distance matrices
5. go to 2

🌲 neighbour joining does not assume all sequences evolve at the same rate

Distance-based methods

Advantages

- 🌲 simple
- 🌲 flexible
- 🌲 fast and scalable

Limitations

- 🌲 sensitive to distance method
- 🌲 evolutionary rates are not estimated
- 🌲 no measure of uncertainty for the tree obtained

Thank you!

Today: Evolutionary Trees I

Next time: Evolutionary Trees II