Summary: *SLINK: An optimally efficient algorithm for the single-link cluster method*, by R Sibson (King's College Research Centre, King's College, Cambridge and Cambridge University Statistical Laboratory when paper was written in 1972)

Main point Sibson gives an $O(n^2)$ algorithm for single-linkage clustering, and proves that this algorithm achieves the theoretically optimal lower time bound for obtaining a single-linkage dendrogram. This improves upon the naive $O(n^3)$ implementation of single linkage clustering.

A single linkage dendrogram is a tree, where each level of the tree corresponds to a different threshold dissimilarity measure h. The nodes of a dataset are grouped into "equivalence classes" $c(h)$ at each level of the dendrogram, where two classes C_i and C_j are merged if there is a pair of "OTU's" (vertices) $v_i \in C_i$ and $v_j \in C_j$ such that the dissimilarity measure between v_i and v_j is less than h, or $D(v_i, v_j) < h$. For example, consider a set of 10 vertices $v_1, \ldots, v_{10}$ for which the dissimilarity matrix D is given below, with D_{ij} equal to the dissimilarity between v_i and v_j.

$$D = \begin{pmatrix} 0 & 1.2 & 5 & 5 & 4.2 & 7 & 9 & 7.6 & 11 & 4.3 \\ 1.2 & 0 & 3.4 & 4.1 & 5 & 6 & 4.1 & 6.4 & 5.3 & 4.5 \\ 5 & 3.4 & 0 & 2.1 & 6 & 6.2 & 4.6 & 9 & 11.3 & 22 \\ 5 & 4.1 & 2.1 & 0 & 11 & 5 & 13 & 4.1 & 4.3 & 5.5 \\ 4.2 & 5 & 6 & 11 & 0 & 1.9 & 7 & 9 & 5.5 & 4.3 \\ 7 & 6 & 6.2 & 5 & 1.9 & 0 & 7.5 & 5.6 & 6.3 & 4.5 \\ 9 & 4.1 & 4.6 & 13 & 7 & 7.5 & 0 & 3.6 & 8 & 10 \\ 7.6 & 6.4 & 9 & 4.1 & 9 & 5.6 & 3.6 & 0 & 4.9 & 2.9 \\ 11 & 5.3 & 11.3 & 4.3 & 5.5 & 6.3 & 8 & 4.9 & 0 & 1.4 \\ 4.3 & 4.5 & 22 & 5.5 & 4.3 & 4.5 & 10 & 2.9 & 1.4 & 0 \end{pmatrix}$$

Suppose we take four cut-off dissimilarity measures h_1, h_2, h_3, h_4 and produce the dendrogram according to these thresholds. An example illustrating how the 10 vertices are grouped into equivalence classes at each level is shown in Figure 1. Since no dissimilarity is at or below 1, each vertex or "OTU" is its own equivalence class at the level corresponding to $h_1 = 1$. At the next level, however, we see that some classes have been merged together because several dissimilarity measures are below $h_2 = 2$. We can see that $c(h_2)$ consists of 6 equivalence classes, $c(h_3)$ has 3 equivalence classes, and $c(h_4 = 4)$ aggregates all the vertices into one equivalence class. In single linkage clustering, the number of levels in the tree is determined by the nearest-neighbor criterion – at each level, at least one new merge is made between two clusters, and the merge is made for clusters C_i and C_j if the minimal distance between vertices $v_i \in C_i$ and $v_j \in C_j$ is the smallest such distance across all the clusters. In other words, the nearest neighbors between clusters C_j and C_i are found, and if these neighbors are closer than all the other nearest-neighbor pairs, then C_i and C_j are merged. A single

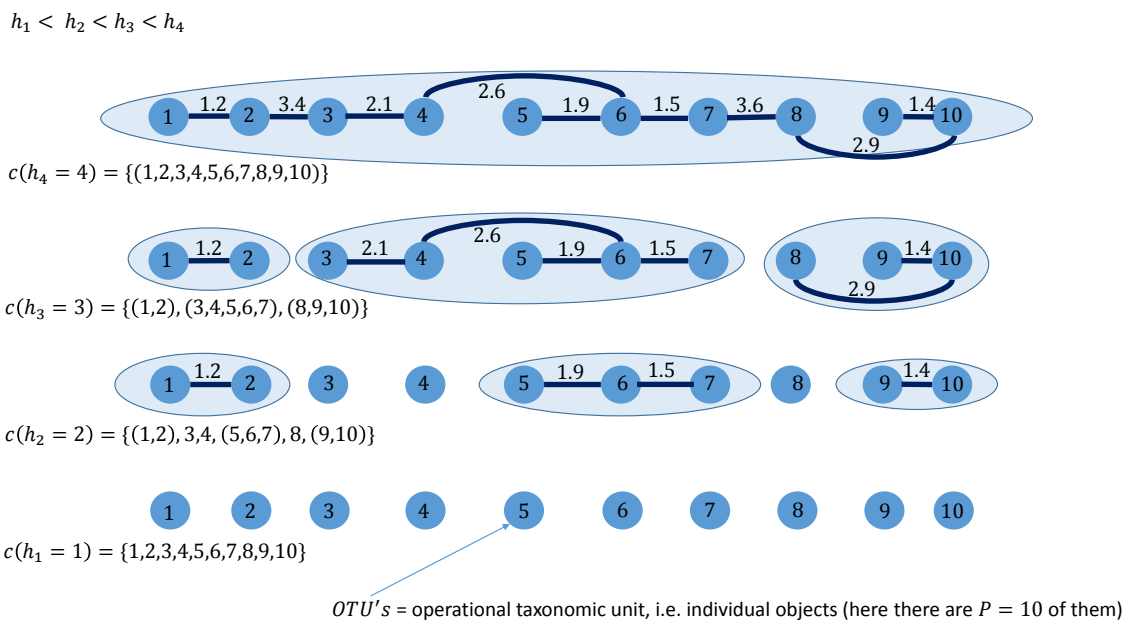

Figure 1: An example of the equivalence classes $c(h)$ formed at each level of the dendrogram with cutoff thresholds h.

linkage dendrogram for the dissimilarity matrix in the example is shown in Figure 2. A naive algorithm constructing this dendrogram would proceed as follows:

1) Construct the dissimilarity matrix D ($O(n^2)$), and initialize each cluster to contain a single vertex, set the level in the dendrogram to $i = 0$

2) Increment i, find the smallest dissimilarities between two clusters $O(n^2)$, and merge the clusters together. Update D to contain one less row and column than before, corresponding to this merge. In the example above, after merging vertices 1 and 2 we would have:

$$D = \begin{pmatrix} 0 & 3.4 & 4.1 & 4.2 & 6 & 4.1 & 6.4 & 5.3 & 4.3 \\ 3.4 & 0 & 2.1 & 6 & 6.2 & 4.6 & 9 & 11.3 & 22 \\ 4.1 & 2.1 & 0 & 11 & 5 & 13 & 4.1 & 4.3 & 5.5 \\ 4.2 & 6 & 11 & 0 & 1.9 & 7 & 9 & 5.5 & 4.3 \\ 6 & 6.2 & 5 & 1.9 & 0 & 7.5 & 5.6 & 6.3 & 4.5 \\ 4.1 & 4.6 & 13 & 7 & 7.5 & 0 & 3.6 & 8 & 10 \\ 6.4 & 9 & 4.1 & 9 & 5.6 & 3.6 & 0 & 4.9 & 2.9 \\ 5.3 & 11.3 & 4.3 & 5.5 & 6.3 & 8 & 4.9 & 0 & 1.4 \\ 4.3 & 22 & 5.5 & 4.3 & 4.5 & 10 & 2.9 & 1.4 & 0 \end{pmatrix}$$

Notice that D is now a 9×9 matrix, and the distance from cluster

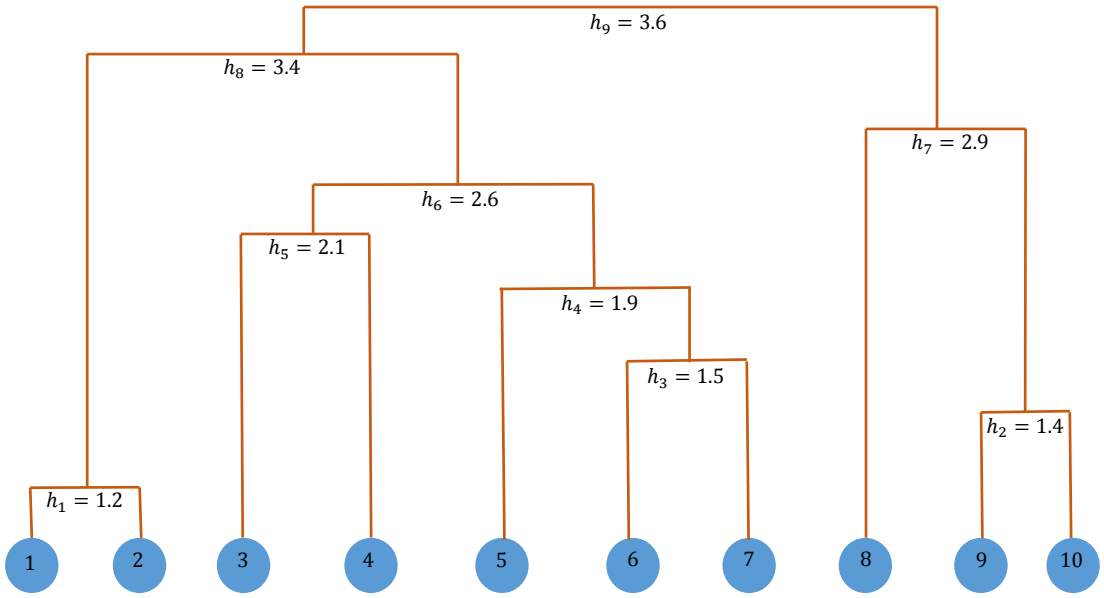

Figure 2: An example of a single linkage dendrogram, where each level corresponds to a threshold h that is the minimum nearest-neighbor link between clusters at the previous level.

$C_1 = \{v_1, v_2\}$ to the other vertices is given in the first row and column of the new D. These distances are simply the minima of the distances of v_1 and v_2 to the other vertices. For instance, $d(\{v_1, v_2\}, v_{10}) = \min(d(v_2, v_{10}), d(v_1, v_{10})) = 4.3$. Level i of the dendrogram contains the newly merged cluster and everything below it as equivalence classes. In the example, level 1 contains the 9 clusters or equivalence classes: $\{1, 2\}, 3, 4, 5, 6, 7, 8, 9, 10$. The threshold h_i at this level is the smallest distance just found, so in the example $h_1 = 1.2$.

3) delete the row and column corresponding to each of these clusters, and add a new row and column corresponding to the new, merged cluster;

4) repeat steps (2) and (3) until all the vertices are merged into one cluster

As written, the naive single linkage algorithm takes $O(n^3)$ time, because at each of $O(n)$ iterations we take $O(n^2)$ time to find the smallest nearest-neighbor link. Sibson realized that we don't need to do this at each stage, because all $O(n^2)$ dissimilarities are already computed when we computed the dissimilarity matrix – therefore, at each step of SLINK, we simply maintain an array of size $O(n)$ that tells us the nearest neighbor of each vertex and the distance of that neighbor. While computing the dissimilarity matrix, we can also compute the

nearest neighbor of each vertex and store it in an array A_V and its corresponding distance in an array A_D. After merging two clusters C_i and C_j, where $i < j$ and C_i is indexed by i, into C_{ij}, we update the dissimilarity matrix the same way as in the naive algorithm. We then update $A_D(i)$ to be the new smallest distance to C_{ij}, which is found by taking the minimum D_{ik} and D_{jk} for all $k \neq i, j$, taking $O(n)$ time, and set $A_V(i)$ to the cluster index corresponding to this distance. Set $A_D(j)$ to be ∞. For all other clusters C_k, if $A_V(C_k) = i$ or $A_V(C_k) = j$, update this entry to $A_V(C_k) = i$, otherwise leave $A_V(C_k)$ as it was. Thus, instead of searching for the smallest dissimilarity in each step in the updated dissimilarity matrix, we instead take $O(n)$ time to find it in A_D and merge the corresponding clusters that are stored in A_V. For the example above, we would start with:

$$A_D = \begin{bmatrix} 1.2 & 1.2 & 2.1 & 1.9 & 1.5 & 1.5 & 2.9 & 1.4 & 1.4 \end{bmatrix}$$

$$A_V = \begin{bmatrix} 2 & 1 & 4 & 3 & 6 & 7 & 6 & 10 & 10 & 9 \end{bmatrix}$$

In our example, after merging vertices v_1 and v_2, we would get the following:

$$A_D = \begin{bmatrix} 3.4 & \infty & 2.1 & 1.9 & 1.5 & 1.5 & 2.9 & 1.4 & 1.4 \end{bmatrix}$$

$$A_V = \begin{bmatrix} 3 & 1 & 4 & 3 & 6 & 7 & 6 & 10 & 10 & 9 \end{bmatrix}$$

We proceed in this way, at each step saving the equivalence classes that make up the dendrogram. Thus, each iteration takes $O(n)$ time as opposed to $O(n^2)$, giving a time complexity of $O(n^2)$ for SLINK.