Chapter 4
Consensus trees

Consensus trees are a convenient way to summarise the agreement between two or more trees. This chapter describes the consensus methods available in COMPONENT.

At present COMPONENT implements five consensus methods: strict, majority rule, semi-strict, Nelson, and Adams consensus. A consensus method not discussed in this chapter but also available in COMPONENT is the agreement subtree (Kubicka et al, 1992; also called the common pruned tree [Finden and Gordon, 1985]). This method is described in Chapter 5.

Overview

Cluster methods

Strict and *majority rule* consensus methods are based on simple counts of the frequency of occurrence of clusters in the set of trees; the strict consensus tree contains only those clusters found in all the trees, the majority rule consensus tree contains all clusters occurring in at least half the trees. In the case of two trees, the majority rule consensus tree is also the strict consensus tree.

The *semi-strict* consensus tree includes the clusters retained by the strict consensus tree, but also contains any clusters that are not contradicted by any other clusters in the profile. The *Nelson* consensus tree comprises the set of mutually compatible clusters that are most frequently replicated in the set of trees. In the case of two trees the semi-strict consensus tree is the same as the Nelson consensus tree.

Intersection methods

At present COMPONENT implements a single intersection method: *Adams* consensus.

Choosing a consensus method

Some consensus methods are more suited to certain tasks than are others. Strict consensus is useful primarily if you want to see which clusters are always supported by all the trees in the profile. If your input trees contain polytomies then there may be clusters in the profile that are never contradicted but which do not occur in all the trees (this may be the case if the trees were produced by a parsimony program that collapses zero-length branches). In this case semi-strict consensus would be appropriate.

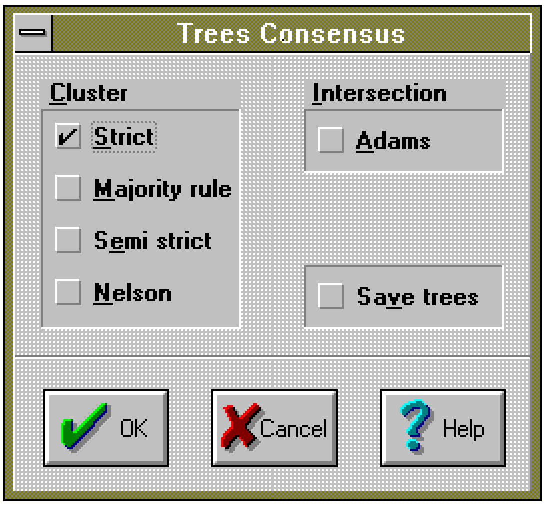
Majority rule consensus has often been used to summarise bootstrap trees. Because it is also a median consensus tree it is ideally suited to constructing "confidence intervals" for trees using the method described by Sanderson (1989; see Chapter 8).

Adams consensus trees are often maligned as difficult to interpret because they contain clusters not found in any trees in the profile. This is not unexpected, given that the method is not a cluster based consensus method. Adams consensus is particularly useful for identifying common tree structure when one or more taxa have very different positions in the trees in the profile.

Computing consensus trees

Choose **Consensus** from the **Trees** menu. The following dialog box will appear:

Figure 4.1
The Consensus dialog box



Trees Consensus

Cluster

☑ Strict

□ Majority rule

□ Semi strict

□ Nelson

Intersection

□ Adams

□ Save trees

Use the check boxes to choose which methods you would like. You can also elect to save the consensus tree(s) to a disk file by checking the **Save trees** check box.

Output

For the cluster based consensus methods COMPONENT outputs all the clusters found and their frequencies using a cluster table such as:

```
             1
    1234567890
    ----------
  1 **........    20
  2 ..***.....    16
  .
  .
  .
 25 .*..*.....     2
```

The clusters are listed in the order in which they are encountered by the program. The elements of each cluster are represented by an asterix "*" so that, for example, the cluster {1, 2} is represented by

```
    **........
```

The number of trees each cluster occurs in is listed to the right of each cluster.

Semi-strict and Nelson consensus methods also output the pairwise compatibility matrix for all the clusters in the set of trees. Two clusters A and B are compatible if A ∩ B ∈ {∅, A, B}, that is, A and B are identical, or either A or B is nested inside the other, or A and B are disjoint. If two clusters are compatible the matrix contains a "1", otherwise it contains a ".". Below the matrix is a line showing which clusters (marked by "*") are compatible with every other cluster in the set of trees:

```
2| 1
3| 11
4| 1.1
5| 1111
6| 1..11
 +--------
   123456

   *...*.
```

In this example, clusters 1 and 2 are compatible, but clusters 2 and 4 are not. Clusters 1 and 5 are compatible with every other cluster (and hence would appear in a semi-strict consensus tree).

The consensus trees are output in the display buffer (and if the **Save trees** option has been chosen, written to a disk file). Cluster based consensus trees have their internal nodes labelled with the percentage frequency of each cluster in the profile:

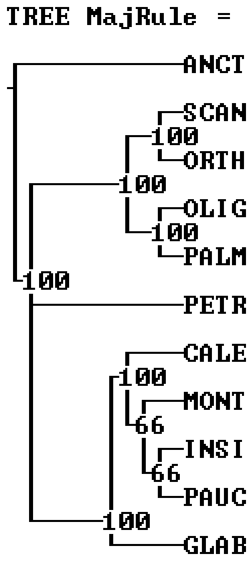


Figure 4.2
A majority rule consensus tree with the clusters labelled with their frequency in the profile

If the trees in the profile are unrooted then the first cluster (containing all but one leaf) is arbitrary and hence is not labelled (as in the tree above).

Strict consensus

The strict consensus tree of a collection of trees contains only those clusters found in all the trees in the profile.

☞ *In the literature the strict consensus tree has often been confused with the Nelson (1979) consensus tree, hence most published "Nelson" consensus trees are in fact strict consensus trees. Note that the* ***nelsen*** *command in Hennig86 computes strict, not Nelson consensus trees.*

Algorithm

If no other cluster based consensus method is chosen, COMPONENT uses Day's (1985) algorithm. Otherwise, the program first stores all the clusters in a hash table, then builds the strict consensus tree using the clusters that occur in every tree.

Rooted versus unrooted trees

COMPONENT arbitrarily roots unrooted trees with the first leaf before computing the strict consensus tree. The choice of leaf does not affect the result.

Output

If no other cluster based consensus method is chosen, COMPONENT outputs just the strict consensus tree, otherwise COMPONENT also displays a table of the frequencies of each cluster in the profile.

Majority rule consensus

The majority rule consensus tree of a profile contains only those clusters found in a majority (> 50%)of the trees in the profile (Margush and McMorris, 1981).

The majority rule consensus tree is also a *median* tree, in that the sum of the pairwise partition metric distances (see Chapter 5) between the consensus tree and the trees in the profile is the minimum possible for any tree. If the number of the trees in the profile is odd then the majority rule tree is the unique median tree. If the number of trees is even and two or more incompatible clusters are found in exactly half of the trees then there will be more than one median tree.

Algorithm

COMPONENT stores all the clusters in a hash table, then builds the majority rule consensus tree using those clusters that occur in >50% of the trees.

Rooted versus unrooted trees

COMPONENT arbitrarily roots unrooted trees with the first leaf before computing the majority consensus tree. This choice of leaf does not affect the result.

Output

COMPONENT displays the following information:

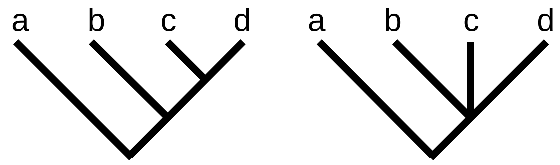
1. A table of all the clusters in the profile and their frequencies.

2. The majority rule consensus tree, with the internal nodes labelled with the percentage of trees in the profile that the corresponding cluster occurs in. If the input trees are unrooted the first cluster (containing all but one leaf) is arbitrary and so is not labelled.

Semi-strict consensus

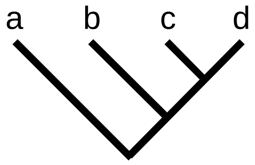
A semi-strict consensus tree contains all the uncontradicted clusters in a profile of trees. Hence it contains all the clusters of the strict consensus tree, together with any additional clusters that are compatible with every other cluster in the profile. As an example consider these two trees:

Figure 4.3
A profile of two trees.



Although the cluster {c, d} occurs in he first tree but not the second, it is compatible with all the clusters in the second tree, hence it appears in the semi-strict consensus tree:

Figure 4.4
Semi-strict consensus tree for the two trees in Figure 4.3



The semi-strict consensus tree was described by Bremer (1990) as the "combinable component consensus tree." For consistency I've followed Swofford (1990) in calling this method "semi-strict consensus."

Algorithm

COMPONENT uses a hash table to store the clusters in the profile, then constructs a pairwise compatibility matrix between all the clusters. Those clusters having no incompatibilities are retained in the consensus tree.

Rooted versus unrooted trees

COMPONENT arbitrarily roots unrooted trees with the first leaf before computing the semi-strict consensus tree. The choice of leaf does not affect the result.

Output

COMPONENT displays the following information:

1. A table of all the clusters in the profile and their frequencies.

2. The pairwise compatibility matrix for all clusters in the profile.

3. The semi-strict consensus tree, with the internal nodes labelled with the percentage of trees in the profile that the corresponding cluster occurs in. If the input trees are unrooted the first cluster (containing all but one leaf) is arbitrary and so is not labelled.

Nelson consensus

A Nelson (1979) consensus tree is the largest clique of most frequently replicated clusters in a profile of trees. A *clique* of clusters is a set of mutually compatible clusters, so that every cluster in the clique is compatible with every other cluster in the same clique. If there is more than one such clique then the Nelson consensus tree is the set of clusters common to those cliques.

This method partitions the clusters in the profile into three categories:

1. clusters in the consensus tree ("true").

2. clusters in some but not all of the cliques with the most replications of clusters ("ambiguous").

3. clusters in neither category (1) nor (2) ("false").

☞ *In the literature the Nelson (1979) consensus tree has often been confused with the strict consensus tree. The two methods are quite different (Page, 1989).*

Algorithm

COMPONENT uses the algorithm described by Page (1989).

Rooted versus unrooted trees

COMPONENT arbitrarily roots unrooted trees with the first leaf before computing the Nelson consensus tree. The choice of leaf does not affect the result.

Output

COMPONENT displays the following information:

1. A table of all the clusters in the profile and their frequencies.

2. The pairwise compatibility for all clusters in the profile.

3. The three cluster partitions: **T**rue, **A**mbiguous, and **F**alse. Each partition is displayed in the form

   ```
   T: [1 2 6..9 10]
   ```

   where the notation 6..9 means that clusters 6 through to 9 (i.e., 6, 7, 8, and 9) are members of the set T.

4. The Nelson consensus tree, with the internal nodes labelled with the percentage of trees in the profile that the corresponding cluster occurs in. If the input trees are unrooted the first cluster (containing all but one leaf) is arbitrary and so is not labelled.

Adams consensus

An Adams consensus tree contains the nestings common to all trees in a profile. Given two sets of leaves, A and B, and a tree T, set A nests inside set B if (1) A is a subset of B, and (2) the leaves in set A have a more recent common ancestor in T than do the leaves in set B. For example, given the tree (A,(B,(C,D))), the sets {C,D}, {B,C}, and {B,D} all nest inside {A,B,C,D}, but only {C,D} nests inside {B,C,D}.

Algorithm

COMPONENT uses the algorithm described by Adams (1986:315).

Rooted versus unrooted trees

Adams consensus trees are not invariant under different rootings. If the trees in the profile are unrooted, COMPONENT treats them as if they were rooted trees.

Output

COMPONENT draws the Adams consensus tree in the display buffer. While COMPONENT is computing the Adams consensus tree it tells you which tree it is processing (the program usually requires several passes through the set of trees) and how many clusters with > 3 leaves the consensus tree currently contains.