Introduction to Phylogenetics

Yao-ban Chan and Rob Lanfear

Melbourne Integrative Genomics workshop

What is phylogenetics?

Phylogenetics is the study of evolution, and evolutionary relationships between species.

What is phylogenetics?

Phylogenetics is the study of evolution, and evolutionary relationships between species.

It traces its origin back to Darwin, in his famous book *On the Origin of Species* (1859).

What is phylogenetics?

Phylogenetics is the study of evolution, and evolutionary relationships between species.

It traces its origin back to Darwin, in his famous book *On the Origin of Species* (1859).

The core idea is that all life on Earth evolved from a single ancestor, a long, long time ago.

What is phylogenetics?

Phylogenetics is the study of evolution, and evolutionary relationships between species.

It traces its origin back to Darwin, in his famous book *On the Origin of Species* (1859).

The core idea is that all life on Earth evolved from a single ancestor, a long, long time ago (in a galaxy far, far away).

What is phylogenetics?

At various points in time, new species were created, and began evolving separately from each other. This process is known as speciation.

What is phylogenetics?

At various points in time, new species were created, and began evolving separately from each other. This process is known as speciation.

However, some species can die out (extinction).

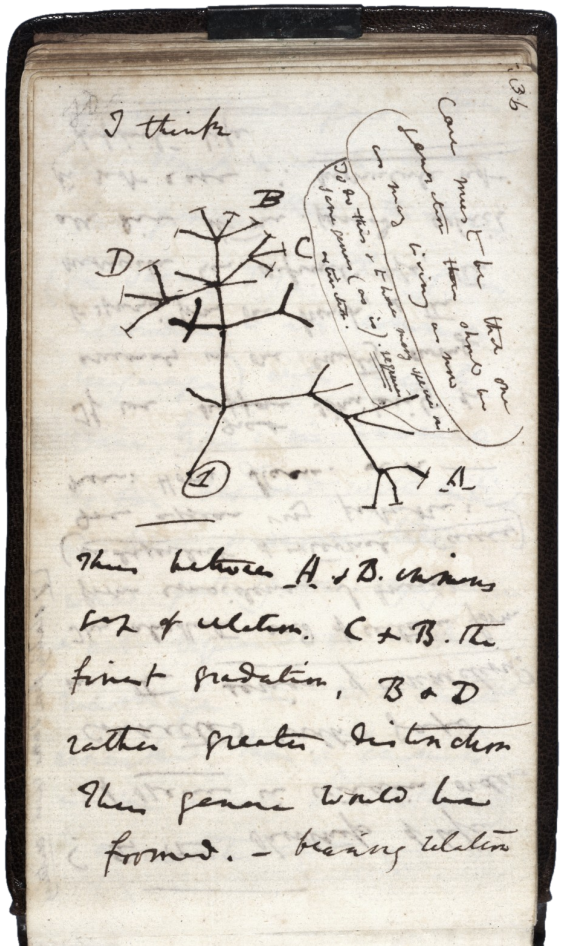
What is phylogenetics?

At various points in time, new species were created, and began evolving separately from each other. This process is known as speciation.

However, some species can die out (extinction).

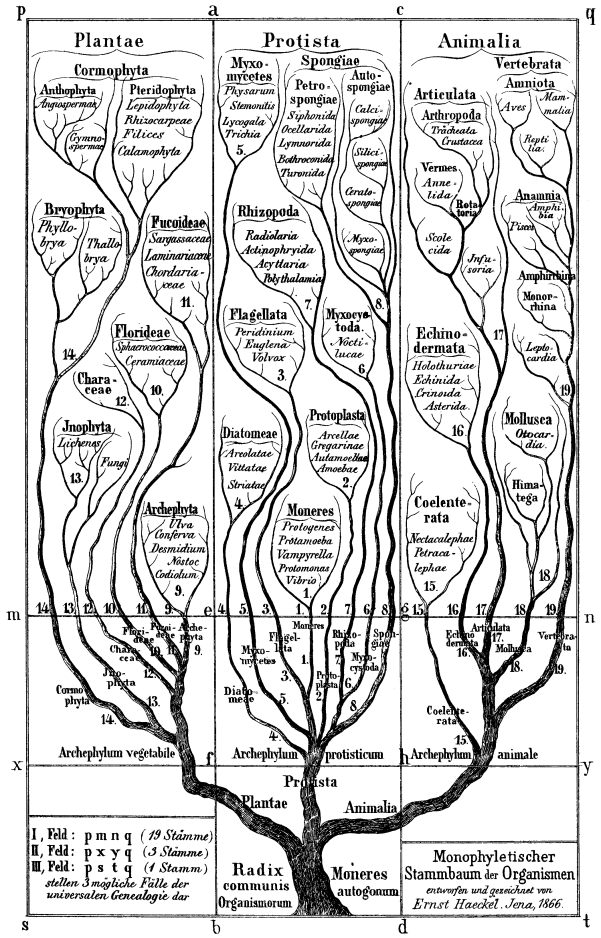
The combination of these two processes produces what we call the Tree of Life.

Ancient phylogenetics



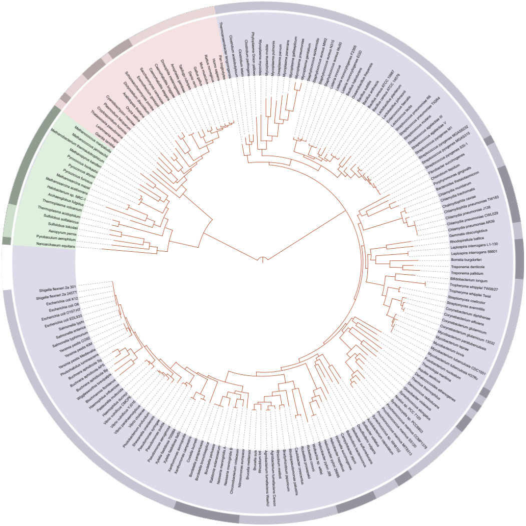
(Darwin's notes, 1837)

Ancient phylogenetics



(Haeckel, 1866)

A more modern tree



Phylogenetics of what?

Phylogenetics was originally conceived as a way to represent the evolutionary history of species.

Phylogenetics of what?

Phylogenetics was originally conceived as a way to represent the evolutionary history of species.

However, it is useful for much more than that. We can represent the evolutionary history of:

- Individual genes;

Phylogenetics of what?

Phylogenetics was originally conceived as a way to represent the evolutionary history of species.

However, it is useful for much more than that. We can represent the evolutionary history of:

- Individual genes;
- Samples of viruses or bacteria;

Phylogenetics of what?

Phylogenetics was originally conceived as a way to represent the evolutionary history of species.

However, it is useful for much more than that. We can represent the evolutionary history of:

- Individual genes;
- Samples of viruses or bacteria;
- Individual people (population genetics).

Phylogenetics of what?

Phylogenetics was originally conceived as a way to represent the evolutionary history of species.

However, it is useful for much more than that. We can represent the evolutionary history of:

- Individual genes;
- Samples of viruses or bacteria;
- Individual people (population genetics).

In many cases, the samples also come from different times (Covid-19 trees or ancient DNA).

Trees

(Mathematical) trees are the basic structure used to represent evolutionary history.

Trees

(Mathematical) trees are the basic structure used to represent evolutionary history.

Mathematically, they consist of a set of nodes (or vertices), and a set of edges (or branches) joining the nodes.

Trees

(Mathematical) trees are the basic structure used to represent evolutionary history.

Mathematically, they consist of a set of nodes (or vertices), and a set of edges (or branches) joining the nodes.

A traditional mathematical definition is that a tree is a graph without cycles.

Trees

(Mathematical) trees are the basic structure used to represent evolutionary history.

Mathematically, they consist of a set of nodes (or vertices), and a set of edges (or branches) joining the nodes.

A traditional mathematical definition is that a tree is a graph without cycles.

We will consider only binary trees, where each internal (non-root) node has degree 3.

Phylogenetic trees

In a phylogenetic tree:

- Leaf nodes (nodes with degree 1) represent current-day (extant) samples, or taxon (plural taxa);

Phylogenetic trees

In a phylogenetic tree:

- Leaf nodes (nodes with degree 1) represent current-day (extant) samples, or taxon (plural taxa);
- Edges represent (hypothetical) ancestors;

Phylogenetic trees

In a phylogenetic tree:

- Leaf nodes (nodes with degree 1) represent current-day (extant) samples, or taxon (plural taxa);
- Edges represent (hypothetical) ancestors;
- Internal nodes represent speciation events;

Phylogenetic trees

In a phylogenetic tree:

- Leaf nodes (nodes with degree 1) represent current-day (extant) samples, or taxon (plural taxa);
- Edges represent (hypothetical) ancestors;
- Internal nodes represent speciation events;
- Extinction events are not represented.

Phylogenetic trees

In a phylogenetic tree:

- Leaf nodes (nodes with degree 1) represent current-day (extant) samples, or taxon (plural taxa);
- Edges represent (hypothetical) ancestors;
- Internal nodes represent speciation events;
- Extinction events are not represented.

The branching structure of the tree is called its topology.

Phylogenetic trees

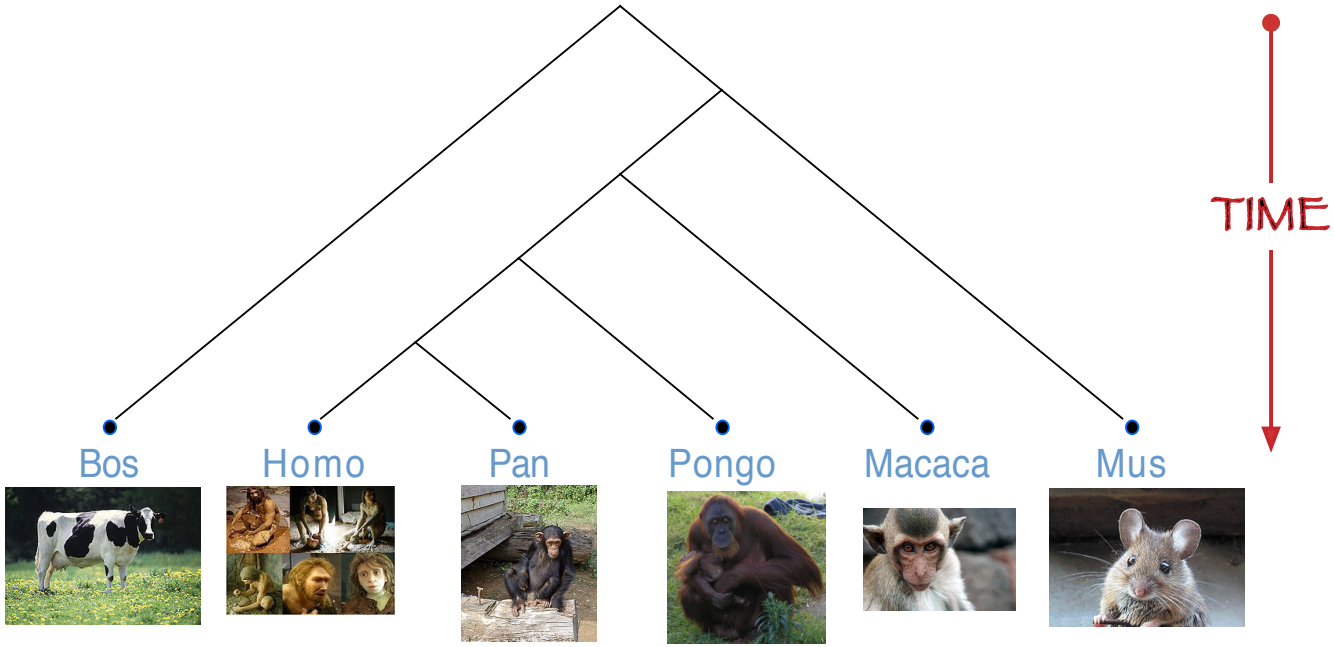
In a phylogenetic tree:

- Leaf nodes (nodes with degree 1) represent current-day (extant) samples, or taxon (plural taxa);
- Edges represent (hypothetical) ancestors;
- Internal nodes represent speciation events;
- Extinction events are not represented.

The branching structure of the tree is called its topology.

Additionally, each branch may have a number corresponding to it, representing evolutionary distance.

Phylogenetic trees



Rooted vs. unrooted trees

A phylogenetic tree can be rooted or unrooted.

Rooted vs. unrooted trees

A phylogenetic tree can be rooted or unrooted.

A rooted tree provides an explicit time direction:

Rooted vs. unrooted trees

A phylogenetic tree can be rooted or unrooted.

A rooted tree provides an explicit time direction:

- There is a node (with degree 2) which depicts the most recent common ancestor (MRCA).

Rooted vs. unrooted trees

A phylogenetic tree can be rooted or unrooted.

A rooted tree provides an explicit time direction:

- There is a node (with degree 2) which depicts the most recent common ancestor (MRCA).
- Each branch represents the evolution of one sample from ancestor to descendant.

Rooted vs. unrooted trees

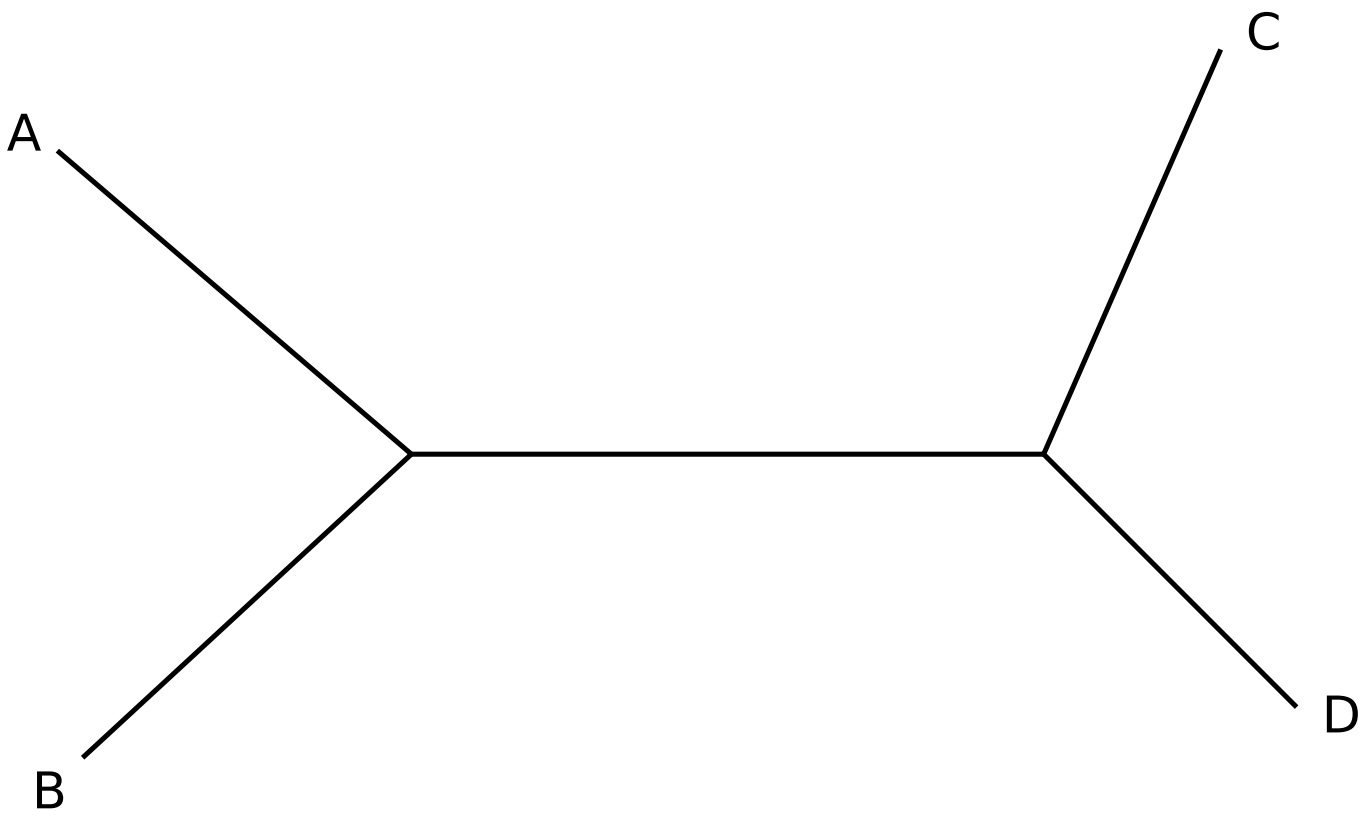
An unrooted tree provides no explicit time direction — it does not state which end of the branch is the ancestor and which is the descendant.

Rooted vs. unrooted trees

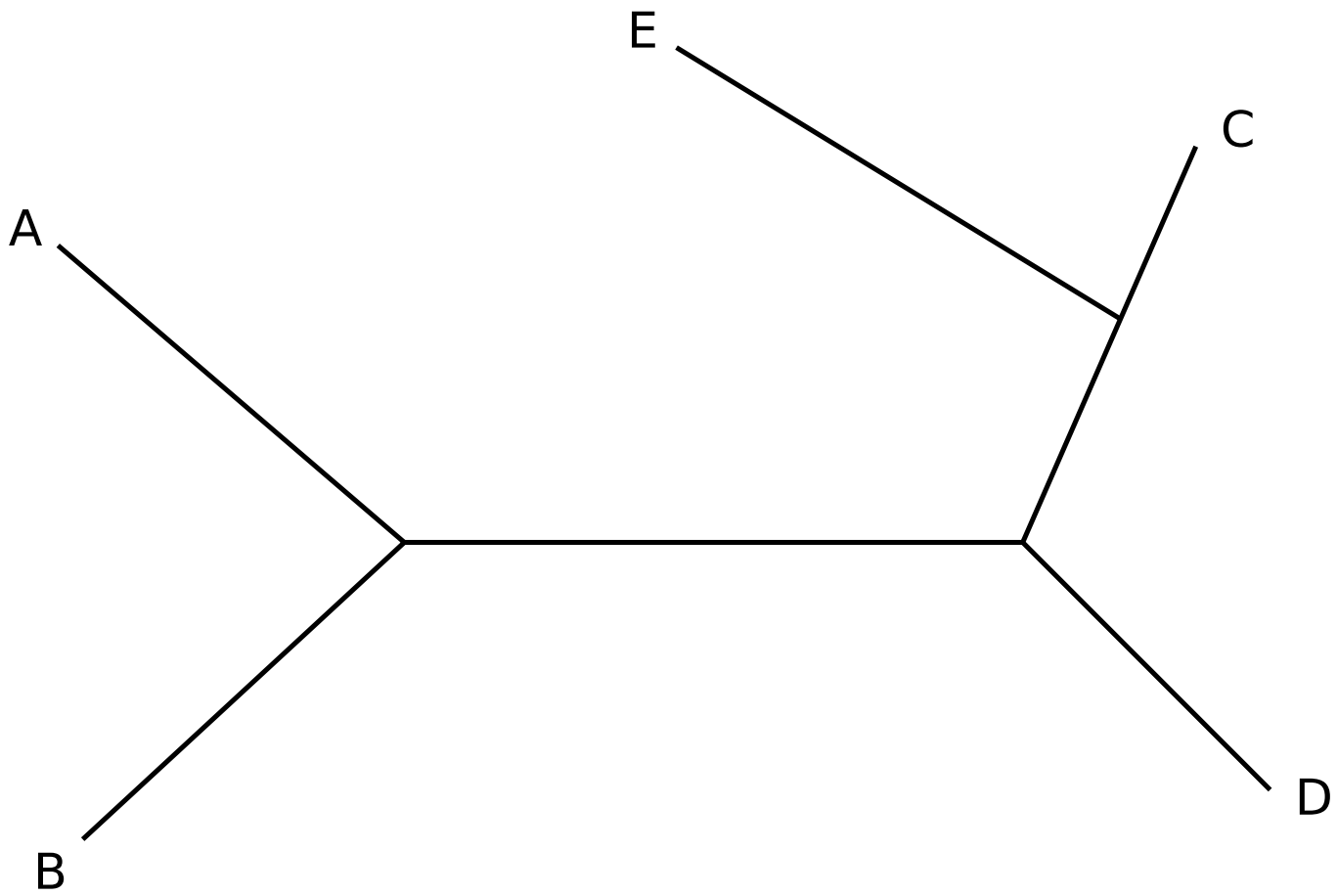
An unrooted tree provides no explicit time direction — it does not state which end of the branch is the ancestor and which is the descendant.

We can root an unrooted tree by means of an outgroup — a sample which we "know" to be more distantly related to any of the original samples than any other.

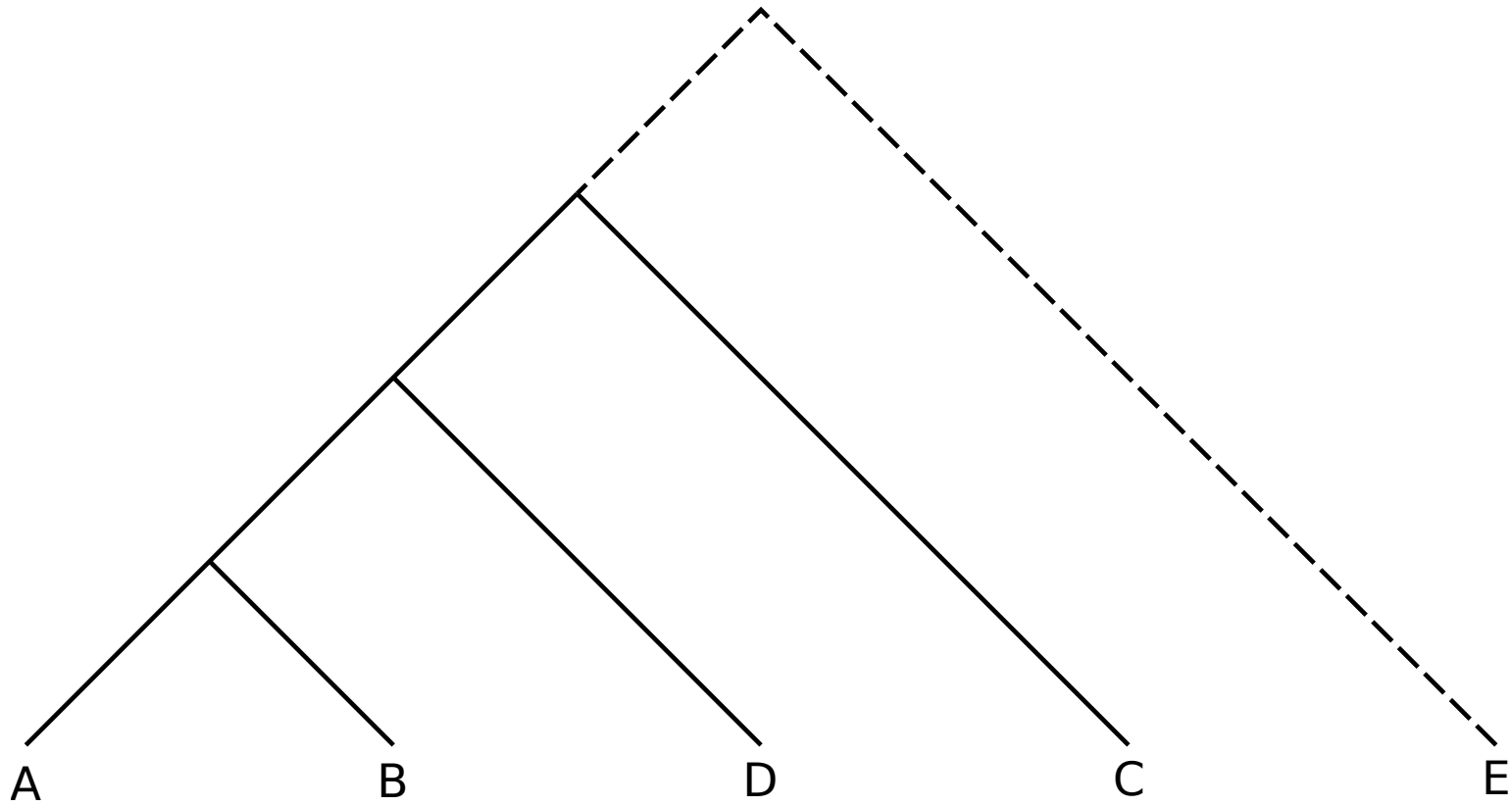
Rooting an unrooted tree



Rooting an unrooted tree



Rooting an unrooted tree



The molecular clock hypothesis

The molecular clock hypothesis states that evolution happens at a constant rate through time.

The molecular clock hypothesis

The molecular clock hypothesis states that evolution happens at a constant rate through time.

Applied to a phylogenetic tree, this means that the tree is ultrametric: the distance from the root to every leaf is the same.

The molecular clock hypothesis

The molecular clock hypothesis states that evolution happens at a constant rate through time.

Applied to a phylogenetic tree, this means that the tree is ultrametric: the distance from the root to every leaf is the same.

Under this hypothesis, branch lengths can be interpreted as representing time (scaled in some fashion). But this does not always hold!

The molecular clock hypothesis

The molecular clock hypothesis states that evolution happens at a constant rate through time.

Applied to a phylogenetic tree, this means that the tree is ultrametric: the distance from the root to every leaf is the same.

Under this hypothesis, branch lengths can be interpreted as representing time (scaled in some fashion). But this does not always hold!

Rates of evolution can vary among samples, through time, and between sites.

Newick format

A common way to store trees in text is Newick format.

Newick format

A common way to store trees in text is Newick format.

Each node is represented by a list of its children, separated by "," and enclosed in "()". The tree must end in ";".

Newick format

A common way to store trees in text is Newick format.

Each node is represented by a list of its children, separated by "`,`" and enclosed in "`()`". The tree must end in "`;`".

Example. The tree above can be written as `(((A,B),D),C);`

Newick format

A common way to store trees in text is Newick format.

Each node is represented by a list of its children, separated by "`,`" and enclosed in "`()`". The tree must end in "`;`".

Example. The tree above can be written as `(((A,B),D),C);`

Branch lengths can also be included after a "`:`", for example `(((A:1,B:1):2,D:3):1,C:4);`

Newick format

A common way to store trees in text is Newick format.

Each node is represented by a list of its children, separated by "`,`" and enclosed in "`()`". The tree must end in "`;`".

Example. The tree above can be written as `(((A,B),D),C);`

Branch lengths can also be included after a "`:`", for example `(((A:1,B:1):2,D:3):1,C:4);`

Internal nodes can also be named: `(((A:1,B:1)E:2,D:3)F:1,C:4)G;`

How do we find a tree?

In the early days of phylogenetics, trees were drawn using morphological traits (number of limbs, wings, hair, etc.).

How do we find a tree?

In the early days of phylogenetics, trees were drawn using morphological traits (number of limbs, wings, hair, etc.).

This only gives a small number of traits, and furthermore these traits are highly susceptible to selective pressures.

How do we find a tree?

In the early days of phylogenetics, trees were drawn using morphological traits (number of limbs, wings, hair, etc.).

This only gives a small number of traits, and furthermore these traits are highly susceptible to selective pressures.

Nowadays, we use gene and genome sequences, which provide much more data.

How do we find a tree?

Methods to find a phylogenetic tree fall into two paradigms:

How do we find a tree?

Methods to find a phylogenetic tree fall into two paradigms:

- Distance-based methods calculate a matrix of pairwise distances between the sequences, and use it to form a tree;

How do we find a tree?

Methods to find a phylogenetic tree fall into two paradigms:

- Distance-based methods calculate a matrix of pairwise distances between the sequences, and use it to form a tree;
- Character-based methods consider each character in the alignment and try to find a tree which explains these the best.

How do we find a tree?

Methods to find a phylogenetic tree fall into two paradigms:

- Distance-based methods calculate a matrix of pairwise distances between the sequences, and use it to form a tree;
- Character-based methods consider each character in the alignment and try to find a tree which explains these the best.
  - Maximum likelihood methods attempt to find the tree which is the most likely under a statistical model;

How do we find a tree?

Methods to find a phylogenetic tree fall into two paradigms:

- Distance-based methods calculate a matrix of pairwise distances between the sequences, and use it to form a tree;
- Character-based methods consider each character in the alignment and try to find a tree which explains these the best.
  - Maximum likelihood methods attempt to find the tree which is the most likely under a statistical model;
  - Bayesian methods apply a prior, and sample trees from the posterior distribution.

Alignments

First, we need to find an alignment of the sequences.

Alignments

First, we need to find an alignment of the sequences.

```
...CTGCGATTACACCGGAGTCGACCTAG...
...CTGCTATTACCCGGGAGTAGACCTAG...
...ATTCAATGACATTGGGATTACCCTAG...
...ATTCCATGACATTCGCTAGCGTCTAG...
```

Alignments

First, we need to find an alignment of the sequences.

```
...CTGCGATTACACCGGAGTCGACCTAG...
...CTGCTATTACCCGGGAGTAGACCTAG...
...ATTCAATGACATTGGGATTACCCTAG...
...ATTCCATGACATTCGCTAGCGTCTAG...
```

An alignment determines which sites are considered to be homologous among the sequences.

Alignments

First, we need to find an alignment of the sequences.

```
...CTGCGATTACACCGGAGTCGACCTAG...
...CTGCTATTACCCGGGAGTAGACCTAG...
...ATTCAATGACATTGGGATTACCCTAG...
...ATTCCATGACATTCGCTAGCGTCTAG...
```

An alignment determines which sites are considered to be homologous among the sequences.

This is very important as a first step in phylogenetic inference!

Calculating distances

To apply a distance-based method, we must first calculate a matrix of distances between each pair of taxa.

Calculating distances

To apply a distance-based method, we must first calculate a matrix of distances between each pair of taxa.

The simplest way is to use mismatch distance: the distance between two sequences is the number (or the proportion) of nucleotides which are different between the two sequences.

Calculating distances

To apply a distance-based method, we must first calculate a matrix of distances between each pair of taxa.

The simplest way is to use mismatch distance: the distance between two sequences is the number (or the proportion) of nucleotides which are different between the two sequences.

This works well for a rough calculation, or for sequences which are not too far from each other.

Calculating distances

```
A     ...CTGCGATTACACCGGAGTCGACCTAG...
B     ...CTGCTATTACCCGGGAGTAGACCTAG...
C     ...ATTCAATGACATTGGGATTACCCTAG...
D     ...ATTCCATGACATTCGCTAGCGTCTAG...
```

$$\begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{pmatrix} 0 & & & \\ & 0 & & \\ & & 0 & \\ & & & 0 \end{pmatrix} \end{array}$$

Calculating distances

```
A     ...CTGCGATTACACCGGAGTCGACCTAG...
B     ...CTGCTATTACCCGGGAGTAGACCTAG...
C     ...ATTCAATGACATTGGGATTACCCTAG...
D     ...ATTCCATGACATTCGCTAGCGTCTAG...
```

$$\begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{pmatrix} 0 & 3 & & \\ 3 & 0 & & \\ & & 0 & \\ & & & 0 \end{pmatrix} \end{array}$$

Calculating distances

```
A     ...CTGCGATTACACCGGAGTCGACCTAG...
B     ...CTGCTATTACCCGGGAGTAGACCTAG...
C     ...ATTCAATGACATTGGGATTACCCTAG...
D     ...ATTCCATGACATTCGCTAGCGTCTAG...
```

$$
\begin{array}{c} \\ A \\ B \\ C \\ D \end{array}
\begin{array}{c}
\begin{array}{cccc} A & B & C & D \end{array} \\
\begin{pmatrix} 0 & 3 & 10 & \\ 3 & 0 & & \\ 10 & & 0 & \\ & & & 0 \end{pmatrix}
\end{array}
$$

Calculating distances

```
A     ...CTGCGATTACACCGGAGTCGACCTAG...
B     ...CTGCTATTACCCGGGAGTAGACCTAG...
C     ...ATTCAATGACATTGGGATTACCCTAG...
D     ...ATTCCATGACATTCGCTAGCGTCTAG...
```

$$\begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{pmatrix} 0 & 3 & 10 & 14 \\ 3 & 0 & 11 & 15 \\ 10 & 11 & 0 & 9 \\ 14 & 15 & 9 & 0 \end{pmatrix} \end{array}$$

Calculating distances

For longer distances, there is a small problem: for our distance metric, we want to have additive distances, i.e., they should satisfy the relationship

$$d(a, b) + d(b, c) = d(a, c)$$

when b is on the evolutionary path from a to c.

Calculating distances

For longer distances, there is a small problem: for our distance metric, we want to have additive distances, i.e., they should satisfy the relationship

$$d(a, b) + d(b, c) = d(a, c)$$

when b is on the evolutionary path from a to c.

Mismatch distance does not have this property, because of back-mutations: nucleotides which mutate twice.

Calculating distances

For longer distances, there is a small problem: for our distance metric, we want to have additive distances, i.e., they should satisfy the relationship

$$d(a, b) + d(b, c) = d(a, c)$$

when b is on the evolutionary path from a to c.

Mismatch distance does not have this property, because of back-mutations: nucleotides which mutate twice.

This can be corrected using a statistical model of evolution, such as the Jukes-Cantor model.

From distances to trees

Now suppose we have a pairwise distance matrix. How do we calculate a phylogenetic tree?

From distances to trees

Now suppose we have a pairwise distance matrix. How do we calculate a phylogenetic tree?

A classic method is neighbour-joining (NJ, Saitou and Nei, 1987).

From distances to trees

Now suppose we have a pairwise distance matrix. How do we calculate a phylogenetic tree?

A classic method is neighbour-joining (NJ, Saitou and Nei, 1987).

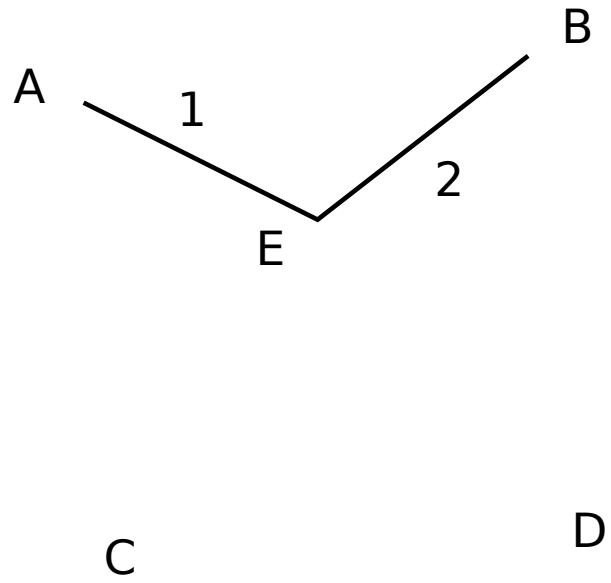
The idea of this method is to recursively join taxa (extant or ancestral) which are "close" to each other, until all taxa have been joined into one tree.

Neighbour-joining example

$$\begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{pmatrix} 0 & 3 & 10 & 14 \\ 3 & 0 & 11 & 15 \\ 10 & 11 & 0 & 9 \\ 14 & 15 & 9 & 0 \end{pmatrix} \end{array}$$

Neighbour-joining example

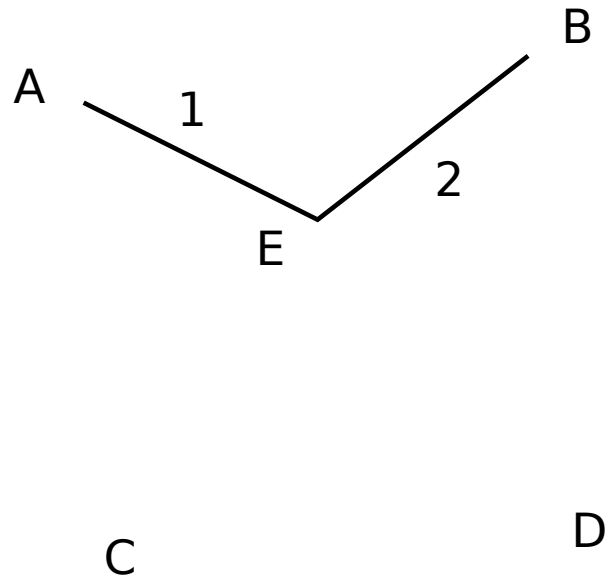
$$\begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{pmatrix} 0 & 3 & 10 & 14 \\ 3 & 0 & 11 & 15 \\ 10 & 11 & 0 & 9 \\ 14 & 15 & 9 & 0 \end{pmatrix} \end{array}$$



We join A and B, and create a new node E.

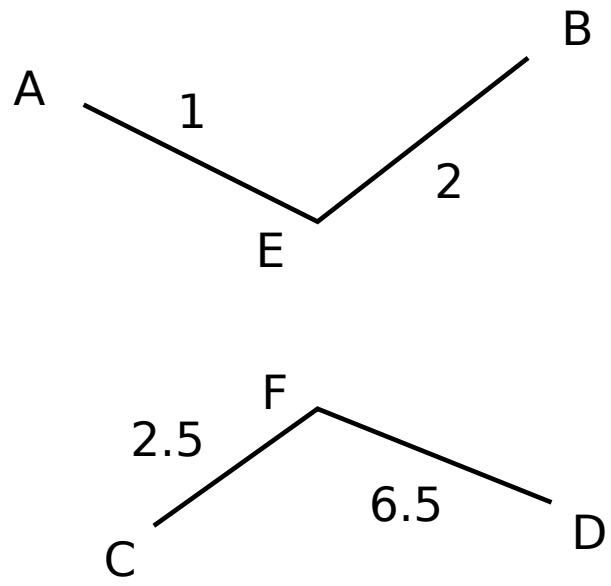
Neighbour-joining example

$$\begin{array}{c} \\ E \\ C \\ D \end{array} \begin{array}{c} \begin{array}{ccc} E & C & D \end{array} \\ \begin{pmatrix} 0 & 9 & 13 \\ 9 & 0 & 9 \\ 13 & 9 & 0 \end{pmatrix} \end{array}$$



Neighbour-joining example

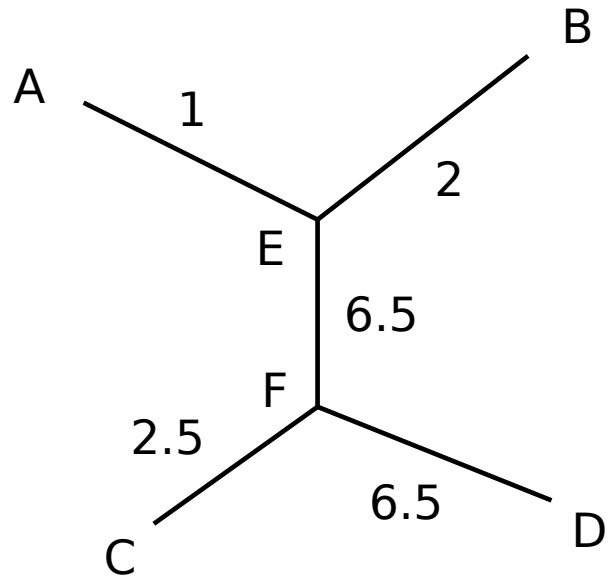
$$\begin{array}{c} \\ E \\ C \\ D \end{array} \begin{array}{c} \begin{array}{ccc} E & C & D \end{array} \\ \begin{pmatrix} 0 & 9 & 13 \\ 9 & 0 & 9 \\ 13 & 9 & 0 \end{pmatrix} \end{array}$$



We join C and D, and create a new node F.

Neighbour-joining example

$$\begin{array}{c} \\ E \\ C \\ D \end{array} \begin{array}{c} \begin{array}{ccc} E & C & D \end{array} \\ \begin{pmatrix} 0 & 9 & 13 \\ 9 & 0 & 9 \\ 13 & 9 & 0 \end{pmatrix} \end{array}$$



We join C and D, and create a new node F.

We have two nodes left, so we join them and stop.

Neighbour-joining: advantages and disadvantages

Neighbour-joining is a "quick and easy" way to rapidly generate a phylogenetic tree.

Neighbour-joining: advantages and disadvantages

Neighbour-joining is a "quick and easy" way to rapidly generate a phylogenetic tree.

Its main advantage is that it is (really) fast: a simple implementation is $O(n^3)$, but this can be trimmed down to $O(n^2)$.

Neighbour-joining: advantages and disadvantages

Neighbour-joining is a "quick and easy" way to rapidly generate a phylogenetic tree.

Its main advantage is that it is (really) fast: a simple implementation is $O(n^3)$, but this can be trimmed down to $O(n^2)$.

If the distance matrix actually comes from a tree, neighbour-joining is guaranteed to reproduce it.

Neighbour-joining: advantages and disadvantages

Neighbour-joining is a "quick and easy" way to rapidly generate a phylogenetic tree.

Its main advantage is that it is (really) fast: a simple implementation is $O(n^3)$, but this can be trimmed down to $O(n^2)$.

If the distance matrix actually comes from a tree, neighbour-joining is guaranteed to reproduce it.

However, if not, weird things (negative branch lengths!) can happen.

Neighbour-joining: advantages and disadvantages

Neighbour-joining is a "quick and easy" way to rapidly generate a phylogenetic tree.

Its main advantage is that it is (really) fast: a simple implementation is $O(n^3)$, but this can be trimmed down to $O(n^2)$.

If the distance matrix actually comes from a tree, neighbour-joining is guaranteed to reproduce it.

However, if not, weird things (negative branch lengths!) can happen.

It also doesn't necessarily produce ultrametric trees.

Neighbour-joining: advantages and disadvantages

Philosophically, calculating distances from sequences (and then throwing away the sequences) loses some information.

Neighbour-joining: advantages and disadvantages

Philosophically, calculating distances from sequences (and then throwing away the sequences) loses some information.

As a stand-alone method, neighbour-joining has largely been superseded by more accurate character-based methods.

Neighbour-joining: advantages and disadvantages

Philosophically, calculating distances from sequences (and then throwing away the sequences) loses some information.

As a stand-alone method, neighbour-joining has largely been superseded by more accurate character-based methods.

However it is still useful for very large datasets, or more generally as a way to get an initial tree which can be further improved by other methods.

Character-based methods

Character-based methods tend to be hill-climbing algorithms: starting from an initial tree, perform a heuristic search for a tree which optimises a given criterion.

Character-based methods

Character-based methods tend to be hill-climbing algorithms: starting from an initial tree, perform a heuristic search for a tree which optimises a given criterion.

This is not guaranteed to find a global optimum, but given the number of possible trees, it is the best we can do.

Character-based methods

Character-based methods tend to be hill-climbing algorithms: starting from an initial tree, perform a heuristic search for a tree which optimises a given criterion.

This is not guaranteed to find a global optimum, but given the number of possible trees, it is the best we can do.

They require two parts: how to evaluate any given tree, and how to propose a new tree to evaluate.

Maximum likelihood

The dominant way of evaluating a tree is to formulate a statistical model of evolution and calculate the likelihood of the alignment given the tree.

Maximum likelihood

The dominant way of evaluating a tree is to formulate a statistical model of evolution and calculate the likelihood of the alignment given the tree.

We then search for the tree which gives us the maximum likelihood.

Maximum likelihood

The dominant way of evaluating a tree is to formulate a statistical model of evolution and calculate the likelihood of the alignment given the tree.

We then search for the tree which gives us the maximum likelihood.

This is a time-honoured statistical technique, and it performs very well in practice.

Maximum likelihood

The dominant way of evaluating a tree is to formulate a statistical model of evolution and calculate the likelihood of the alignment given the tree.

We then search for the tree which gives us the maximum likelihood.

This is a time-honoured statistical technique, and it performs very well in practice.

Of course, it depends on having a realistic (and simple!) model of evolution.

Statistical models of evolution

The traditional statistical model of evolution is that each character evolves independently according to a continuous-time Markov process.

Statistical models of evolution

The traditional statistical model of evolution is that each character evolves independently according to a continuous-time Markov process.

The most basic model is the Jukes-Cantor model (Jukes and Cantor, 1969).

Statistical models of evolution

The traditional statistical model of evolution is that each character evolves independently according to a continuous-time Markov process.

The most basic model is the Jukes-Cantor model (Jukes and Cantor, 1969).

In this model, every nucleotide is equally likely, and they transition to each other at identical rates.

Statistical models of evolution

The traditional statistical model of evolution is that each character evolves independently according to a continuous-time Markov process.

The most basic model is the Jukes-Cantor model (Jukes and Cantor, 1969).

In this model, every nucleotide is equally likely, and they transition to each other at identical rates.

The mathematics of this is expressed as a continuous-time Markov process.

Jukes-Cantor model

After some math, the transition probabilities for the Jukes-Cantor model are given by:

$$P_{i,i}(t) = \frac{1}{4}\left(1 + 3e^{-4\alpha t}\right),$$
$$P_{i,j}(t) = \frac{1}{4}\left(1 - e^{-4\alpha t}\right) \qquad \text{for } i \neq j.$$

Jukes-Cantor model

After some math, the transition probabilities for the Jukes-Cantor model are given by:

$$P_{i,i}(t) = \frac{1}{4}\left(1 + 3e^{-4\alpha t}\right),$$
$$P_{i,j}(t) = \frac{1}{4}\left(1 - e^{-4\alpha t}\right) \qquad \text{for } i \neq j.$$

We can invert the Jukes-Cantor model to calculate an improved distance measure between sequences.

Jukes-Cantor model

After some math, the transition probabilities for the Jukes-Cantor model are given by:

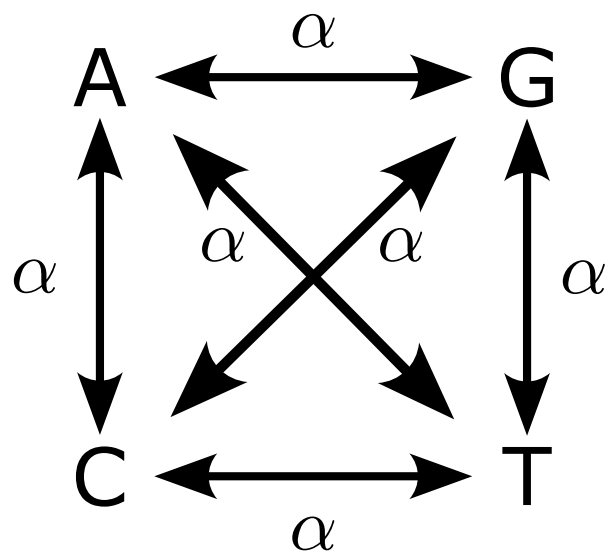
$$P_{i,i}(t) = \frac{1}{4}\left(1 + 3e^{-4\alpha t}\right),$$
$$P_{i,j}(t) = \frac{1}{4}\left(1 - e^{-4\alpha t}\right) \qquad \text{for } i \neq j.$$

We can invert the Jukes-Cantor model to calculate an improved distance measure between sequences.

This can then be used in a distance-based method such as neighbour-joining, etc.

Other models of evolution

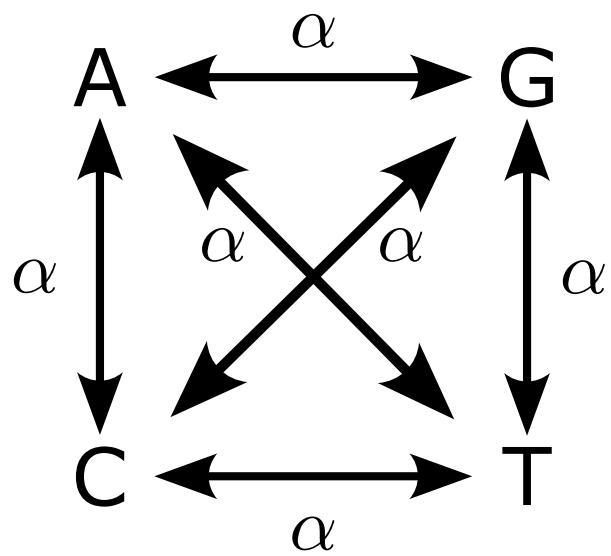
The Jukes-Cantor is only the simplest model of nucleotide evolution, and not very realistic.



Other models of evolution

The Jukes-Cantor is only the simplest model of nucleotide evolution, and not very realistic.

Other models include:



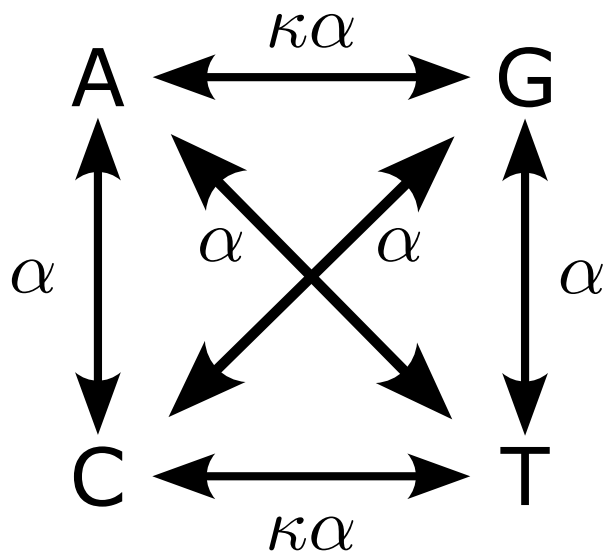
Other models of evolution

The Jukes-Cantor is only the simplest model of nucleotide evolution, and not very realistic.

Other models include:

K80 (Kimura, 1980)

Equal base frequency, but transitions ($A \leftrightarrow G$, $C \leftrightarrow T$) and transversions (everything else) happen at different rates.



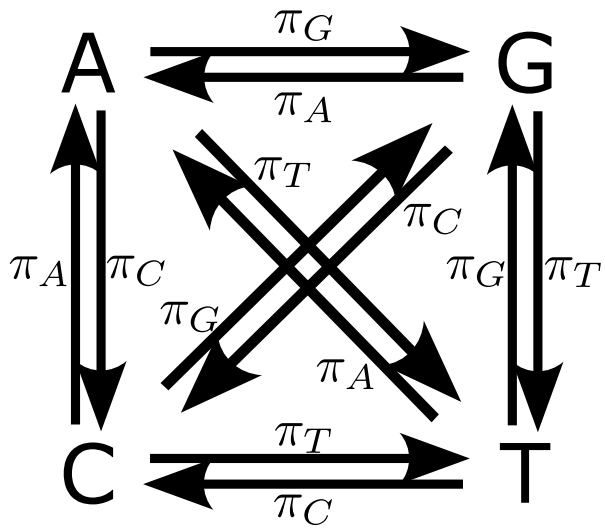
Other models of evolution

The Jukes-Cantor is only the simplest model of nucleotide evolution, and not very realistic.

Other models include:

F81 (Felsenstein, 1981)

Unequal base frequency, but transitions to the same state happen at the same rate.



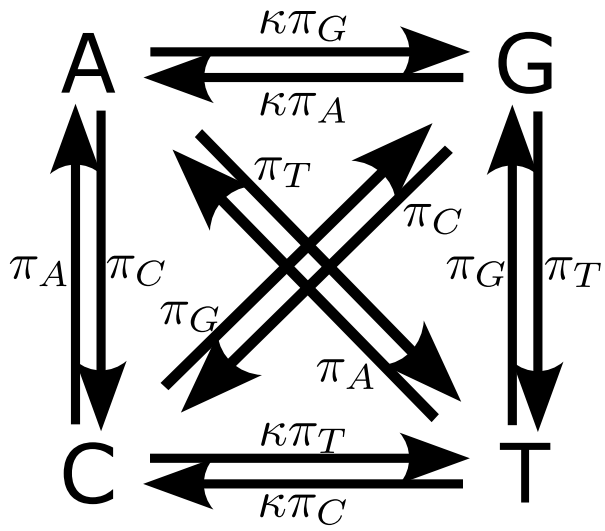
Other models of evolution

The Jukes-Cantor is only the simplest model of nucleotide evolution, and not very realistic.

Other models include:

HKY85 (Hasegawa, Kishino and Yano, 1985)

Unequal base frequency, and transitions and transversions happen at different rates.



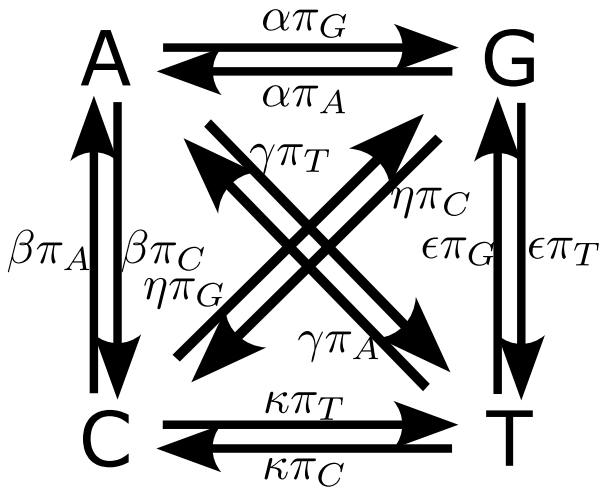
Other models of evolution

The Jukes-Cantor is only the simplest model of nucleotide evolution, and not very realistic.

Other models include:

GTR (Tavaré, 1986)

Unequal base frequency and all transition rates, but the model must be time-reversible: transitions from one state to another must happen at the same rate as the reverse transition.



More models of evolution

These models all assume the same evolution process for each site.

More models of evolution

These models all assume the same evolution process for each site.

In addition, we can vary the overall rate of evolution among sites:

- $+\mathbf{\Gamma}$ varies the overall rate according to a discrete gamma distribution.

More models of evolution

These models all assume the same evolution process for each site.

In addition, we can vary the overall rate of evolution among sites:

- $+\mathbf{\Gamma}$ varies the overall rate according to a discrete gamma distribution.
- $+\mathbf{I}$ allows some sites to be invariant (zero rate).

More models of evolution

These models all assume the same evolution process for each site.

In addition, we can vary the overall rate of evolution among sites:

- $+\mathbf{\Gamma}$ varies the overall rate according to a discrete gamma distribution.
- $+\mathbf{I}$ allows some sites to be invariant (zero rate).
- $+\mathbf{R}$ allows rates to vary freely.

More models of evolution

These models all assume the same evolution process for each site.

In addition, we can vary the overall rate of evolution among sites:

- $+\mathbf{\Gamma}$ varies the overall rate according to a discrete gamma distribution.
- $+\mathbf{I}$ allows some sites to be invariant (zero rate).
- $+\mathbf{R}$ allows rates to vary freely.

These can also be combined (e.g., GTR$+\Gamma+$I).

Model selection

As models get more complicated, they use a larger number of parameters that must be fitted to the data.

Model selection

As models get more complicated, they use a larger number of parameters that must be fitted to the data.

A more complicated model will always "fit better" than a simpler model, but we do not always want the most complicated model.

Model selection

As models get more complicated, they use a larger number of parameters that must be fitted to the data.

A more complicated model will always "fit better" than a simpler model, but we do not always want the most complicated model.

We must select the most appropriate model according to the fit to the data and the model complexity.

Parsimony

A simple scoring criterion is that of parsimony.

Parsimony

A simple scoring criterion is that of parsimony.

Parsimony suggests that evolutionary "events" (here, substitutions) are rare, and so we seek to minimise their occurrence.

Parsimony

A simple scoring criterion is that of parsimony.

Parsimony suggests that evolutionary "events" (here, substitutions) are rare, and so we seek to minimise their occurrence.

The parsimony score of a tree is the minimum number of substitutions required for the tree to generate the given data.

Parsimony

A simple scoring criterion is that of parsimony.

Parsimony suggests that evolutionary "events" (here, substitutions) are rare, and so we seek to minimise their occurrence.

The parsimony score of a tree is the minimum number of substitutions required for the tree to generate the given data.

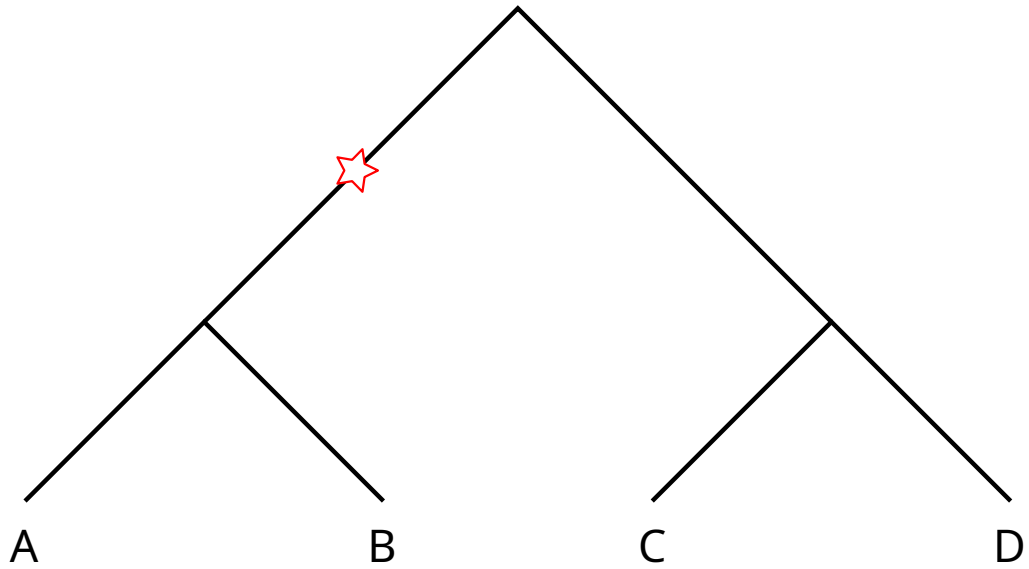
It is easy and fast to calculate the parsimony score for any one tree, but (NP-)hard to calculate the most parsimonious tree. In practice it is not as accurate as likelihood-based methods.

Parsimony

```
A     ...CTGCGATTACACCGGAGTCGACCTAG...
B     ...CTGCTATTACCCGGGAGTAGACCTAG...
C     ...ATTCAATGACATTGGGATTACCCTAG...
D     ...ATTCCATGACATTCGCTAGCGTCTAG...
```

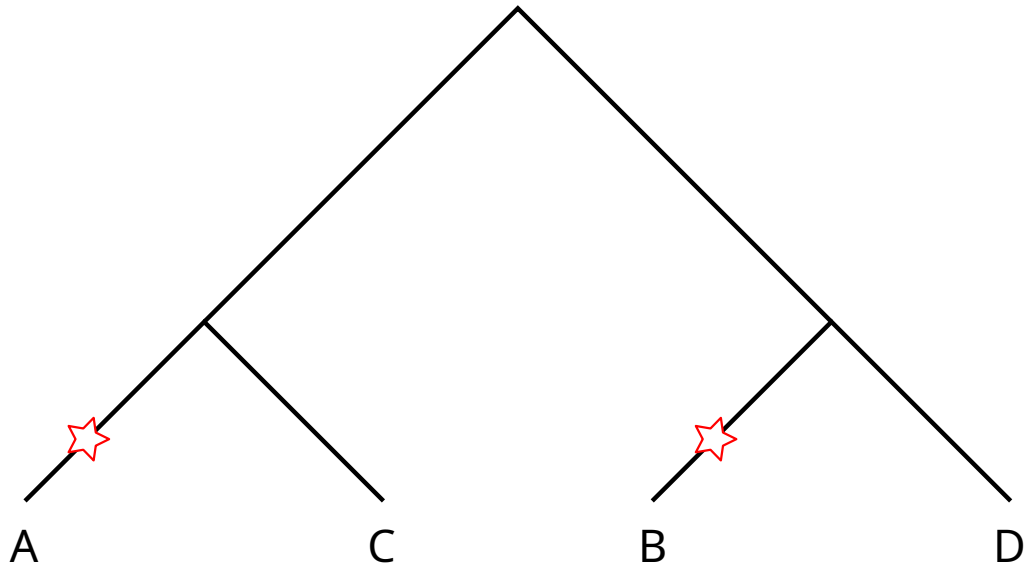
Parsimony

```
A     ...CTGCGATTACACCGGAGTCGACCTAG...
B     ...CTGCTATTACCCGGGAGTAGACCTAG...
C     ...ATTCAATGACATTGGGATTACCCTAG...
D     ...ATTCCATGACATTCGCTAGCGTCTAG...
```



Parsimony

```
A     ...CTGCGATTACACCGGAGTCGACCTAG...
B     ...CTGCTATTACCCGGGAGTAGACCTAG...
C     ...ATTCAATGACATTGGGATTACCCTAG...
D     ...ATTCCATGACATTCGCTAGCGTCTAG...
```



Calculating the likelihood

For a statistical model of evolution, we want to calculate the likelihood of a tree, given an alignment.

Calculating the likelihood

For a statistical model of evolution, we want to calculate the likelihood of a tree, given an alignment.

To do this, we use Felsenstein's pruning algorithm (Felsenstein, 1981).

Calculating the likelihood

For a statistical model of evolution, we want to calculate the likelihood of a tree, given an alignment.

To do this, we use Felsenstein's pruning algorithm (Felsenstein, 1981).

This is a dynamic programming algorithm which computes "partial" likelihoods at each branch.

Calculating the likelihood

For a statistical model of evolution, we want to calculate the likelihood of a tree, given an alignment.

To do this, we use Felsenstein's pruning algorithm (Felsenstein, 1981).

This is a dynamic programming algorithm which computes "partial" likelihoods at each branch.

We first assume that characters evolve independently, so we can compute the likelihood of each character and multiply them.

The pruning algorithm

We know the states of each character at the leaves of the tree, but we don't know the states at each internal node.

The pruning algorithm

We know the states of each character at the leaves of the tree, but we don't know the states at each internal node.

Given states at each internal node, we can easily calculate the likelihood using the Jukes-Cantor (or other model) probabilities.

The pruning algorithm

We know the states of each character at the leaves of the tree, but we don't know the states at each internal node.

Given states at each internal node, we can easily calculate the likelihood using the Jukes-Cantor (or other model) probabilities.

We want to sum over all possible internal states, but the number is too large to do this directly.

The pruning algorithm

We know the states of each character at the leaves of the tree, but we don't know the states at each internal node.

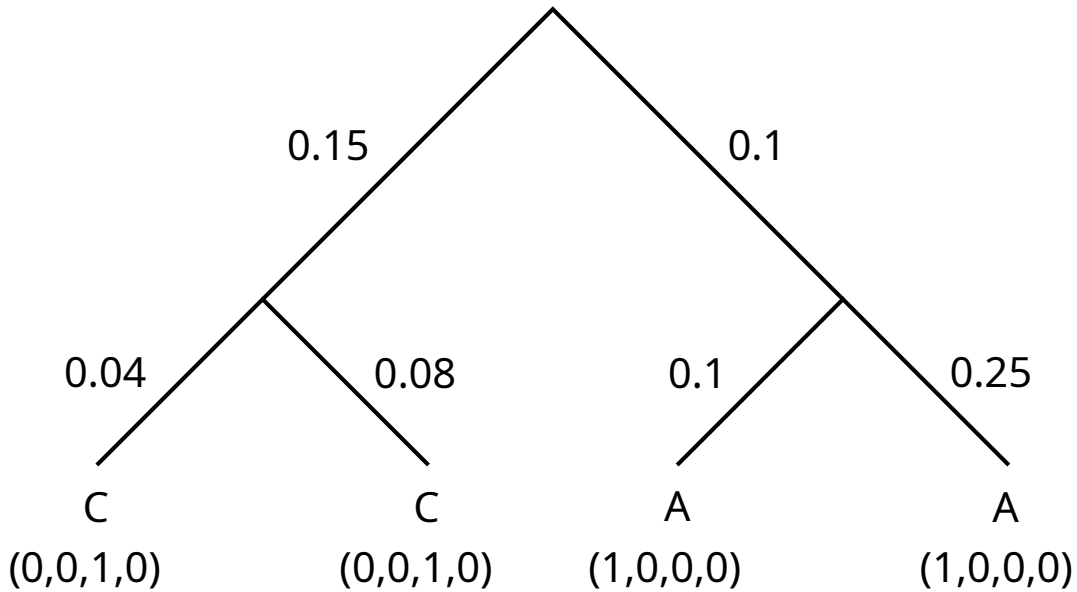
Given states at each internal node, we can easily calculate the likelihood using the Jukes-Cantor (or other model) probabilities.

We want to sum over all possible internal states, but the number is too large to do this directly.

Instead, we use dynamic programming, starting from the leaves and working back to the root.

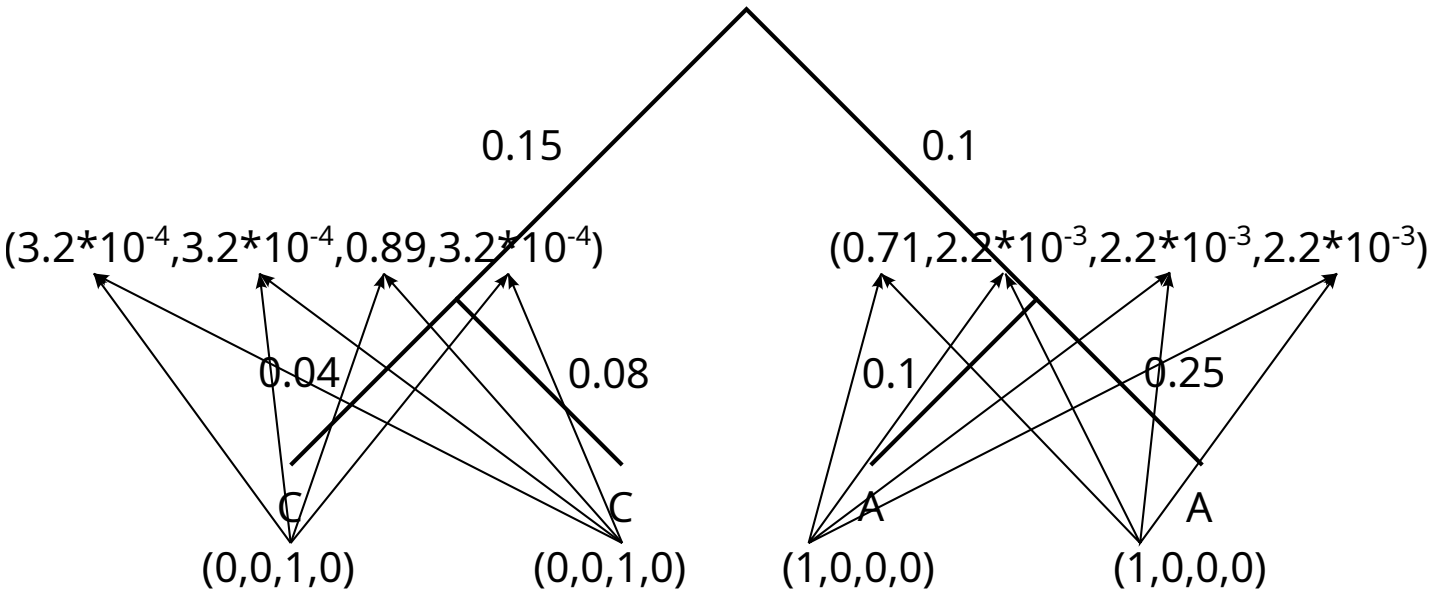
The pruning algorithm

```
A     ...CTGCGATTACACCGGAGTCGACCTAG...
B     ...CTGCTATTACCCGGGAGTAGACCTAG...
C     ...ATTCAATGACATTGGGATTACCCTAG...
D     ...ATTCCATGACATTCGCTAGCGTCTAG...
```



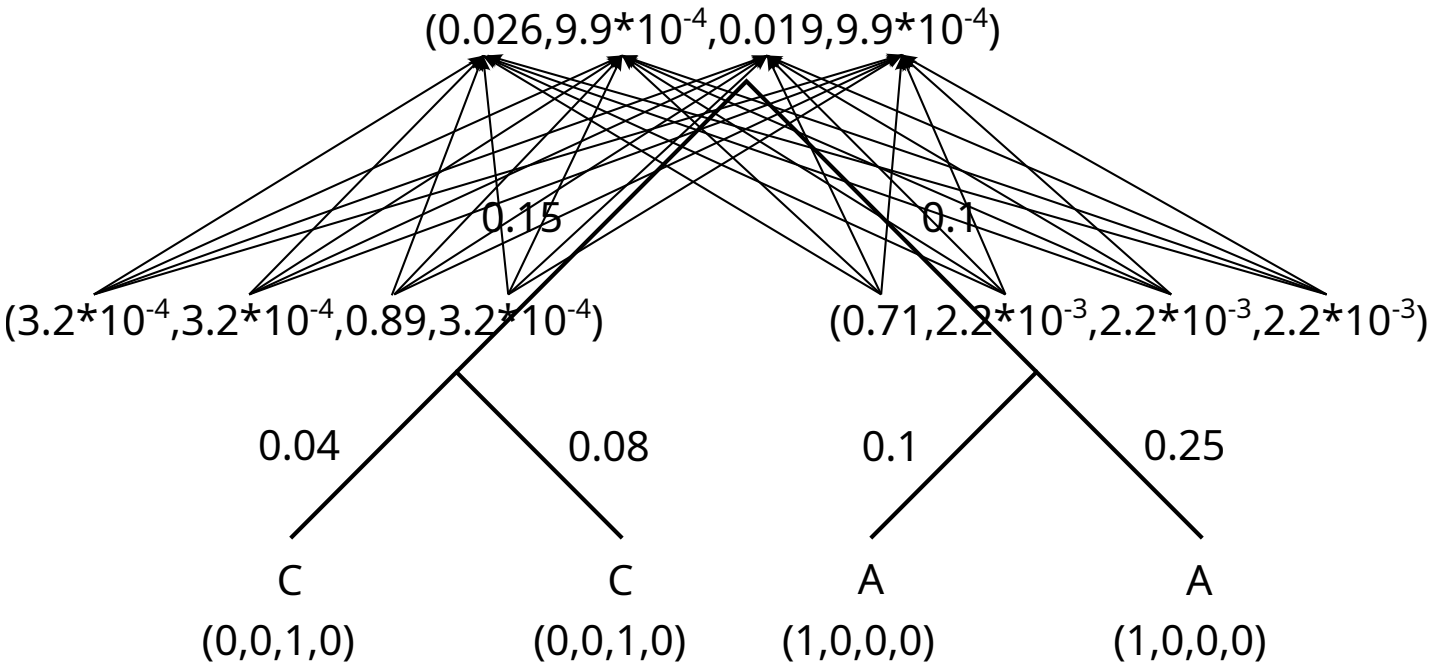
The pruning algorithm

```
A     ...CTGCGATTACACCGGAGTCGACCTAG...
B     ...CTGCTATTACCCGGGAGTAGACCTAG...
C     ...ATTCAATGACATTGGGATTACCCTAG...
D     ...ATTCCATGACATTCGCTAGCGTCTAG...
```



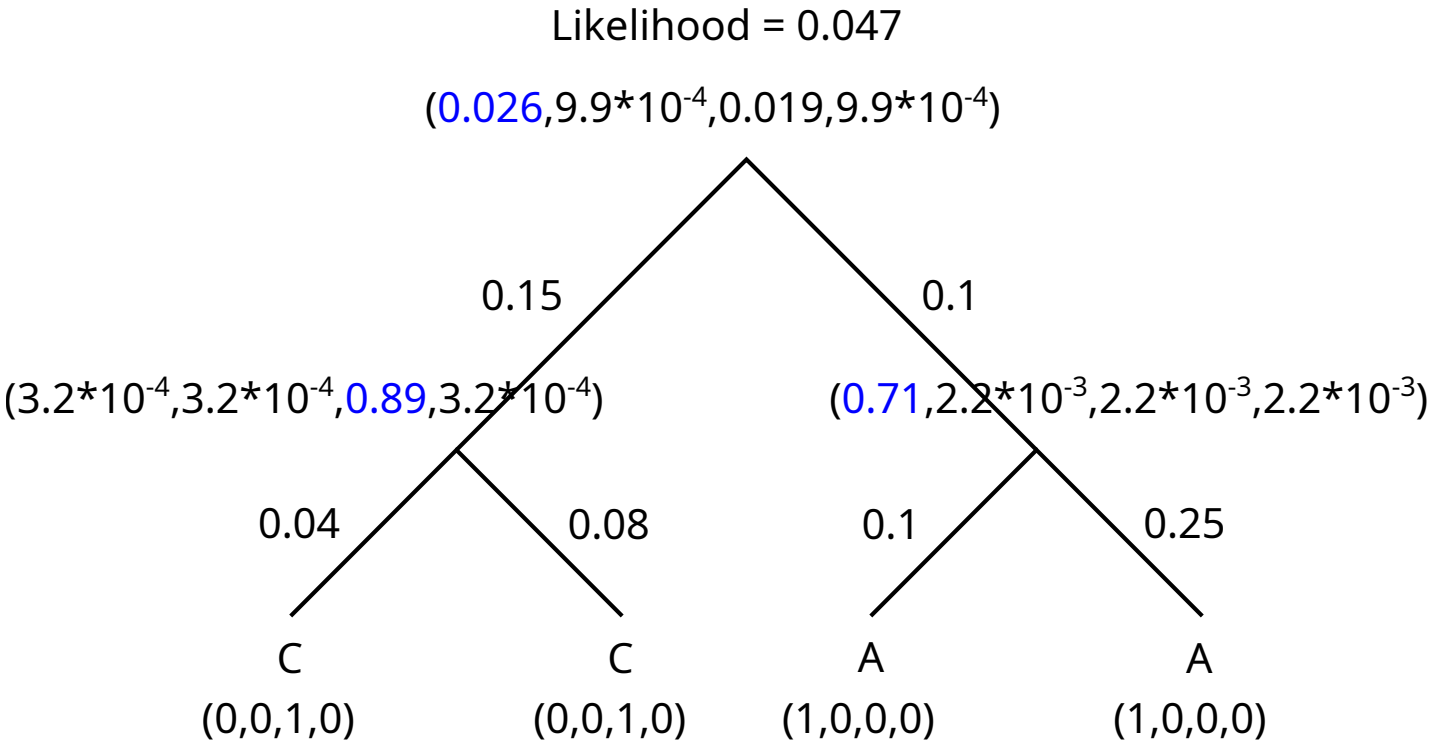
The pruning algorithm

```
A     ...CTGCGATTACACCGGAGTCGACCTAG...
B     ...CTGCTATTACCCGGGAGTAGACCTAG...
C     ...ATTCAATGACATTGGGATTACCCTAG...
D     ...ATTCCATGACATTCGCTAGCGTCTAG...
```



The pruning algorithm

```
A     ...CTGCGATTACACCGGAGTCGACCTAG...
B     ...CTGCTATTACCCGGGAGTAGACCTAG...
C     ...ATTCAATGACATTGGGATTACCCTAG...
D     ...ATTCCATGACATTCGCTAGCGTCTAG...
```



Tree searching

The pruning algorithm makes it easy to calculate the likelihood of any given tree (+ branch lengths).

Tree searching

The pruning algorithm makes it easy to calculate the likelihood of any given tree (+ branch lengths).

But how do we find the most likely tree?

Tree searching

The pruning algorithm makes it easy to calculate the likelihood of any given tree (+ branch lengths).

But how do we find the most likely tree?

We can break this down into two parts: finding the tree topology, and finding the branch lengths.

Tree searching

The pruning algorithm makes it easy to calculate the likelihood of any given tree (+ branch lengths).

But how do we find the most likely tree?

We can break this down into two parts: finding the tree topology, and finding the branch lengths.

Branch lengths can be found (for a given topology) by optimising the likelihood numerically.

Tree searching

But how do we find the tree topology?

Tree searching

But how do we find the tree topology?

The search space is massive, so we can't try all possible trees.

Tree searching

But how do we find the tree topology?

The search space is massive, so we can't try all possible trees.

Algorithms rely on heuristic hill-climbing: start with a reasonable tree (maybe the NJ tree), and make small changes.

Tree searching

But how do we find the tree topology?

The search space is massive, so we can't try all possible trees.

Algorithms rely on heuristic hill-climbing: start with a reasonable tree (maybe the NJ tree), and make small changes.

If one of the small changes results in a tree with a higher likelihood, keep it and repeat.

Tree searching

But how do we find the tree topology?

The search space is massive, so we can't try all possible trees.

Algorithms rely on heuristic hill-climbing: start with a reasonable tree (maybe the NJ tree), and make small changes.

If one of the small changes results in a tree with a higher likelihood, keep it and repeat.

We may also choose to accept (with some probability) a tree with a lower likelihood, to escape local maxima.

Tree moves

There are a variety of ways in which a tree can be changed.

Tree moves

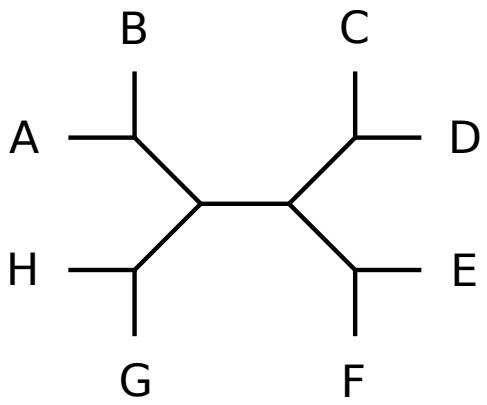
There are a variety of ways in which a tree can be changed.

Nearest neighbour interchange swaps two branches which are next to each other.

Tree moves

There are a variety of ways in which a tree can be changed.

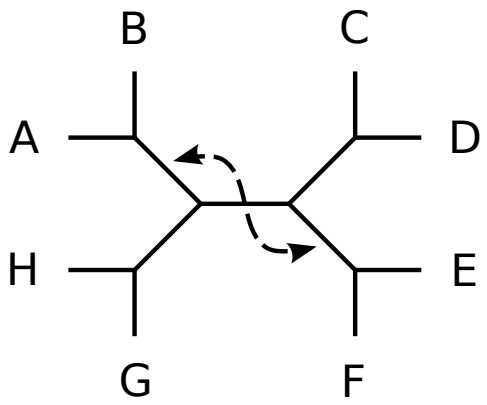
Nearest neighbour interchange swaps two branches which are next to each other.



Tree moves

There are a variety of ways in which a tree can be changed.

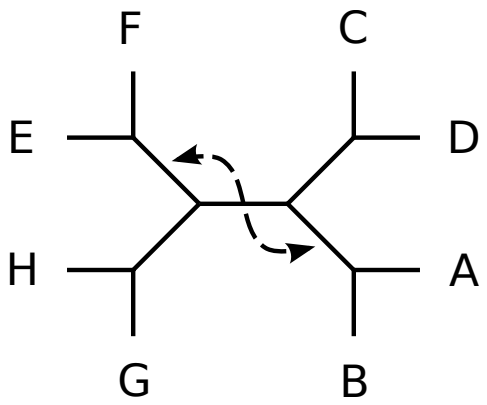
Nearest neighbour interchange swaps two branches which are next to each other.



Tree moves

There are a variety of ways in which a tree can be changed.

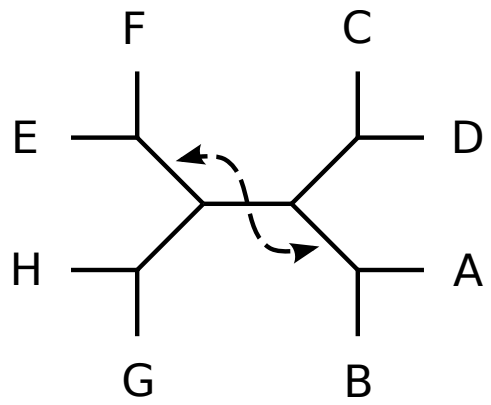
Nearest neighbour interchange swaps two branches which are next to each other.



Tree moves

There are a variety of ways in which a tree can be changed.

Nearest neighbour interchange swaps two branches which are next to each other.



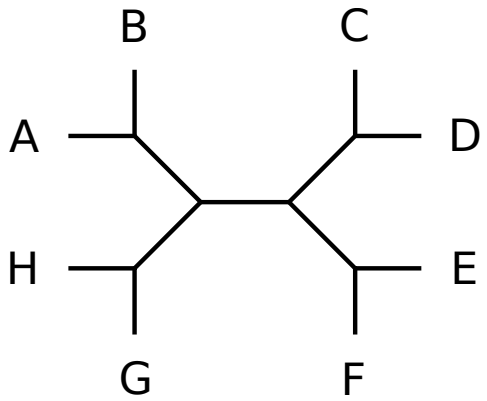
This is quick, but a "small" move, so it is difficult to escape local maxima.

Tree moves

Subtree pruning and regrafting removes one subtree and re-attaches it to another part of the tree.

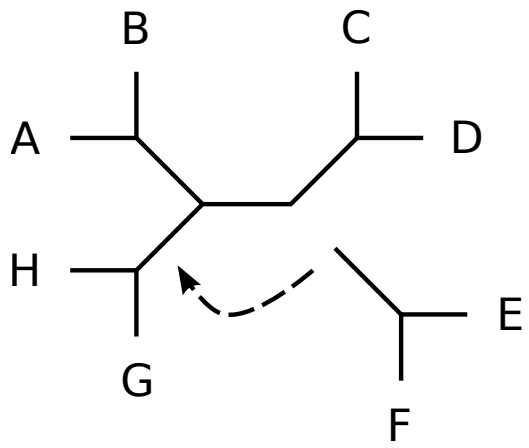
Tree moves

Subtree pruning and regrafting removes one subtree and re-attaches it to another part of the tree.



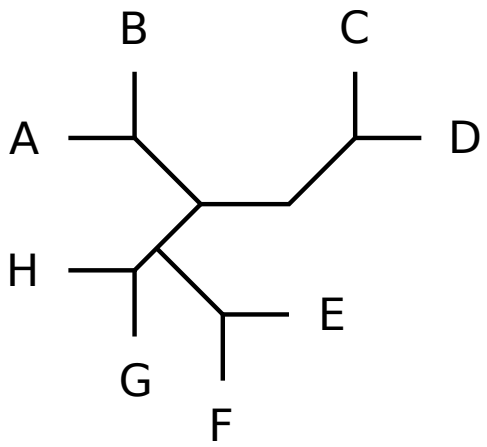
Tree moves

Subtree pruning and regrafting removes one subtree and re-attaches it to another part of the tree.



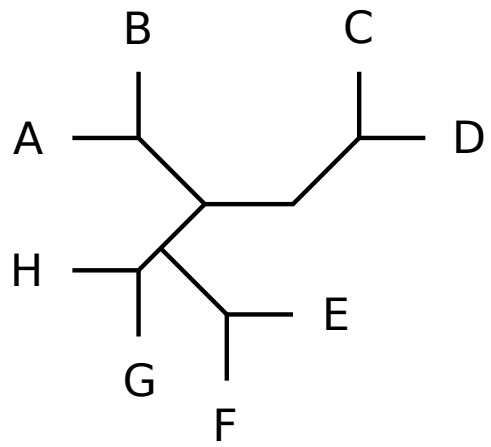
Tree moves

Subtree pruning and regrafting removes one subtree and re-attaches it to another part of the tree.



Tree moves

Subtree pruning and regrafting removes one subtree and re-attaches it to another part of the tree.



This is a "bigger" move, but more computationally intensive.

Maximum likelihood methods

Modern phylogenetic methods use the basic ideas we have shown here, with some variations in move types and probabilities. There are many different methods and each have their own proponents!

Maximum likelihood methods

Modern phylogenetic methods use the basic ideas we have shown here, with some variations in move types and probabilities. There are many different methods and each have their own proponents!

- Phylip (Felsenstein, 1993) starts with a small tree and adds leaves using maximum likelihood in a fixed order;

Maximum likelihood methods

Modern phylogenetic methods use the basic ideas we have shown here, with some variations in move types and probabilities. There are many different methods and each have their own proponents!

- Phylip (Felsenstein, 1993) starts with a small tree and adds leaves using maximum likelihood in a fixed order;
- PhyML (Guindon and Gascuel, 2003) starts with a NJ tree, then modifies it using NNI operations (applied simultaneously);

Maximum likelihood methods

Modern phylogenetic methods use the basic ideas we have shown here, with some variations in move types and probabilities. There are many different methods and each have their own proponents!

- Phylip (Felsenstein, 1993) starts with a small tree and adds leaves using maximum likelihood in a fixed order;
- PhyML (Guindon and Gascuel, 2003) starts with a NJ tree, then modifies it using NNI operations (applied simultaneously);
- RAxML (Stamatakis, 2006) starts with a parsimony tree, then modifies it using (limited) SPR moves.

Maximum likelihood methods

Modern phylogenetic methods use the basic ideas we have shown here, with some variations in move types and probabilities. There are many different methods and each have their own proponents!

- Phylip (Felsenstein, 1993) starts with a small tree and adds leaves using maximum likelihood in a fixed order;
- PhyML (Guindon and Gascuel, 2003) starts with a NJ tree, then modifies it using NNI operations (applied simultaneously);
- RAxML (Stamatakis, 2006) starts with a parsimony tree, then modifies it using (limited) SPR moves.
- IQ-TREE (Nguyen *et al.*, 2015) uses a combination of hill-climbing and stochastic perturbation.

Maximum likelihood: advantages and disadvantages

Maximum likelihood has a number of advantages:

Maximum likelihood: advantages and disadvantages

Maximum likelihood has a number of advantages:

- It is soundly based on a statistical model of evolution.

Maximum likelihood: advantages and disadvantages

Maximum likelihood has a number of advantages:

- It is soundly based on a statistical model of evolution.
- It is usually the most accurate of the methods.

Maximum likelihood: advantages and disadvantages

Maximum likelihood has a number of advantages:

- It is soundly based on a statistical model of evolution.
- It is usually the most accurate of the methods.
- It can be also be used to infer mutation rates and ancestral sequences.

Maximum likelihood: advantages and disadvantages

However, it also has some disadvantages:

Maximum likelihood: advantages and disadvantages

However, it also has some disadvantages:

- It can be computationally intensive.

Maximum likelihood: advantages and disadvantages

However, it also has some disadvantages:

- It can be computationally intensive.
- Algorithms do not promise a global optimum.

Maximum likelihood: advantages and disadvantages

However, it also has some disadvantages:

- It can be computationally intensive.
- Algorithms do not promise a global optimum.
- If your statistical model is wrong...

Bootstrapping

The methods we have outlined above all produce a single tree (and nothing more).

Bootstrapping

The methods we have outlined above all produce a single tree (and nothing more).

However, we would like some idea of how reliable the tree is. Is it supported by lots of the data, or is it distorted by a few unusual characters?

Bootstrapping

The methods we have outlined above all produce a single tree (and nothing more).

However, we would like some idea of how reliable the tree is. Is it supported by lots of the data, or is it distorted by a few unusual characters?

Moreover, alignments can be error-prone. Could a small number of errors produce a large change in the tree?

Bootstrapping

The methods we have outlined above all produce a single tree (and nothing more).

However, we would like some idea of how reliable the tree is. Is it supported by lots of the data, or is it distorted by a few unusual characters?

Moreover, alignments can be error-prone. Could a small number of errors produce a large change in the tree?

To (partly) answer these questions, we use bootstrapping.

Bootstrapping

Bootstrapping is a well-used statistical technique which involves re-sampling data with replacement.

Bootstrapping

Bootstrapping is a well-used statistical technique which involves re-sampling data with replacement.

Here, we re-sample characters with replacement from the original alignment, until we have a new alignment with the same length as the original.

Bootstrapping

Bootstrapping is a well-used statistical technique which involves re-sampling data with replacement.

Here, we re-sample characters with replacement from the original alignment, until we have a new alignment with the same length as the original.

We then apply our method of choice to construct a new tree. Then we repeat this many times.

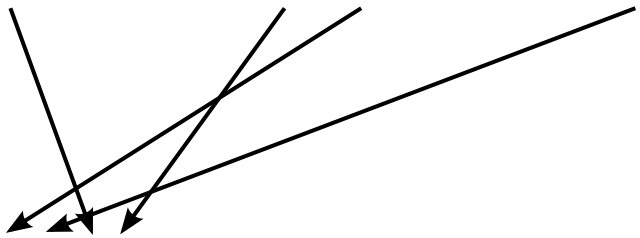
Bootstrapping

...CTGCGATTACACCGGAGTCGACCTAG...

...CTGCTATTACCCGGGAGTAGACCTAG...

...ATTCAATGACATTGGGATTACCCTAG...

...ATTCCATGACATTCGCTAGCGTCTAG...



...CGCTTTGACGAGCAAGATGAAGACAC...

...CGCTTAGCCGGGTAATGTGCCGCTCA...

...CAAGTAGACGGTTAGAGTGCCTCTAT...

...CTAGAACACCGTTACCGACTTTTTAG...

Bootstrapping

To summarise the bootstrapping results, we look at each branch in our final tree and see in what proportion of bootstrap trees this branch (as defined by its descendant taxa) also appears in.

Bootstrapping

To summarise the bootstrapping results, we look at each branch in our final tree and see in what proportion of bootstrap trees this branch (as defined by its descendant taxa) also appears in.

This proportion is known as the bootstrap support value. Values close to 1 indicate a well-supported branch.

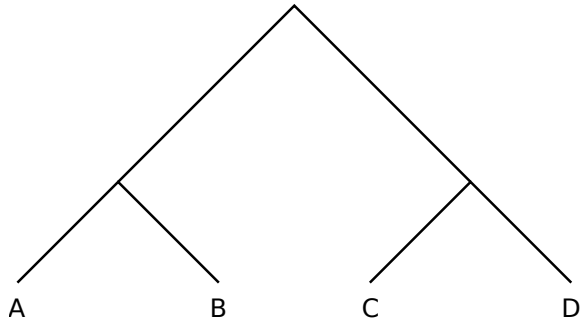
Bootstrapping

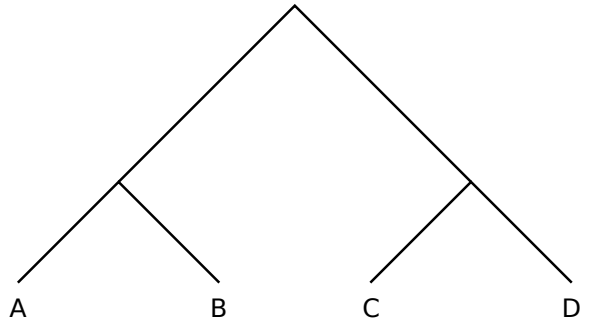
To summarise the bootstrapping results, we look at each branch in our final tree and see in what proportion of bootstrap trees this branch (as defined by its descendant taxa) also appears in.

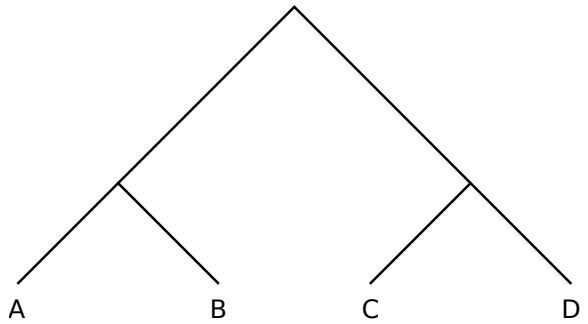
This proportion is known as the bootstrap support value. Values close to 1 indicate a well-supported branch.

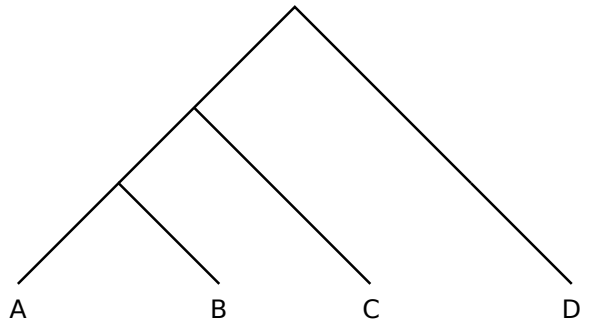
We then attach the bootstrap support value to each branch in our final tree.

Bootstrapping

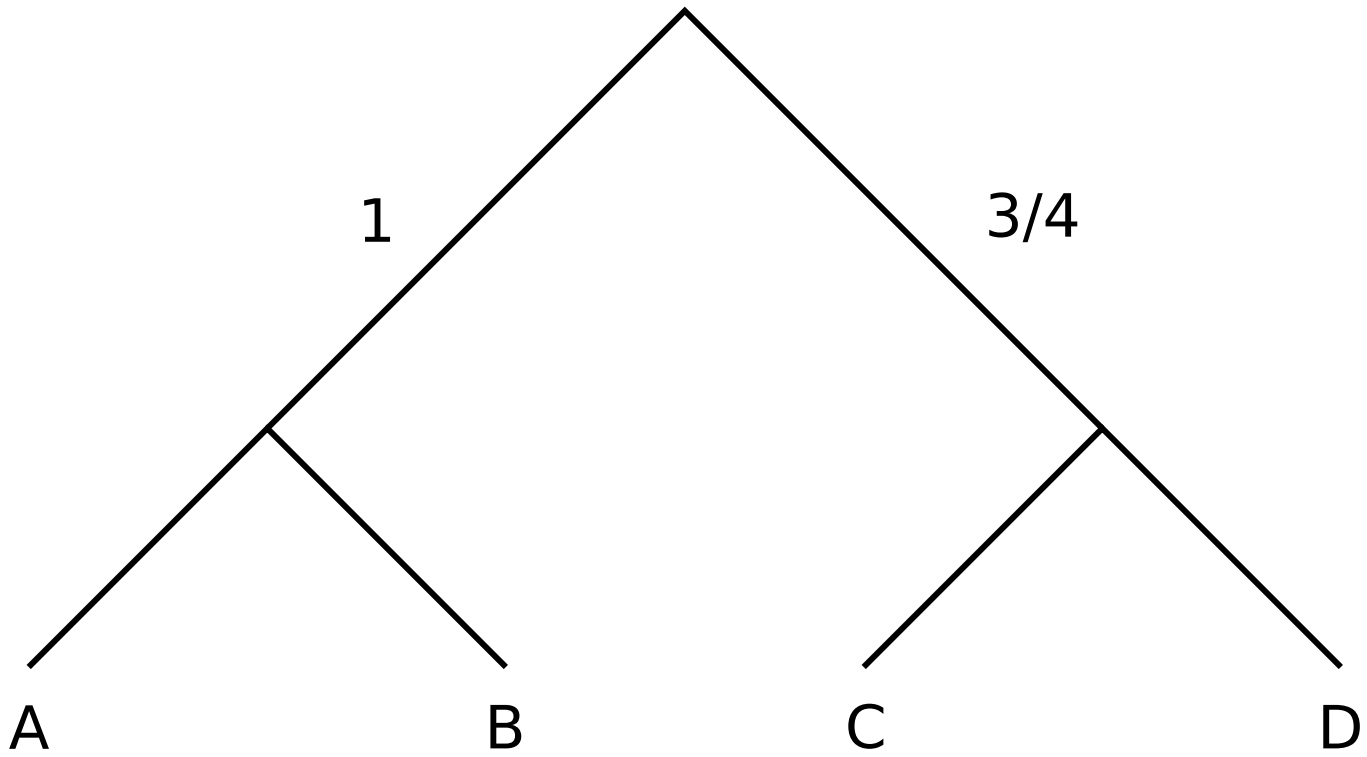








Bootstrapping



Fast bootstrapping

Bootstrapping requires taking many (re)samples and analysing them individually, so it can be slow.

Fast bootstrapping

Bootstrapping requires taking many (re)samples and analysing them individually, so it can be slow.

Alternative approaches have been used to speed up this process.

Fast bootstrapping

Bootstrapping requires taking many (re)samples and analysing them individually, so it can be slow.

Alternative approaches have been used to speed up this process.

RELL (resampling estimated log-likelihoods, Kishino *et al.* 1990) resamples the calculated likelihood of each site, rather than the sites themselves.

Fast bootstrapping

Bootstrapping requires taking many (re)samples and analysing them individually, so it can be slow.

Alternative approaches have been used to speed up this process.

RELL (resampling estimated log-likelihoods, Kishino *et al.* 1990) resamples the calculated likelihood of each site, rather than the sites themselves.

UFBoot (Minh *et al.*, 2013) combines this approach with an efficient way of sampling nearby alternative trees.