Trees from Trees

Creevey, Christopher J; McInerney, James O

Published in:
Methods in Molecular Biology

DOI:
10.1007/978-1-59745-251-9_7

Publication date:
2009

Citation for published version (APA):
Creevey, C. J., & McInerney, J. O. (2009). Trees from Trees: Construction of Phylogenetic Supertrees Using Clann. In *Methods in Molecular Biology* (Vol. 537, pp. 139-61). Springer Nature. https://doi.org/10.1007/978-1-59745-251-9_7

Trees from Trees: Construction of Phylogenetic Supertrees using Clann.

Christopher J. Creevey*[1] and James O. McInerney[2].

[1] EMBL Heidelberg, Meyerhofstrasse 1, 69117 Heidelberg, Germany.

[2] Dept. Biology, National University of Ireland Maynooth, Co. Kildare, Ireland.

* To whom correspondence should be addressed.

Name and address for correspondence:

Christopher Creevey
EMBL Heidelberg, Meyerhofstrasse 1, 69117 Heidelberg, Germany.
Telephone: +49 6221 387 8534
Fax: +49 6221 387 8517
E-mail: chris.creevey@gmail.com

Keywords:

Supertree software, phylogenetic reconstruction, phylogeny, congruency test, phylogenetic signal detection,

Running head:

Trees from trees.

Abstract

Supertree methods combine multiple phylogenetic trees to produce the overall best “supertree”. They can be used to combine phylogenetic information from datasets only partially overlapping and from disparate sources (like molecular and morphological data), or to break down problems thought to be computationally intractable. Some of the longest standing phylogenetic conundrums are now being brought to light using supertree approaches. We describe the most widely used supertree methods implemented in the software program “Clann” and provide a step by step tutorial for investigating phylogenetic information and reconstructing the best supertree. Clann is freely available for Windows, Mac and Unix/Linux operating systems under the GNU public licence at http://bioinf.nuim.ie/software/clann.

1. Introduction

1.1 What are supertrees?

Supertree methods combine the information from a set of taxonomically overlapping phylogenetic trees sometimes called source, or input trees and produce a supertree, or set of equally good supertrees, containing a complete set of all leaves found in the input trees. The analysis requires that the source trees be connected by sets of shared taxa. Source trees that share no taxa in common cannot be combined, however two non-overlapping source trees may be "bridged" by a third that shares taxa with both (**Fig.** 1).

In terms of phylogenetic methods, supertrees are in their infancy, the first papers outlining methods only appeared in the early 1980s (***1, 2***). They are generally characterised by a set of rules detailing how the phylogenetic information from the source trees may be combined. Different methods use different rules but the end result should allow not only the combination of information contained in the source trees, but also the inference of relationships not present in any one source tree. It is also desirable that the resulting supertree does not contain any relationship contradicted in every source tree (***3***).

Supertree methods can be considered a generalisation of consensus methods, except they combine information from partially overlapping trees. However, given a set of completely overlapping source trees (the so called "consensus setting") a supertree method should work exactly like a consensus method in combining the phylogenetic information.

1.2. Why would you want to make supertrees?

1.2.1 Cumulative evidence is philosophically preferable.

There are both philosophical and practical reasons for preferring to use a supertree method rather than other alternatives. From a philosophical viewpoint, the more data you can use to solve a problem, the better the result is likely to be. Supertree methods allow the inclusion of information from disparate sources and this opens the possibility for relationships to be inferred – a situation that would not be possible from one data source alone. For instance, supertree methods have been used to combine genetic data with morphological data (***4, 5***). This combination results in datasets that contain both macro-evolutionary and micro-evolutionary information allowing statements to be made about relationships over evolutionary distances not possible with either of the datasets alone.

From a practical viewpoint, few datasets contain exactly the same species/strains/proteins, and so their combination can be difficult. Furthermore, patchy gene distribution mean that only a few (estimated at 1% (***6***)) genes are universally distributed in single copy. Traditionally phylogenetic studies have relied on this 1% to reconstruct phylogenies of organisms. The ideal situation would be to reconstruct a phylogenetic tree using 100% of the available data. Supertrees provide the only realistic possibility of using 100% of information to reconstruct the tree of life, as a concatenated alignment of all genes from all organisms is likely to have up to too much missing data. The only restriction to using supertree methods is that it must be possible to represent the data as a trees.

1.2.2 To identify trees that are similar and trees that are different.

In real biological datasets, the biggest problem is not data overlap but the conflict that exists between different data types. Conflicting phylogenetic signals can occur because of model misspecification in generating the source trees, hidden paralogy, poor homology determination, lineage sorting or horizontal gene transfer (***7-9***), causing gene trees to differ from the "true" history of speciation events (the species tree). If the factors causing the differences in the gene trees are randomly distributed, then the combination of many gene trees should reveal the species tree. However, when the difference are too great it is not always desirable to simply combine the information, a more investigative approach is desirable in order to identify level of compatibility within the individual source trees. The classic example of this problem is horizontal gene transfer (HGT) in genomic information. A lot has been said recently about the role of HGT in the evolution of microorganisms (***7, 10-16***). Opinions differ over the role HGT has played and whether it has obliterated any possibility of accurately reconstructing the tree of life using the majority of genes (***17, 18***).

It is possible to investigate the role of HGT in a dataset using supertree methods (***7***). Using this approach, trees made from orthologous genes are used to identify the overall phylogenetic signal existing in the majority of genes. Next, the gene trees that differ significantly from this signal can be identified. At this point, depending on the goal of our research, we could examine the genes individually to see why they differ, or see if there is an alternative phylogenetic signal underlying the first. An example of this approach was the recent work on the origin of Eukaryotes, where individual signals were stripped from the data and the secondary and subsequent phylogenetic signals were examined (***19***).

The same approach works for datasets affected by “hidden paralogy”, a situation where deletions of paralogs on different lineages can result in the remaining paralogs being misclassified as orthologs. Long branch attraction (***20***), systematic biases caused by the GC content of synonymous sites (***21***) or evolutionary model misspecification (***22***) also result in spurious source tree topologies and can be identified by investigating the data further.

1.2.3 Divide and conquer? Lots of small trees are easier to construct than one big tree.

Biological datasets are becoming progressively larger and more computationally difficult to handle. A divide-and-conquer approach therefore is sometimes necessary whereby a single large problem is divided into smaller and easier to handle subsets. Each of these subsets can then be solved independently and the results combined to form the overall solution. Supertree methods are well suited to this approach. Lots of small trees are easier to construct than one big tree and their combination into a single tree is exactly the purpose for which supertree methods were designed.

1.3. Are there alternatives?

Other methods exist that can tackle the same type of problems as supertree methods.

1.3.1 Supermatrix.

Concatenating alignments together to produce a supermatrix is a very popular approach to combine information from different sources (***23, 24***). These methods work best when the alignments to be combined have very little or no missing data. The appeal of this approach is that the resulting tree is created directly from the sequences without the necessity for any intermediate step such as the construction of source trees as happens in a supertree analysis. Another alluring feature of the supermatrix approach is based on the assumption that misleading evidence of phylogeny (homoplasy) will be random, whereas true phylogenetic signal, however, weak, will be additive and with enough data this signal should emerge.

Weaknesses of this approach include the inability to explicitly deal with missing data leading to a ‘situation where we don’t know the effect that differing levels of missing data have on the result, although some have attempted to estimate this affect (***25***). Finally, the analysis of a supermatrix requires much more computational power than the divide-and-conquer approach taken by supertree methods. Because of these computational requirements, it is sometimes not possible to build a tree from a concatenated alignment.

1.3.2 Genome content

There are several other methods available for reconstructing a tree when using data from whole genomes. The most commonly used is called a genome content approach (***26***). In this analysis, gene families are identified in the genomes of interest and the presence (and sometimes number of copies) of genes from each family in each genome is recorded. Genomes from closely related species are expected to have similar genome content. This information is combined into a data-matrix that can be used to build a phylogeny. Another variant on this approach is a gene order analysis (***27***). Here, the order rather than the presence or absence of genes is used as a phylogenetic marker. This is a more 'fine-grained' approach than genome content analyses and is better suited to reconstructing relationships between closely related organisms.

The advantage of genome content methods is that they are computationally more tractable than supermatrix or supertree approaches and they have the ability to use all the information from all the genes in a genome. The disadvantages include being very 'coarse-grained' and not using the phylogenetic information from the genes themselves, and being very sensitive to hidden paralogy including HGT. Furthermore, even though there are a few simple, but interesting models of whole genome evolution (***28-30***) we have no realistic models at this scale (***8***) and so we must rely on basic parsimony principles the majority of the time to reconstruct the overall tree.

1.3.3 Conditioned reconstruction

Recently a method of using whole genomes to reconstruct organismal relationships called 'conditioned reconstruction' has been proposed (***31***). Similar to genome content methods, conditioned reconstruction uses information of presence or absence of genes, but between pairs of genomes. This information is further enriched with rates of gene loss and gain within the frequencies of presence or absence. A 'conditioning' genome is used to calculate these frequencies between it and all other genomes in the analysis. In order to calculate the shared absence of genes between genomes several conditioning genomes are used and the combined information is used to reconstruct phylogenetic relationships (***8***). The advantages to this approach are an increased sensitivity to more recent relationships over standard genome content methods while still retaining the ability to use all the information within all the genomes. Disadvantages include our lack of knowledge of how well it performs under violations to the assumptions made by the method. Also it does not use the information that may be gained by directly comparing the sequences of the genes used (***8***).

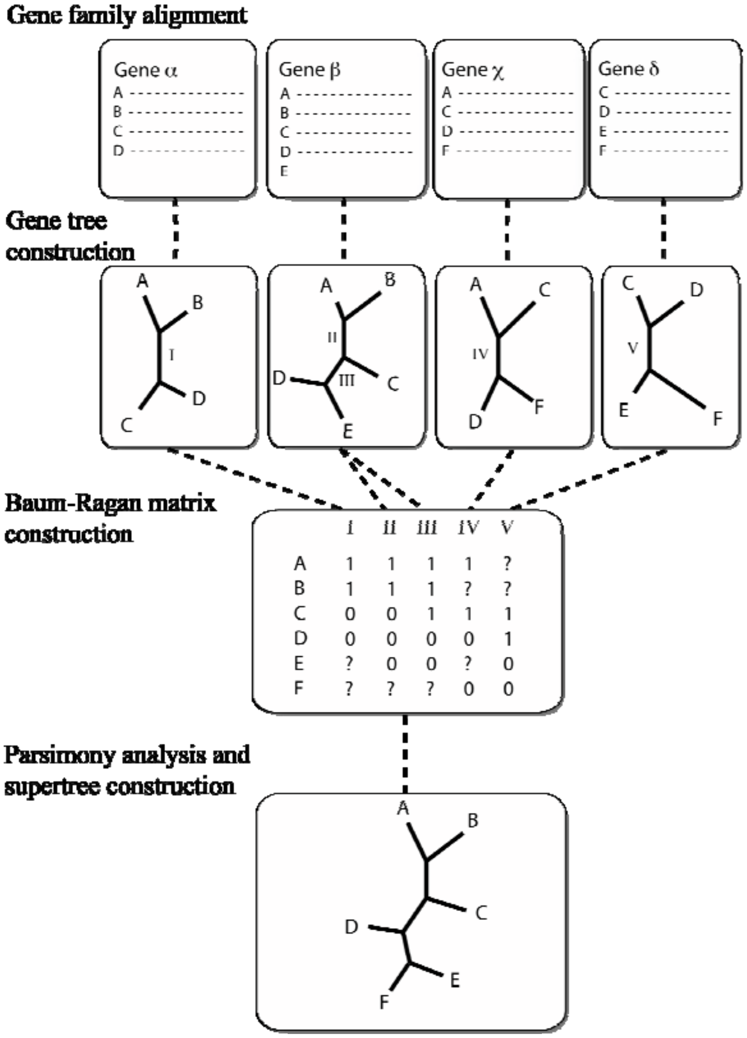
1.4. What supertree methods are out there?

A variety of supertree methods have been developed since the original publications. Each method approaches the problem of combining the information from multiple trees in different ways.

1.4.1 Matrix representation.

The most widely used supertree method is based upon a method proposed independently by both Baum (***32***) and Ragan (***33***). Called matrix representation with parsimony (MRP) it uses a coding scheme to construct a matrix representing the relationships within the source trees. Typically, a maximum parsimony algorithm is then used to reconstruct a supertree from this matrix.

The method identifies the internal branches (also called splits) within each of the source trees and a simple coding scheme of 1s and 0s are used to determine which taxa are on either side of the split (**Fig.** 1). All the taxa on one side of the split are marked with a 1 and the taxa on the other side of the split, with a 0. Any taxa not present on this source tree are marked with a '?' (**Fig.** 1). For unrooted source trees (as is most common with genomic data) it makes no difference which side of a split is marked with a 1 or a 0. The coding for all the internal branches across all the source trees are then combined into a single matrix and this is used to reconstruct the supertree. Despite being widely used, there have been major criticisms about biases in this method, including a tendency to favour the relationships of larger source trees than smaller trees and toward source trees with certain phylogenetic shapes (***3***).



	I	II	III	IV	V
A	1	1	1	1	?
B	1	1	1	?	?
C	0	0	1	1	1
D	0	0	0	0	1
E	?	0	0	?	0
F	?	?	?	0	0

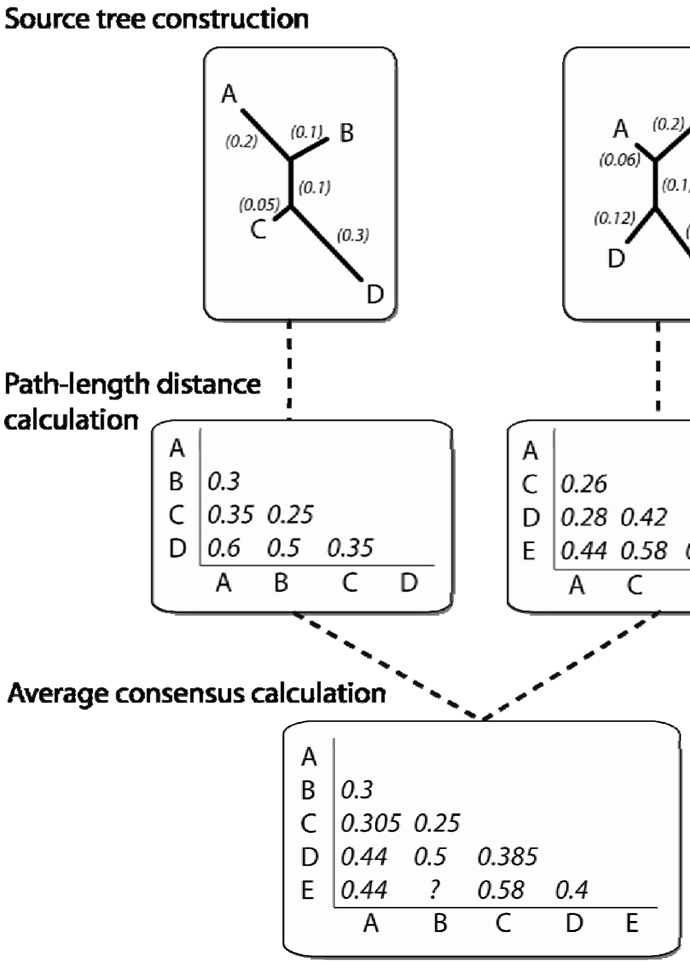
Figure 1

Matrix representation using parsimony (MRP) method.

The MRP procedure is as follows: once the alignment of the gene families is complete, trees are built for each of the genes separately. Within each of these trees, the internal branches (or splits) are identified (I to V above). A Baum-Ragan coding scheme is constructed, containing a column for each of the internal branches. The coding scheme groups the taxa into those that appear on either side of the split. For instance for internal branch I, taxa A and B are on one side and taxa C and D are on the other. In the coding scheme, taxa A and B are both marked with a ‘1’ and taxa C and D are both marked with a ‘0’. As taxa E and F are not in this tree, they are marked with a ‘?’ in column I. When the matrix is completed, a maximum parsimony approach is generally used to reconstruct the supertree.

1.4.2 Average Consensus

A second approach to reconstructing a supertree involves calculating distance matrices representing the relationships within the source trees. These methods may make use of the branch lengths on the source trees and can result in a supertree with branch lengths. One such method is called “Average consensus” (***34***). In this approach the path-length distances of each taxon to every other taxon is calculated across each of the source trees. The average distance of each taxon to every other is then used in a final distance matrix, from which a supertree is constructed (**Fig.** 2). Sometimes however there is an example where two taxa never appear together on any source tree, in this case the average distance of each of these taxa to taxa they both share in common is used to estimate the distance that they would be from each other, if they appeared together on a tree. This is essentially ‘filling in the blanks’ where we have no information concerning their evolutionary relationships to each other. Several methods have been developed to calculate these missing values, see (***35***) for more details. Once the average consensus matrix is complete, a variety of distance-based phylogenetic methods can be used to reconstruct the best supertree. The most commonly used is a least-squares fit (***35***), but it is also possible to use a simple neighbor-joining algorithm (***36***). The advantage of this method is that it produces a supertree with branch lengths.



	A	B	C	D
A				
B	0.3			
C	0.35	0.25		
D	0.6	0.5	0.35	

	A	C	D	E
A				
C	0.26			
D	0.28	0.42		
E	0.44	0.58	0.4	

	A	B	C	D	E
A					
B	0.3				
C	0.305	0.25			
D	0.44	0.5	0.385		
E	0.44	?	0.58	0.4	

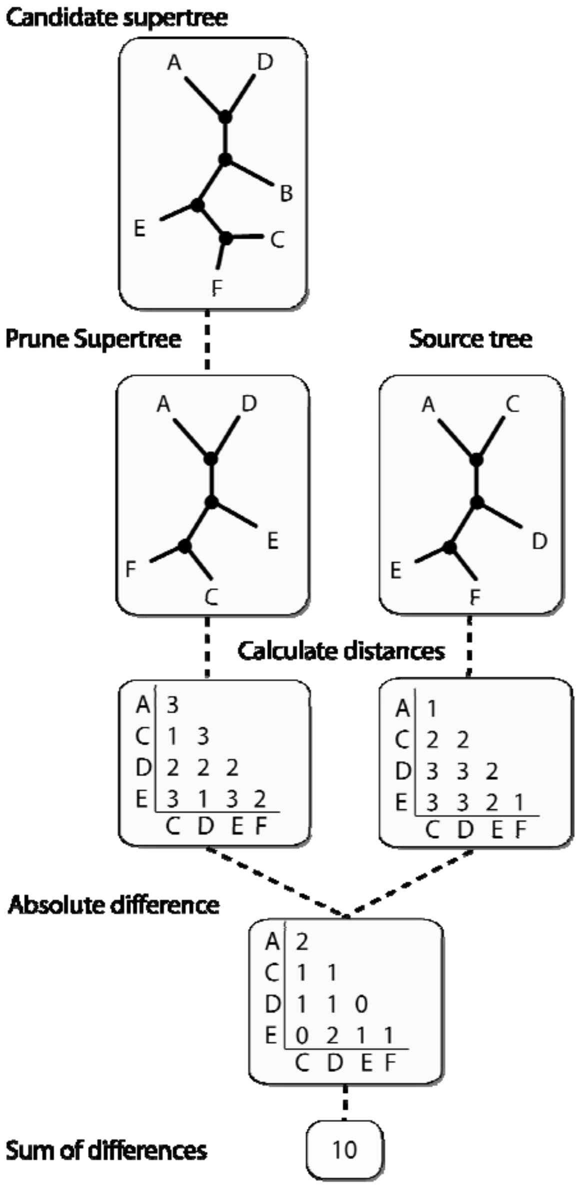
Figure 2

Calculation of average consensus.

In this approach, the branch lengths from the source trees are used to calculate the path length distances of each taxon to every other taxa. In the trees above, the numbers in brackets indicate the lengths of their associated branches. For example: the source tree on the left has a path length distance from taxon A to taxon D of 0.6 (0.2 + 0.1 + 0.3). The average distance each taxon to every other taxon is then calculated for the average consensus. For example: the distance from taxon A to taxon C in the two source trees are 0.35 and 0.26 respectively. The average of these (0.305) is the result put in the average consensus matrix. Some distances are not possible to calculate because the taxa do not appear together in any tree (like with Taxa B and E above), in these cases the value is estimated from the surrounding values in the average consensus matrix.

1.4.3 MSSA-type

Other distance-matrix based methods use different approaches to find the best supertree. The most similar supertree algorithm (MSSA) (*7*) searches for the best supertree without averaging the information from the source trees. Instead, a heuristic search of supertree-space is carried using a scoring function, which when minimised, returns the supertree that is the most similar to the set of source trees. This scoring function works by comparing a candidate supertree to each of the source trees individually. As the supertree will contain all taxa and any source tree is likely to contain only a subset, for each comparison to a source tree the candidate supertree is pruned down to the same taxon set as the source tree. A direct comparison is then possible between the pruned candidate supertree and the source tree. The difference between the two trees is calculated by summing the absolute differences between the path-length distance matrices of the two trees. In this case the path length is defined by the number of internal nodes separating any two taxa on a tree (**Fig.** 3). This pruning-then-comparison method is carried out against every source tree and the sum of the absolute differences is used as a score representing the similarity of the candidate supertree to the set of source trees. A score of zero represents the situation where every source tree is identical to the supertree (when the supertree is pruned to the same size for comparison). Multiple candidate supertrees are tested to find the one that minimises the score function when compared to the source trees. An exhaustive search of supertree-space can be carried out or a standard heuristic search can be used (for instance: nearest neighbor interchange (NNI) or sub-tree pruning and regrafting (SPR)).



	C	D	E	F
A	3			
C	1	3		
D	2	2	2	
E	3	1	3	2

	C	D	E	F
A	1			
C	2	2		
D	3	3	2	
E	3	3	2	1

	C	D	E	F
A	2			
C	1	1		
D	1	1	0	
E	0	2	1	1

Figure 3

Most Similar Supertree Algorithm (MSSA).

In this approach, a function is used to assess candidate supertrees. A heuristic or exhaustive search of supertree space is carried out and the supertree that minimises the function is the most similar supertree to the set of source trees. The difference between the candidate supertree and each source tree is calculated separately and the sum of these scores is the overall score for the supertree. For each comparison to a source tree the supertree is firstly pruned down to the same taxa set as the source tree (above). Next a path-length distance score representing the differences between the two trees is calculated. The path-length distances are the number of internal nodes (filled circles in the trees above) that are in the path between any two taxa on the tree. The sum of the absolute differences between the matrices is the score representing the difference between the supertree and this source tree. This value is usually divided by either the number of comparisons in the matrix or the number of species shared by the supertree and the source tree to counteract biases from large source trees.

1.4.4 Quartets

Quartet methods generally break down the source trees into their constituent quartets and use various approaches to find the supertree that shares the most quartets with the set of source tree quartets. A set of quartets is all possible 4-taxon trees that can be made by pruning the any tree. The optimum supertree can be found using several techniques, including by simply counting the number of shared quartets or by using a “puzzling step” whereby random subsets of quartets are combined in a step-wise manner to “grow” the supertree. This is then repeated many times to see which supertree relationships are reconstructed the most often.

2. Program Usage

Clann (***37***) is a command-line software package for investigating phylogenetic information through supertree analyses and is freely available under a GNU public license agreement. Version 3.1 implements 5 different supertree methods, including matrix representation using parsimony (MRP), average consensus, and the most similar supertree algorithm (MSSA). In this version of Clann the MRP criterion requires the use of an external parsimony program like PAUP* (***38***), future versions will remove this requirement.

2.1 Installation

Clann is available at http://bioinf.nuim.ie/software/clann. On the download page the choice of three different operating systems are available (Mac OSX, Linux and Microsoft Windows). In Mac OSX an installation script is included which installs the readline and ncurses libraries (if needed) before putting Clann into the folder /usr/bin/. Once in this location (and the user starts a new terminal window) Clann will be visible to the operating system from any directory. An administrative password will be needed to successfully install Clann.

On a Linux operating system, the Clann program should either be located in the same directory as the input files, or somewhere on your path (e.g. *~/bin/* or */usr/local/bin*). If you do not know which directories are on your path, ask your system administrator.

On the Microsoft Windows double clicking on the icon associated with Clann will run the program. Using this operating system, the Clann program **must** be located in the same directory as the input files, an alias or shortcut to Clann will not suffice.

To run Clann on the MacOSX or Linux operating systems, type the command "*./clann*" or "*clann*" in a terminal window.

2.2 File formats

Clann accepts source trees in two different formats: newick (also called phylip format) and nexus format (*see* **Note 1**). Multiple trees can be contained in the same file in both formats. Newick formatted trees are the simplest to construct and can contain branch lengths, internal branch labels, tree weights and tree names (*see* **Note 2**) (**Fig.** 4).

The nexus format is a modular system for representing many types of systematic information, including sequences and trees (***39***). Branch lengths and internal branch labels are indicated in the same manner to newick-formatted files. Different types of systematic information are contained within "blocks". It is also possible to include a clann block, containing the commands to be executed on the data in the file. To load a file of trees into clann use the command `exe filename` or include the name of the file to be executed along with the call for clann at the operating system prompt (i.e. "clann filename") (*see* **Note 3**). After completing a summary of the relationships between the trees in the file, clann will return the prompt "clann>". From this point all the different commands available can be executed (*see* **Notes 4 and 5**).

```
i)    (A,B,(C,D));
ii)   (A:0.01,B:0.02,(C:0.01,D:0.03):0.01);
iii)  (A,B,(C,D)Int1);
iv)   (A,B,(C,D))[1.5];
v)    (A,B,(C,D)); [Tree name]
vi)   (A:0.01,B:0.02,(C:0.01,D:0.03)Int1:0.01)[1.5]; [Tree name]
```

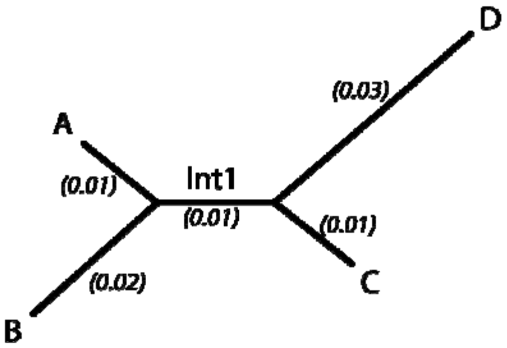


Figure 4

Newick (phylip) format trees.

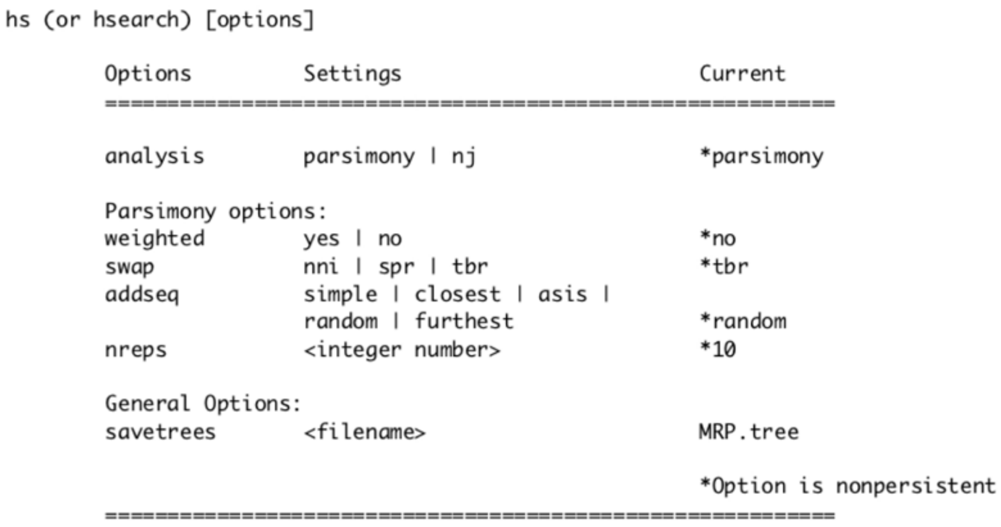
Newick formatted trees can contain a variety of information. i) The simplest form which just contains the tree topology. ii) Branch length information incorporated onto the tree (in brackets on the tree above). iii) Internal branch name (Int1) included, this could also be used to indicate a bootstrap proportion value. iv) A tree weight can be included within square brackets before the semi colon. This may be used if more or less emphasis should be applied to the relationships any tree (1 is the default). v) Trees may be given specific names within square brackets after the semi colon, representing the datasets from which they were constructed. vi) All possible information from i) to v) above included on a single tree. Multiple trees can be contained in a single file.

2.3 Building Supertrees

Each supertree method in clann is a separate *criterion* on which operations such as heuristic searches, exhaustive searches or bootstrap resampling analyses can be carried out (*see* **Note 6**). The 5 criteria implemented in version 3.1 of clann are MRP (matrix representation using parsimony), DFIT (Most similar supertree algorithm), SFIT (split fit algorithm), QFIT (quartet fit algorithm) and AVCON (average consensus method).

2.3.1 Constructing an MRP tree

From the clann prompt the command `set criterion=mrp` tells clann that all following commands are to use MRP as the criterion for assessing supertrees (*see* **Note 7**). The quickest way to reconstruct a supertree in this criterion is using a heuristic search of tree-space. The command `hs ?` lists the possible options for a heuristic search (**Fig.** 5).



```
hs (or hsearch) [options]

        Options         Settings                        Current
        ===========================================================

        analysis        parsimony | nj                  *parsimony

        Parsimony options:
        weighted        yes | no                        *no
        swap            nni | spr | tbr                 *tbr
        addseq          simple | closest | asis |
                        random | furthest               *random
        nreps           <integer number>                *10

        General Options:
        savetrees       <filename>                      MRP.tree

                                                        *Option is nonpersistent
        ===========================================================
```

Figure 5

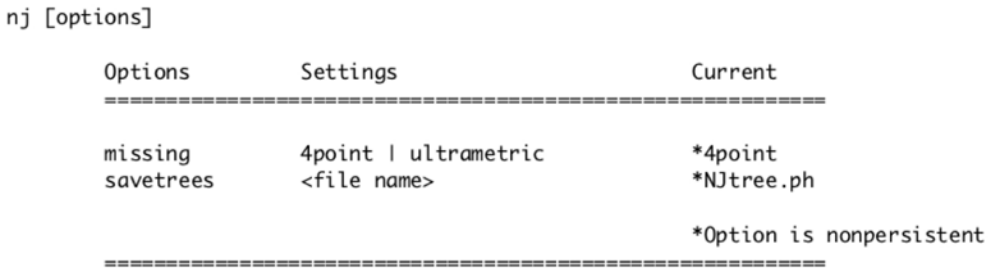
Options available with the heuristic search (hs) command using the matrix representation using parsimony (MRP) criterion.

By default a heuristic search will use PAUP* (***38***) to carry out the parsimony analysis. Clann will try to run PAUP* but if it fails to do so, will return an error and suggest that the user should execute the created Baum-Ragan matrix in PAUP* separately. The use of other parsimony programs is also possible. The best supertree(s) is saved to the file “MRP.tree”, although it is possible to change the name of the file using the option

`savetrees=new-file-name`.

2.3.2 Making an average consensus tree.

The quickest method to construct a supertree is to create a neighbor joining tree using the average consensus method to create the distance matrix. This can be carried out under any criteria using the command `nj`. The resulting tree is both displayed on screen and saved to the file “NJtree.ph”. The options for this approach only concern the methods used to fill in the missing values in the distance matrix. Typing the command `nj ?` returns the options possible with this command (**Fig.** 6).



```
nj [options]

        Options         Settings                        Current
        ===========================================================

        missing         4point | ultrametric            *4point
        savetrees       <file name>                     *NJtree.ph

                                                        *Option is nonpersistent
        ===========================================================
```

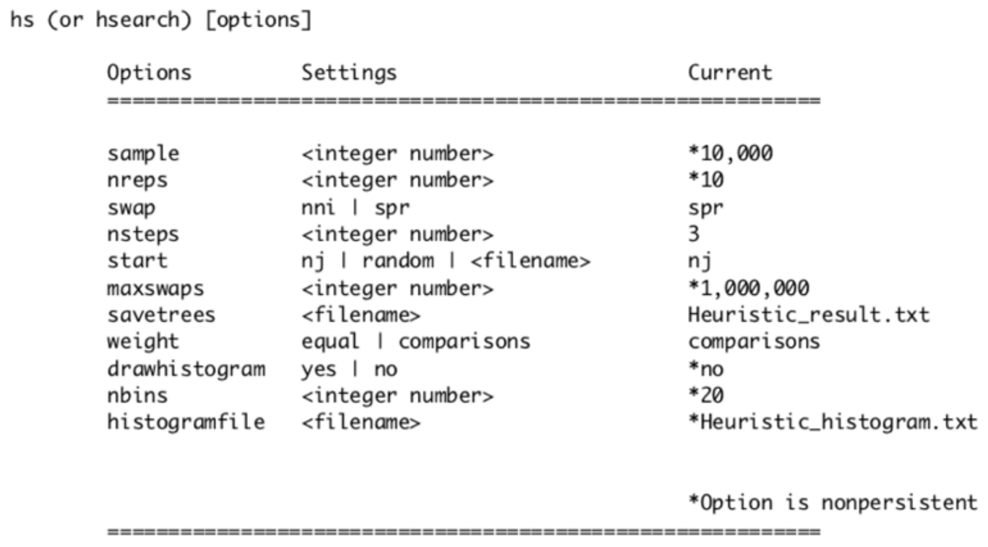
Figure 6

Options available with the neighbor joining supertree (nj) option.

To carry out a full average consensus analysis it is necessary to change the criterion to “avcon”. By default, Clann uses PAUP*(***38***) to carry out the heuristic search using the least-squared objective function. The resulting tree is displayed on screen and saved to the file “Heuristic_result.txt”.

2.3.3 Making an MSSA tree

The MSSA algorithm is called under the criterion “DFIT” in clann. The command `hs ?` displays the options available under the MSSA criterion (*see* **Note 8**) (**Fig.** 7).



```
hs (or hsearch) [options]

        Options         Settings                        Current
        ===========================================================

        sample          <integer number>                *10,000
        nreps           <integer number>                *10
        swap            nni | spr                       spr
        nsteps          <integer number>                3
        start           nj | random | <filename>        nj
        maxswaps        <integer number>                *1,000,000
        savetrees       <filename>                      Heuristic_result.txt
        weight          equal | comparisons             comparisons
        drawhistogram   yes | no                        *no
        nbins           <integer number>                *20
        histogramfile   <filename>                      *Heuristic_histogram.txt


                                                        *Option is nonpersistent
        ===========================================================
```

Figure 7

Options available with the heuristic search (hs) command using the most similar supertree algorithm (MSSA) criterion.

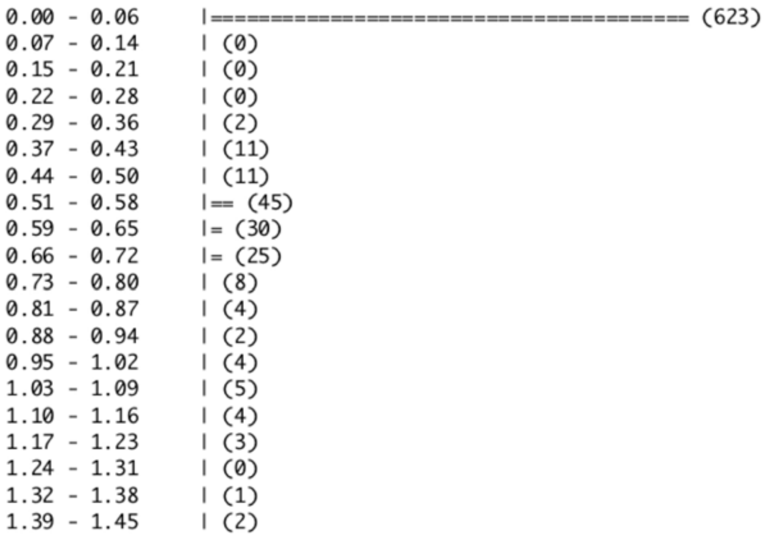
By default clann will create 10 neighbor-joining trees with some random changes and carry out the heuristic search from these starting points. It is also possible to specify a random pre-sample of supertree space to find the best starting points from which to carry out the heuristic search using the option “start”. When the search is complete, the best supertree(s) are displayed to screen.

2.4 Visualising output.

By default, any supertrees reconstructed by clann are saved into their respective files in newick format. There are a variety of tree-viewing application that can read these files including stand-alone applications like Treeview (***40***) and online tools like iTOL (***41***). Clann also saves the trees returned from heuristic searches as a post-script image file (called “trees.ps”) that can be viewed by a variety of applications.

2.5 Interrogating input trees

One of the strengths of Clann is its ability to allow the user to investigate the phylogenetic support in the source trees for a supertree (*see* **Notes 9 and 10**). At the most basic level, the user can choose to rank the source trees according to their similarity to the best supertree. This is carried out during a heuristic search for the best supertree. If the option “drawhistogram” has been set to “yes”, a histogram providing information on how similar the supertree is to the set of source trees is displayed (where a score of 0 means they are identical) (**Fig.** 8). The information is also saved to the file "Heuristic_histogram.txt”.

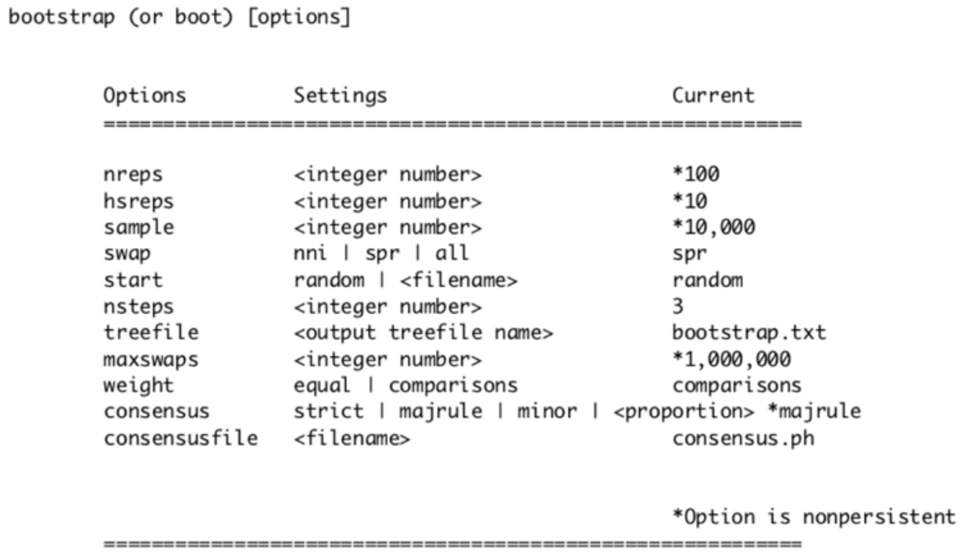


```
0.00 - 0.06     |======================================== (623)
0.07 - 0.14     | (0)
0.15 - 0.21     | (0)
0.22 - 0.28     | (0)
0.29 - 0.36     | (2)
0.37 - 0.43     | (11)
0.44 - 0.50     | (11)
0.51 - 0.58     |== (45)
0.59 - 0.65     |= (30)
0.66 - 0.72     |= (25)
0.73 - 0.80     | (8)
0.81 - 0.87     | (4)
0.88 - 0.94     | (2)
0.95 - 1.02     | (4)
1.03 - 1.09     | (5)
1.10 - 1.16     | (4)
1.17 - 1.23     | (3)
1.24 - 1.31     | (0)
1.32 - 1.38     | (1)
1.39 - 1.45     | (2)
```

Figure 8

Histogram detailing the similarity of the best supertree to the source trees.

A bootstrap analysis of the source trees can also be carried out in clann using the command `bootstrap` or `boot`. This analysis will resample the set of source trees with replacement to create a new set with the same number of trees as the original. This is generally carried out 100 times and the best supertree is found for each. Clann then carries out a summary of the source trees (usually a majority rule consensus) and the relationships with the best support are displayed to screen. All the best supertrees for each bootstrap replicate are saved to the file “bootstrap.txt” and the consensus to “consensus.txt”. The bootstrap analysis can be carried out for each of the 5 different criteria in clann. Typing boot? displays the options available under the current criterion (**Fig.** 9).



```
bootstrap (or boot) [options]


        Options         Settings                        Current
        ===========================================================

        nreps           <integer number>                *100
        hsreps          <integer number>                *10
        sample          <integer number>                *10,000
        swap            nni | spr | all                 spr
        start           random | <filename>             random
        nsteps          <integer number>                3
        treefile        <output treefile name>          bootstrap.txt
        maxswaps        <integer number>                *1,000,000
        weight          equal | comparisons             comparisons
        consensus       strict | majrule | minor | <proportion> *majrule
        consensusfile   <filename>                      consensus.ph


                                                        *Option is nonpersistent
        ===========================================================
```

Figure 9

Options available with the bootstrapping (boot) command using the most similar supertree algorithm (MSSA) criterion.

Finally it is possible to carry out an analysis of the level of congruent phylogenetic signal across the set of source trees using a permutation-tail-probability test. The test implemented in Clann is called the YAPTP (yet another permutation-tail-probability) test. This test compares the score of the best supertree to the score of the best supertrees from 100 randomly permuted versions of the source trees. For each of the 100 replicates of this analysis each source tree is randomised, thereby destroying the any congruent signal between them, while keeping the same taxon distribution and source tree sizes. A heuristic search for the best supertree for each of these randomised datasets is then carried out. If topological congruence between the source trees is better then random, then the supertree score for the real dataset is expected to be better than any of the supertree scores from the randomised datasets. This is essentially testing that the phylogenetic signal shared between the source trees is better then random noise (*7*).

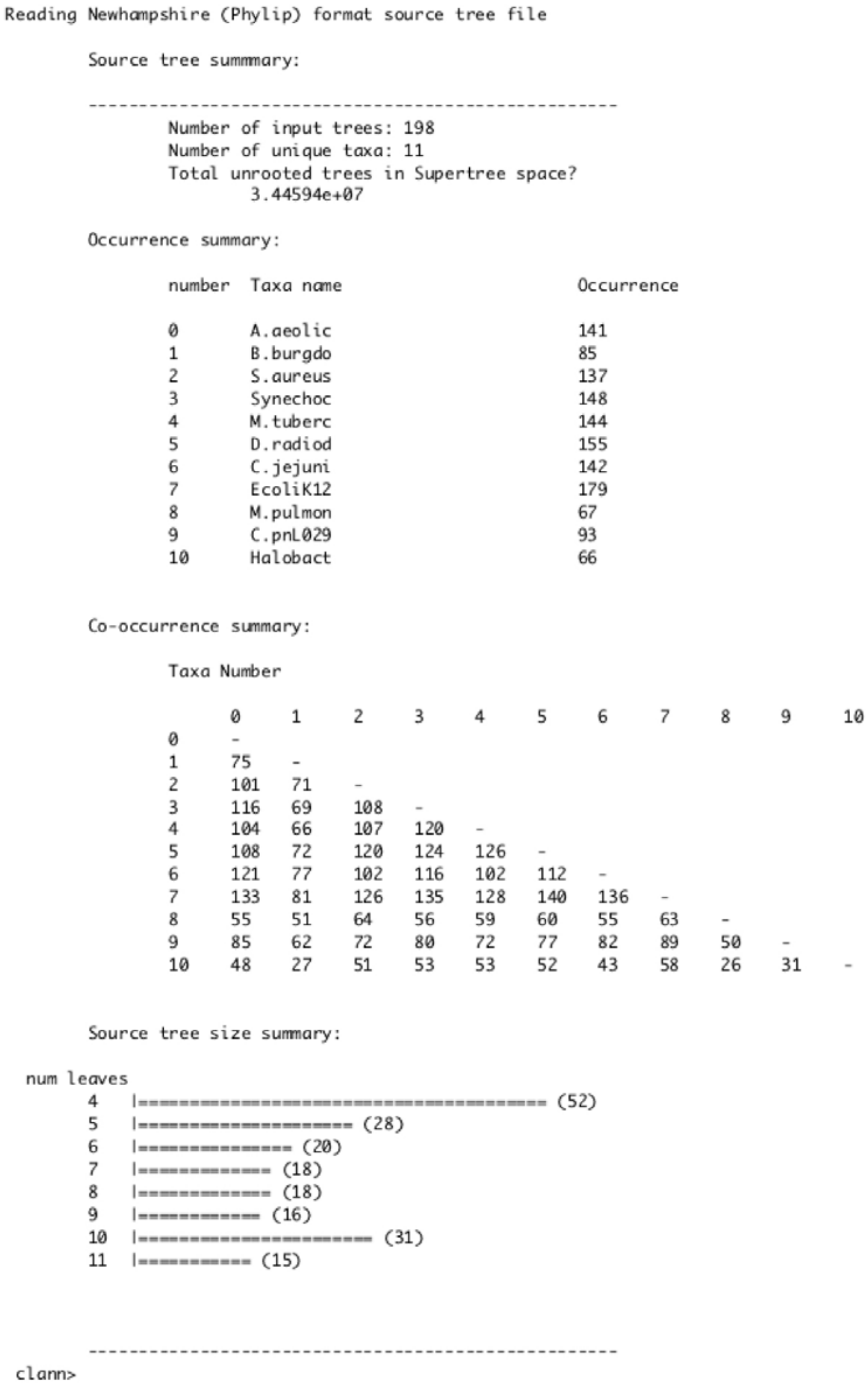
3. Examples

The datasets used in the following examples are from (*7*). This analysis concerns the extent of phylogenetic signal with the prokaryotes. Two datasets were constructed, the first consisting of single-copy gene families from 10 genomes within the gamma-proteobacteria and the second consisting of single-copy gene families from 11 genomes spanning the earliest branches of the prokaryotic tree of life. For more details on the methods used to create the source trees see (*7*). The source trees created are available for download at http://bioinf.nuim.ie/supplementary/royalsoc04/ .

Beginning with the dataset from genomes spanning the earliest branches of the prokaryotes (the file named “11taxonfundamentals.ph.txt”), from within Clann type:

```
exe 11taxonfundamentals.ph.txt
```

Clann will read in the source trees and calculate and display some basic statistics about the data (**Fig.** 10).



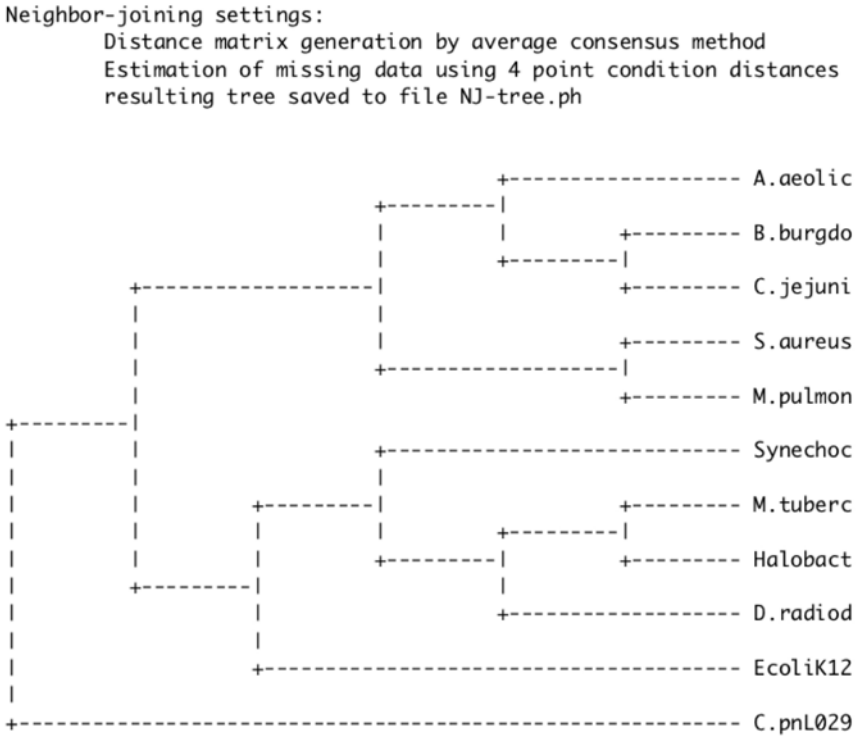
```
Reading Newhampshire (Phylip) format source tree file

        Source tree summary:

        ----------------------------------------------------
                Number of input trees: 198
                Number of unique taxa: 11
                Total unrooted trees in Supertree space?
                        3.44594e+07

        Occurrence summary:

                number  Taxa name                       Occurrence

                0       A.aeolic                        141
                1       B.burgdo                        85
                2       S.aureus                        137
                3       Synechoc                        148
                4       M.tuberc                        144
                5       D.radiod                        155
                6       C.jejuni                        142
                7       EcoliK12                        179
                8       M.pulmon                        67
                9       C.pnL029                        93
                10      Halobact                        66


        Co-occurrence summary:

                Taxa Number

                      0     1     2     3     4     5     6     7     8     9     10
                0     -
                1     75    -
                2     101   71    -
                3     116   69    108   -
                4     104   66    107   120   -
                5     108   72    120   124   126   -
                6     121   77    102   116   102   112   -
                7     133   81    126   135   128   140   136   -
                8     55    51    64    56    59    60    55    63    -
                9     85    62    72    80    72    77    82    89    50    -
                10    48    27    51    53    53    52    43    58    26    31    -


        Source tree size summary:

  num leaves
        4   |======================================== (52)
        5   |===================== (28)
        6   |=============== (20)
        7   |============= (18)
        8   |============= (18)
        9   |============ (16)
        10  |======================= (31)
        11  |=========== (15)



        ----------------------------------------------------
 clann>
```

Figure 10

Output generated by Clann after the execution of a phylip formatted file of multiple source trees.

3.1 Supertree construction

The quickest method of constructing a supertree in Clann is to use the command `nj` which results in a neighbor-joining tree calculated from an average consensus distance matrix to be saved to the file "NJ-tree.ph" and to be displayed on screen (**Fig.** 11).



```
Neighbor-joining settings:
        Distance matrix generation by average consensus method
        Estimation of missing data using 4 point condition distances
        resulting tree saved to file NJ-tree.ph

                                        +------------------- A.aeolic
                              +---------|
                              |         |         +--------- B.burgdo
                              |         +---------|
          +-------------------|                   +--------- C.jejuni
          |                   |
          |                   |                   +--------- S.aureus
          |                   +-------------------|
          |                                       +--------- M.pulmon
+---------|
|         |                   +----------------------------- Synechoc
|         |                   |
|         |         +---------|                   +--------- M.tuberc
|         |         |         |         +---------|
|         |         |         +---------|         +--------- Halobact
|         +---------|                   |
|                   |                   +------------------- D.radiod
|                   |
|                   +--------------------------------------- EcoliK12
|
+----------------------------------------------------------- C.pnL029
```

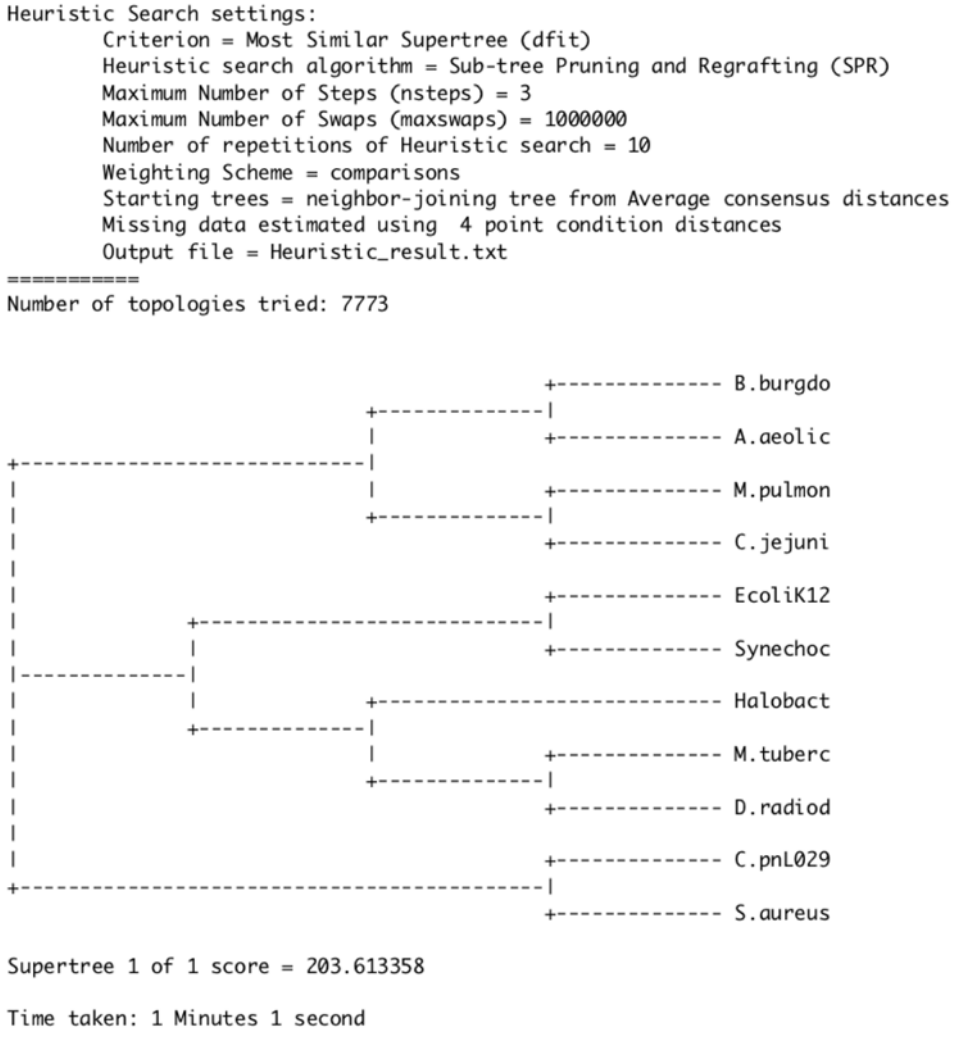
Figure 11

Output generated by the neighbor-joining (nj) command.

It is important to note that the trees displayed by Clann are all unrooted. In this dataset *Halobacterium* (Halobact) is the obvious choice as an outgroup when displaying the best supertree in external phylogeny viewers, as it is the only Archaea in the dataset.

By default Clann uses the MSSA (dfit) algorithm when searching for the best supertree. To carry out a simple heuristic search of tree space for the best supertree, type: `hs`

The resulting tree will be saved to a file called “Heuristic_result.txt” and also displayed on screen (**Fig.** 12).



```
Heuristic Search settings:
        Criterion = Most Similar Supertree (dfit)
        Heuristic search algorithm = Sub-tree Pruning and Regrafting (SPR)
        Maximum Number of Steps (nsteps) = 3
        Maximum Number of Swaps (maxswaps) = 1000000
        Number of repetitions of Heuristic search = 10
        Weighting Scheme = comparisons
        Starting trees = neighbor-joining tree from Average consensus distances
        Missing data estimated using  4 point condition distances
        Output file = Heuristic_result.txt
===========
Number of topologies tried: 7773


                                             +-------------- B.burgdo
                              +--------------|
                              |              +-------------- A.aeolic
+-----------------------------|
|                             |              +-------------- M.pulmon
|                             +--------------|
|                                            +-------------- C.jejuni
|
|                                            +-------------- EcoliK12
|              +-----------------------------|
|              |                             +-------------- Synechoc
|--------------|
|              |              +----------------------------- Halobact
|              +--------------|
|                             |              +-------------- M.tuberc
|                             +--------------|
|                                            +-------------- D.radiod
|
|                                            +-------------- C.pnL029
+--------------------------------------------|
                                             +-------------- S.aureus

Supertree 1 of 1 score = 203.613358

Time taken: 1 Minutes 1 second
```

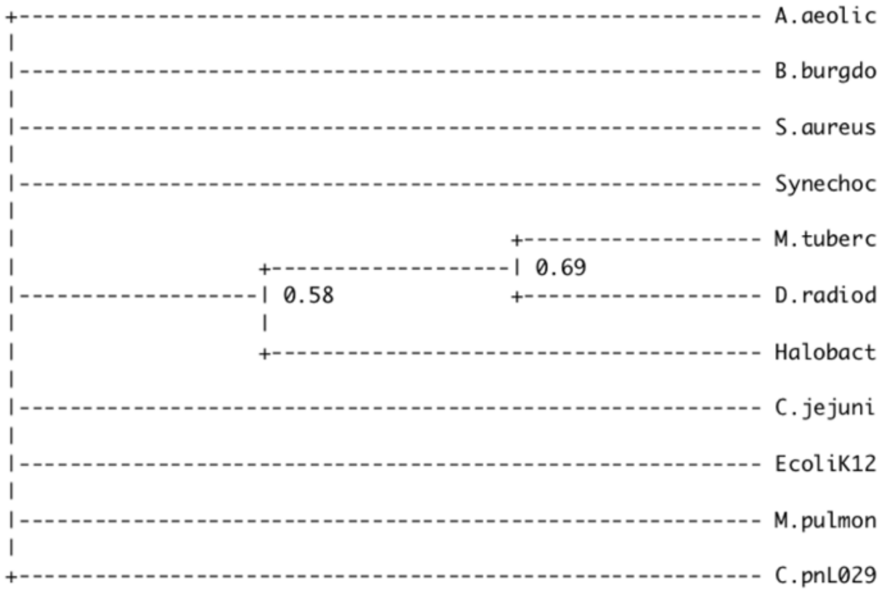
Figure 12
Output generated by Clann using the heuristic search (hs) command under the most similar supertree algorithm (MSSA) criterion.

Comparing the tree from the neighbor-joining (NJ) algorithm to this tree reveals that they are not the same: for instance in the NJ tree *Mycobacterium tuberculosis* (M.tuberc) is most closely related to *Halobacterium* (Halobact), however in this tree *Deinococcus radiodurans* (D.radiod) is its closest relative.

In order to try to resolve to differences between these two trees we can carry out a bootstrapped search of supertree space. In this case we will use the command:

```
boot hsreps=1
```

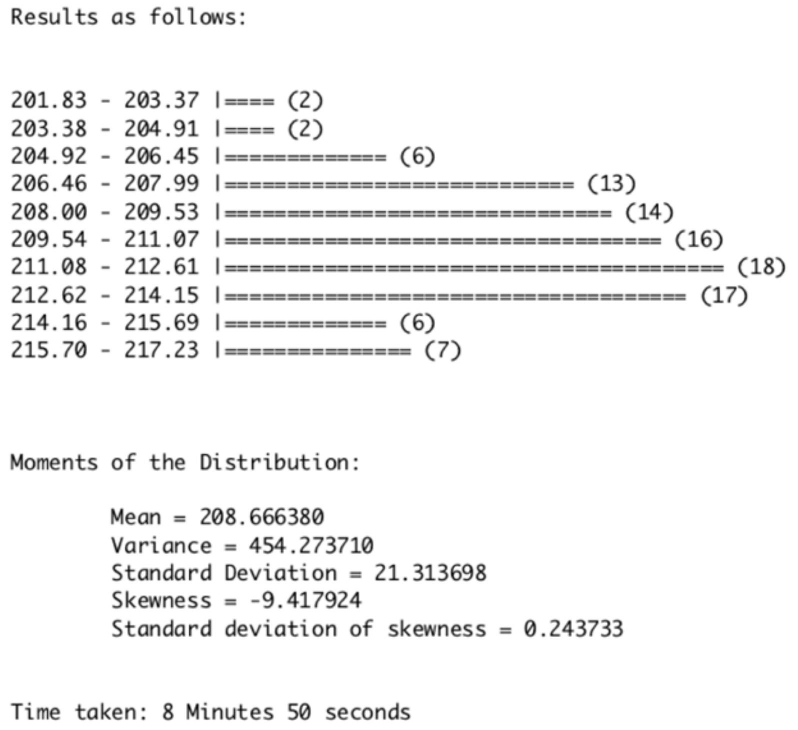
This tells clann to carry out the bootstrap search (100 times by default) but for each replicate only carry out the heuristic search once (the default is 10). This is to speed up the time taken to do the analysis for the purposes of the example.
This analysis returns the results shown in **Fig.** 13.



```
+---------------------------------------------------------- A.aeolic
|
|---------------------------------------------------------- B.burgdo
|
|---------------------------------------------------------- S.aureus
|
|---------------------------------------------------------- Synechoc
|
|                                      +------------------- M.tuberc
|                   +------------------| 0.69
|-------------------| 0.58             +------------------- D.radiod
|                   |
|                   +-------------------------------------- Halobact
|
|---------------------------------------------------------- C.jejuni
|
|---------------------------------------------------------- EcoliK12
|
|---------------------------------------------------------- M.pulmon
|
+---------------------------------------------------------- C.pnL029
```

Figure 13
The output generated by Clann using the bootstrap (boot) command, detailing the support (or lack thereof) in this dataset.

There is obviously very little support for any relationships in this tree, but we can further investigate to see if the phylogenetic signal within the source trees is any better than random noise using the YAPTP test. This test creates randomised versions of the source trees and finds the best supertrees for these randomised datasets. This is carried out 100 times and the scores of the best supertrees for the randomised data is displayed along with the score of the best supertree for the original data. If the phylogenetic signal of the source trees is better than random noise the score of the best supertree should lie well outside the distribution of scores from the randomised data. The command `yaptp` returns the results shown in **Fig.** 14.



```
Results as follows:


201.83 - 203.37 |==== (2)
203.38 - 204.91 |==== (2)
204.92 - 206.45 |============= (6)
206.46 - 207.99 |============================ (13)
208.00 - 209.53 |=============================== (14)
209.54 - 211.07 |=================================== (16)
211.08 - 212.61 |======================================== (18)
212.62 - 214.15 |===================================== (17)
214.16 - 215.69 |============= (6)
215.70 - 217.23 |=============== (7)



Moments of the Distribution:

        Mean = 208.666380
        Variance = 454.273710
        Standard Deviation = 21.313698
        Skewness = -9.417924
        Standard deviation of skewness = 0.243733


Time taken: 8 Minutes 50 seconds
```

Figure 14

The output generated by Clann using the “yet another permutation-tail-probability” (YAPTP) test.

As we know from the hs search the score of the best supertree is 203.6 (using the dfit criterion) then we can say that this score lies within the distribution of random supertrees and leads us to suspect that the overall phylogenetic information in this dataset is no better then random.

The second dataset from within the gamma proteobacteria represent a group of organisms that are nearly at the tips of the prokaryotic tree of life. Follow the same procedure with the corresponding file (10taxonfundamentals.ph.txt) to see if the same conclusion holds for this group.

4. Notes

1. Clann can be used to transform nexus formatted tree files into newick formatted files. This is done by executing the nexus file as normal and then using the command: `showtrees savetrees=yes`. It is also possible to set the name of the file to which the trees are saved, and to stop clann from displaying a graphical representation of each source tree while this is done.
2. Clann can be told only to read the first few characters of each taxa name when reading the source trees into memory. This is useful when it is necessary to have unique identifiers (for instance gene IDs) on the source trees. The option `maxnamelen` in the `exe` command sets this value. If the names are not fixed widths, `maxnamelen=delimited` tells Clann to look for a dot (.) specifying the end of the taxon ID in the trees. For instance using `exe maxnamelen=delimited` on this tree:

   (apple.00121,(orange.1435,lemon.3421), pear.1032);

   Results in clann ignoring the numbers after the dots in the taxa names.
3. The equals sign (=), hyphen (-) and space ( ) are special characters in Clann and by default cannot be used in filenames to be read by clann. If a filename contains one of these characters Clann can only read the name of the file properly by putting the name in inverted commas.

   For example: `exe “my-file.txt”`.
4. The first command that you should run if you don’t know what to do is `help`. This will display the list of the commands that are available. Calling any of the commands followed by a question mark (for instance `hs ?`), will display the options and defaults associated with that command.
5. The command `!` runs a shell terminal on Unix and Mac operating systems allowing system commands can be run without having to quit Clann.
6. Clann can assess supertrees created using other programs. Using the `usertrees` command, clann will read in the file specified and assess all the trees it contains. The best supertree found in the file is displayed.
7. All commands in Clann should be written completely in lowercase, typing the command `boot` is not the same as `Boot` and only the first will be recognised as a valid command.

8. Heuristic and exhaustive searches of supertree space can be interrupted using the key combination “Control-c”. This allows the user to specify if they wish to stop the search now and display the best tree found so far. If this is done during the random sampling phase of a heuristic search, it will allow the user to move straight to the heuristic search without completing the random sampling.
9. Users can assess different configurations of their data by excluding (or including) certain source trees from subsequent commands using the `excludetrees` and `includetrees` commands. Source trees can be selected based on their name, the taxa they contain, their size (number of taxa they contain) or their score when compared to a supertree.
10. Individual (or multiple) taxa can be pruned from the source trees using the command `deletetaxa`. Branch lengths are adjusted to take the deletion of the taxa into account. If the deletion of taxa from a source tree means that there are less than 4 taxa remaining, that source tree is removed from the analysis. Clann will display the names of the source trees removed if this occurs.

References:

1. Aho, A. V., Sagiv, Y., Szymanski, T. G., and Ullman, J. D. (1981) Inferring a Tree from Lowest Common Ancestors with an Application to the Optimization of Relational Expressions. *SIAM Journal on Computing* **10,** 405-21.
2. Gordon, A. D. (1986) Consensus Supertrees: The synthesis of rooted trees containing overlapping sets of laballed leaves. *Journal of Classification* **3,** 335-48.
3. Wilkinson, M., Cotton, J. A., Creevey, C., Eulenstein, O., Harris, S. R., Lapointe, F. J., Levasseur, C., McInerney, J. O., Pisani, D., and Thorley, J. L. (2005) The shape of supertrees to come: tree shape related properties of fourteen supertree methods. *Syst Biol* **54,** 419-31.
4. Liu, F. G., Miyamoto, M. M., Freire, N. P., Ong, P. Q., Tennant, M. R., Young, T. S., and Gugel, K. F. (2001) Molecular and morphological supertrees for eutherian (placental) mammals. *Science* **291,** 1786-9.
5. Beck, R. M., Bininda-Emonds, O. R., Cardillo, M., Liu, F. G., and Purvis, A. (2006) A higher-level MRP supertree of placental mammals. *BMC Evol Biol* **6,** 93.
6. Dagan, T., and Martin, W. (2006) The tree of one percent. *Genome Biol* **7,** 118.
7. Creevey, C. J., Fitzpatrick, D. A., Philip, G. K., Kinsella, R. J., O'Connell, M. J., Pentony, M. M., Travers, S. A., Wilkinson, M., and McInerney, J. O. (2004) Does a tree-like phylogeny only exist at the tips in the prokaryotes? *Proc R Soc Lond B Biol Sci* **271,** 2551-8.
8. McInerney, J. O., and Wilkinson, M. (2005) New methods ring changes for the tree of life. *Trends Ecol Evol* **20,** 105-7.
9. Pollard, D. A., Iyer, V. N., Moses, A. M., and Eisen, M. B. (2006) Widespread discordance of gene trees with species tree in Drosophila: evidence for incomplete lineage sorting. *PLoS Genet* **2,** e173.
10. Doolittle, W. F. (1999) Lateral genomics. *Trends Cell Biol* **9,** M5-8.
11. Jain, R., Rivera, M. C., and Lake, J. A. (1999) Horizontal gene transfer among genomes: the complexity hypothesis. *Proc Natl Acad Sci U S A* **96,** 3801-6.
12. Garcia-Vallve, S., Romeu, A., and Palau, J. (2000) Horizontal gene transfer in bacterial and archaeal complete genomes. *Genome Res* **10,** 1719-25.
13. Brown, J. R., Douady, C. J., Italia, M. J., Marshall, W. E., and Stanhope, M. J. (2001) Universal trees based on large combined protein sequence data sets. *Nature Genetics* **28,** 281-85.
14. Kim, J., and Salisbury, B. A. (2001) A tree obscured by vines: horizontal gene transfer and the median tree method of estimating species phylogeny. *Pac Symp Biocomput***,** 571-82.
15. Dutta, C., and Pan, A. (2002) Horizontal gene transfer and bacterial diversity. *J Biosci* **27,** 27-33.
16. Dagan, T., and Martin, W. (2007) Ancestral genome sizes specify the minimum rate of lateral gene transfer during prokaryote evolution. *Proc Natl Acad Sci U S A* **104,** 870-5.
17. Woese, C. R. (2002) On the evolution of cells. *Proc Natl Acad Sci U S A* **99,** 8742-47.
18. Doolittle, W. F. (1998) A paradigm gets shifty. *Nature* **392,** 15-6.

19. Pisani, D., Cotton, J. A., and McInerney, J. O. (2007) Supertrees disentangle the chimerical origin of eukaryotic genomes. *Mol Biol Evol* **24,** 1752-60.

20. Hendy, M. D., and Penny, D. (1989) A framework for the quantitative study of evolutionary trees. *Systematic Zoology* **38,** 297-309.

21. Foster, P. G., and Hickey, D. A. (1999) Compositional bias may affect both DNA-based and protein-based phylogenetic reconstructions. *J Mol Evol* **48,** 284-90.

22. Keane, T. M., Creevey, C. J., Pentony, M. M., Naughton, T. J., and McLnerney, J. O. (2006) Assessment of methods for amino acid matrix selection and their use on empirical data shows that ad hoc assumptions for choice of matrix are not justified. *BMC Evol Biol* **6,** 29.

23. Rokas, A., Williams, B. L., King, N., and Carroll, S. B. (2003) Genome-scale approaches to resolving incongruence in molecular phylogenies. *Nature* **425,** 798-804.

24. Ciccarelli, F. D., Doerks, T., von Mering, C., Creevey, C. J., Snel, B., and Bork, P. (2006) Toward automatic reconstruction of a highly resolved tree of life. *Science* **311,** 1283-7.

25. Philippe, H., Snell, E. A., Bapteste, E., Lopez, P., Holland, P. W., and Casane, D. (2004) Phylogenomics of eukaryotes: impact of missing data on large alignments. *Mol Biol Evol* **21,** 1740-52.

26. Tekaia, F., Lazcano, A., and Dujon, B. (1999) The genomic tree as revealed from whole proteome comparisons. *Genome Research* **9,** 550-57.

27. Snel, B., Huynen, M. A., and Dutilh, B. E. (2005) Genome Trees and The Nature of Genome Evolution. *Annu Rev Microbiol*.

28. Huson, D. H., and Steel, M. (2004) Phylogenetic trees based on gene content. *Bioinformatics* **20,** 2044-9.

29. Hahn, M. W., De Bie, T., Stajich, J. E., Nguyen, C., and Cristianini, N. (2005) Estimating the tempo and mode of gene family evolution from comparative genomic data. *Genome Res* **15,** 1153-60.

30. Novozhilov, A. S., Karev, G. P., and Koonin, E. V. (2005) Mathematical modeling of evolution of horizontally transferred genes. *Mol Biol Evol* **22,** 1721-32.

31. Lake, J. A., and Rivera, M. C. (2004) Deriving the genomic tree of life in the presence of horizontal gene transfer: conditioned reconstruction. *Mol Biol Evol* **21,** 681-90.

32. Baum, B. R. (1992) Combining Trees as a Way of Combining Data Sets for Phylogenetic Inference, and the Desirability of Combining Gene Trees. *Taxon* **41,** 3-10.

33. Ragan, M. A. (1992) Matrix Representation in Reconstructing Phylogenetic-Relationships among the Eukaryotes. *Biosystems* **28,** 47-55.

34. Lapointe, F.-J., and Cucumel, G. (1997) The average consensus procedure: combination of weighted trees containing identical or overlapping sets of taxa. *Syst Biol* **46,** 306-12.

35. Lapointe, F. J., and Levasseur, C. (2004) *in* "Phylogenetic Supertrees: Combining information to reveal the Tree of Life" (Bininda-Emonds, O. R. P., Ed.), Vol. 4, Kluwer Academic, Dordrecht.

36. Saitou, N., and Nei, M. (1987) The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Mol Biol Evol* **4,** 406-25.

37. Creevey, C. J., and McInerney, J. O. (2005) Clann: investigating phylogenetic information through supertree analyses. *Bioinformatics* **21,** 390-2.

38. Swofford, D. L. (2002) PAUP*. Phylogenetic Analysis Using Parsimony (*and other methods). Version 4, Sinauer Associates, Sunderland, Massachusetts.

39. Maddison, D. R., Swofford, D. L., and Maddison, W. P. (1997) Nexus: An extensible file format for systematic information. *Systematic Biology* **46,** 590-621.

40. Page, R. D. M. (1996) TREEVIEW: An application to display phylogenetic trees on personal computers. *Computer Applications in the Biosciences* **12,** 357-58.

41. Letunic, I., and Bork, P. (2007) Interactive Tree Of Life (iTOL): an online tool for phylogenetic tree display and annotation. *Bioinformatics* **23,** 127-8.