Bioinformatics, 38(3), 2022, 663–670
https://doi.org/10.1093/bioinformatics/btab723
Advance Access Publication Date: 20 October 2021
Original Paper

Sequence analysis

AncestralClust: clustering of divergent nucleotide sequences by ancestral sequence reconstruction using phylogenetic trees

Lenore Pipes [1,*] **and Rasmus Nielsen** [1,2,3,*]

[1]Department of Integrative Biology, University of California-Berkeley, Berkeley, CA 94707, USA, [2]Department of Statistics, University of California-Berkeley, Berkeley, CA 94707, USA and [3]Globe Institute, University of Copenhagen, 1350 København K, Copenhagen, Denmark

*To whom correspondence should be addressed.

Associate Editor: Inanc Birol

Received on March 3, 2021; revised on September 30, 2021; editorial decision on October 14, 2021; accepted on October 15, 2021

Abstract

Motivation: Clustering is a fundamental task in the analysis of nucleotide sequences. Despite the exponential increase in the size of sequence databases of homologous genes, few methods exist to cluster divergent sequences. Traditional clustering methods have mostly focused on optimizing high speed clustering of highly similar sequences. We develop a phylogenetic clustering method which infers ancestral sequences for a set of initial clusters and then uses a greedy algorithm to cluster sequences.

Results: We describe a clustering program *AncestralClust*, which is developed for clustering divergent sequences. We compare this method with other state-of-the-art clustering methods using datasets of homologous sequences from different species. We show that, in divergent datasets, AncestralClust has higher accuracy and more even cluster sizes than current popular methods.

Availability and implementation: AncestralClust is an Open Source program available at https://github.com/lpipes/ancestralclust.

Contact: lpipes@berkeley.edu or rasmus_nielsen@berkeley.edu

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Traditional clustering methods such as UCLUST (Edgar, 2010) and CD-HIT (Fu *et al.*, 2012) use hierarchical or greedy algorithms that rely on user input of a sequence identity threshold. These methods were developed for high speed clustering of a vast quantity of highly similar sequences (Ghodsi *et al.*, 2011; Li *et al.*, 2001; Edgar, 2010) and, generally, these methods are considered unreliable for identity thresholds <75% because of either the poor quality of alignments at low identities (Zou *et al.*, 2018) or because the performance of the method drops dramatically with low identities (Huang *et al.*, 2010). At low identities, these methods produce uneven clusters where the majority of sequences are contained in only one or a few clusters (Chen *et al.*, 2018). A high variance in cluster sizes may reduce the utility of clustering for many practical purposes since the goal of clustering typically is to reduce computational complexity of downstream analyses that are limited by the size of the largest clusters. Clustering of divergent sequences is an important step in genomics analysis because it allows for an early divide-and-conquer strategy that will significantly increase the speed of downstream analyses (Zheng *et al.*, 2019) and many fundamental questions in metagenomics can be addressed by clustering of divergent sequences, such as the identification of gene families, and identification of sequences at the order, class or phylum taxonomic levels. Currently, there are no clustering methods that can accurately cluster large taxonomically divergent metabarcoding reference databases such as the Barcode of Life database (Ratnasingham and Hebert, 2007) in relatively even clusters. Only a few other methods, such as SpClust (Matar *et al.*, 2019) and TreeCluster (Balaban *et al.*, 2019), exist for clustering potentially divergent sequences. SpClust creates clusters based on the use of Laplacian Eigenmaps and a Gaussian Mixture Model based on a similarity matrix calculated on all input sequences. While this approach is highly accurate, the calculation of an all-to-all similarity matrix is computationally demanding. For example, an all-by-all comparison for clustering 8 million environmental DNA (eDNA) reads by Rusch *et al.* (2007) took > 1 year on a 100-CPU cluster. TreeCluster uses user-specified constraints for splitting a

phylogenetic tree into clusters. However, TreeCluster requires an input tree and even though some phylogenetic methods exist to estimate trees from a large number of sequences (Stamatakis, 2014), it can also be prohibitively slow for large numbers of divergent sequences where a phylogenetic tree is difficult to estimate reliably. With the increasing size of reference databases (Schoch *et al.*, 2020), there is a need for new computationally efficient methods that can cluster divergent sequences. Here, we present AncestralClust which is specifically developed for clustering of divergent metabarcoding reference sequences in clusters of relatively even size.

2 Materials and methods

To cluster divergent sequences, we developed AncestralClust written in C (Fig. 1). The algorithm proceeds by first (i) selecting r random sequences for pairwise alignment using the wavefront algorithm (Marco-Sola *et al.*, 2020). We choose a random subset of sequences to reduce the computational burden of performing all-to-all alignments. The wavefront algorithm, which is an approximating algorithm for pairwise alignment with affine gap penalty, is used as a tradeoff between computational time and accuracy since it runs in time linear with the sequence length and divergence. If computational time is not a concern the Needleman-Wunsch algorithm (Needleman and Wunsch, 1970), which has computational complexity that is quadratic in sequence length, can also be used for alignments. (ii) A Jukes-Cantor (Jukes *et al.*, 1969) distance matrix is then constructed from the alignments and a Neighbor-joining phylogenetic tree (Saitou and Nei, 1987) is estimated. The Jukes-Cantor model is chosen for computational speed, but more complex models could in principle be used to potentially increase accuracy but also increase computational time. (iii) The leaves are clustered by cutting the b—1 longest branches in the tree to yield b subtrees (corresponding to b clusters of leaf nodes). These subtrees comprise the initial starting clusters. The next step (iv) is to estimate the ancestral sequence in the root of each subtree. To increase accuracy, the sequences in each initial starting cluster are re-aligned in a multiple sequence alignment using kalign3 (Lassmann, 2020). Kalign3 was chosen to perform the multiple sequence alignments because of its accuracy and speed. Next, a new Neighbor-joining tree (Saitou and Nei, 1987) is constructed from each initial starting cluster, and each tree is midpoint rooted (Farris, 1972). The midpoint root method was chosen for computational speed. The ancestral sequences at the root of the tree of each cluster are estimated using the maximum of the posterior probability of each nucleotide using standard programming algorithms from phylogenetics (see e.g. Yang, 2014). The ancestral sequences are used as the representative sequence for each cluster. Next, (v) the remaining sequences (the ones currently not assigned to any cluster) are aligned to each of the b ancestral sequences and a Jukes-Cantor distance is calculated. If the shortest distance to any of the b ancestral sequences is larger than the average distance between clusters, the sequence is saved for assignment using an iterative algorithm where the sequence will be contained in a new cluster. If the shortest distance to any of the b ancestral sequences is less than or equal to the average distance between clusters, the sequence is assigned to each cluster based on the shortest Jukes-Cantor distance from the wavefront algorithm alignment between the sequence and the b ancestral sequences. This iterative algorithm proceeds by repeating steps 1–5 above for the previously unassigned sequences. If the number of sequences is $< r$, then r becomes the number of unassigned sequences. In each iteration after the first iteration, a cut of a branch in the phylogenetic tree is chosen if the branch is longer than the average length of branches cut in the first iteration. We iterate this process until all sequences are assigned to a cluster.

Algorithm 1 provides an overview of the algorithm using the following notation: N is the number of sequences to assign, $A = \{A_1, \ldots, A_N\}$ is the set of sequences, $B = \{B_1, \ldots, B_r\}$ is a set of r randomly chosen sequences, $\Omega = \{\omega_1, \ldots, \omega_k\}$ is the set of all k clusters to be returned by the algorithm, i.e. a partition of A into k sets,

Algorithm 1: Overview of algorithm

Input : N, $A = \{A_1,\ldots,A_N\}$, r, b
Output: k, $\Omega = \{\omega_1, \ldots, \omega_k\}$
$\Omega \leftarrow \emptyset$, $k \leftarrow 0$, $f \leftarrow 0$
while $N > 0$ **do**
  if $N < r$ **then**
    $r \leftarrow N$
  $B \leftarrow$ random (r, A)
  $A \leftarrow A \setminus B$
  $N \leftarrow N - r$
  $D \leftarrow$ Jukes-Cantor(pairwise_alignment(B))
  $Tree \leftarrow$ Neighbor-joining(D)
  if $f = 0$ **then**
    $(E, q) \leftarrow$ max_branch_cut$(b - 1, Tree)$
  else
    $(E, b) \leftarrow$ fixed_branch_cut$(q, Tree)$
  $k \leftarrow k + b$
  for $E_i \in E$ **do**
    $D_i \leftarrow$ Jukes-Cantor(multiple_alignment(E_i))
    $Tree_i \leftarrow$ Neighbor-joining(D_i)
    $\gamma_i \leftarrow$ Midpoint_root($Tree_i$)
    $X_i \leftarrow$ Ancestral_sequence estimation($\gamma_i, Tree_i$)
  $\nu \leftarrow \overline{d(D_i, D_j)}$;
  for $A_i \in A$ **do**
    for $X_j \in \{X_1, \ldots, X_b\}$ **do**
      $d_j \leftarrow$ Jukes-Cantor(alignment(A_i, X_j))
    $\mu \leftarrow min_j\{d_j\}$
    $m \leftarrow argmin_j\{d_j\}$
    if $\mu < \nu$ **then**
      $E_m \leftarrow E_m \cup A_i$
      $A \leftarrow A \setminus A_i$
      $N \leftarrow N - 1$
  $\Omega \leftarrow \Omega \cup E$
  $f \leftarrow f + 1$

D is a matrix of Jukes-Cantor distances, *Tree* is a binary tree, $E = \{E_1, \ldots, E_b\}$ is a partition of sequences, q is the average length of branches cut in the first iteration, γ is a midpoint rooting (a point in the tree), X_i is the ancestral sequence reconstruction at γ_i for $Tree_i$ estimated from the sequences in E_i, and ν is the average distance between clusters.Here, $B \leftarrow$ random (r, A) is an operation in which r distinct sequences are selected uniformly at random from A to form the set of sequences B.

$(E, q) \leftarrow$ max_branch_cut $(b - 1, Tree)$ is an operation in which the set of leaf nodes in *Tree* are divided into a partition E with b subsets, by cutting the b—1 longest branches in *Tree*, and in which q is set to be equal to the average length of the branches cut.

$E \leftarrow$ fixed_branch_cut(q, *Tree*) is an operation in which the set of leaf nodes in *Tree* are partitioned by cutting all branches in *Tree* with a length larger than q to form a partition E with b subsets.

$d(D_i, D_j)$ is the average Jukes-Cantor distance between sequences in E_i and E_j and $\overline{d(D_i, D_j)}$ is the average of this quantity over all pairs $(E_i, E_j), i \neq j$, in E.

In praxis, only one or two iterations are needed for most datasets if r is defined to be sufficiently large. The method is parallelized during the calculation of D, during the multiple sequence alignment (multiple_alignment (E_i)), and the assignment of unassigned sequences to clusters.

This procedure relies on arbitrary choices of r and b. However, we calibrate the choices of r and b using a procedure described in Supplementary Appendix SA.

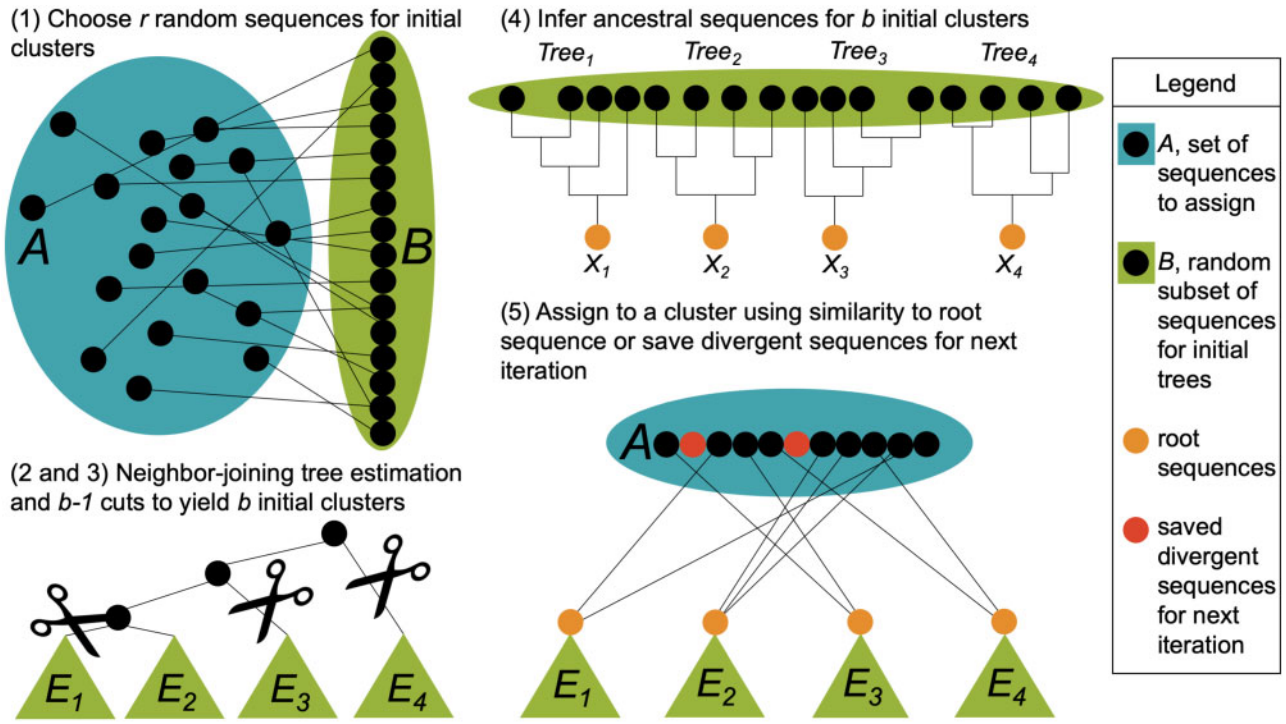

Fig. 1. Overview of AncestralClust. In (1), *r* random sequences are chosen from *A* for the initial clusters. (2) Using the *r* random sequences a Jukes-Cantor distance matrix is constructed. Using the distance matrix, a Neighbor-joining tree is estimated and in (3) *b*—1 cuts are made to create *b* clusters. In (4), each cluster is aligned using a multiple sequence alignment and a Neighbor-joining tree is estimated. Each tree is midpoint rooted, and the ancestral sequences are reconstructed in the root node of each tree. In (5), the rest of the unassigned sequences in *A*, are then aligned to the ancestral sequences of each cluster and the shortest Jukes-Cantor distance to each ancestral sequence is calculated. If the shortest distance from the unassigned sequence to any of the ancestral sequences is larger than the average distance between clusters, then the unassigned sequence is saved for the next iteration. If the shortest distance to any of the ancestral sequences is less than or equal to the average distance between clusters, the sequence is assigned to the cluster with the shortest distance from its ancestral sequence. The process is iterated until all sequences are assigned to a cluster. In this specific example, $r = 16$ and $b = 4$

Notice that, when aligning the *r* sequences in *B*, a pairwise alignment is first used to define clusters and then a multiple alignment is used within each cluster when estimating phylogenetic trees. While this procedure does save computational time, as multiple alignments are expensive, the main reason for this two-step procedure is that multiple alignments, that include highly divergent sequences, can negatively affect alignment accuracy even among more similar sequences in the alignment. Initially, generating a large combined multiple alignment for all sequences in *B* does, therefore, not lead to as good of a performance when estimating phylogenetic trees within each cluster, as when a multiple alignment is performed separately for each cluster.

We compare AncestralClust to two state-of-the-art clustering methods, UCLUST (Edgar, 2010) and CD-HIT (Fu *et al.*, 2012), and a clustering method specifically developed for divergent sequences, SpClust (Matar *et al.*, 2019). We use a variety of measurements to assess the accuracy and evennness of the clustering. We first calculate two traditional measures of accuracy: purity and normalized mutual information (NMI), used in Bonder *et al.* (2012). Although we acknowledge that inaccuracies in taxonomy exist in public sequence databases (see Nilsson *et al.*, 2006), for the purpose of evaluating performance, we define a taxonomic group as belonging to phylum, class, order, family, genus or species as classified by the National Center for Biotechnology Information (NCBI) taxonomic system. Since taxonomic groups are not comparable across taxonomic levels (i.e. classes compared with genera), we calculate accuracy measures for each taxonomic level separately.

To describe the performance measures, we first need to introduce some notation. w_i is, as previously defined, the set of sequences in cluster *i*. The number of different sequences in cluster *i* is the cardinality of w_i, $|w_i|$.

The purity of clusters, as defined by (Schütze *et al.*, 2008) , is then calculated as:

$$purity(\Omega, C) = \frac{1}{N}\sum_{i=1}^{k}\max_{j}|\omega_i \cap c_j| \quad (1)$$

where $C = \{c_1, c_2, \ldots, c_d\}$ is the partition of the data into *d* taxonomic groups, where c_j is the set of all sequences belonging to taxonomic group *j*, and *N* is the total number of sequences. Notice that purity takes a value in $\{k/N, (k+1)/N, \ldots, 1\}$ and is equal to 1 if there are no clusters that contain more than one taxonomic group. Purity tends to increase as the number of clusters increases. For example, purity becomes 1 when each sequence is assigned to its own cluster.

We next describe the calculation of NMI. First, we define the proportion of sequences in cluster *i* is $q_i = |\omega_i|/N$ and the entropy of the clusters as

$$H(\Omega) = -\sum_{i=1}^{k} q_i log_2(q_i).$$

The proportion of sequences with taxonomic assignment *j* is $p_j = |c_j|/N$. The entropy of the taxonomic groups is defined as

$$H(C) = -\sum_{j=1}^{d} p_j log_2(p_j).$$

We also define the frequency of assignment *j* in cluster *i* as

$$p_{j|i} = |\omega_i \cap c_j|/|\omega_i|,$$

and the conditional entropy as

$$H(C|\Omega) = -\sum_{i=1}^{k} q_i \sum_{j=1}^{d} p_{j|i} log_2(p_{j|i}).$$

NMI is then calculated as

$$NMI(\Omega, C) = \frac{2(H(C) - H(C|\Omega))}{[H(\Omega) + H(C)]} \quad (2)$$

To measure the evenness of the clusters, we use the Coefficient of Variation, which is calculated as:

$$CV = \frac{\sqrt{\sum_{j=1}^{d} (|c_j| - m)^2/(d-1)}}{m} \quad (3)$$

where $m = \frac{\sum_{j=1}^{d} |c_j|}{d}$ is the mean size of the clusters. We also use a taxonomic incompatibility measure to assess the accuracy of the clusters. A taxonomic incompatibility is assigned if two taxonomic groups both exist in two different clusters. One taxonomic group can be split into multiple clusters, but if taxonomic groups are monophyletic, two groups should not both be found in more than one cluster. The total taxonomic incompatibility is then calculated by summing over all species found in the dataset. More precisely, let $S_{\omega_i}(c_l, c_j)$ be an indicator variable that returns one if species c_l and c_j are both found in cluster i (i.e. $|\omega_i \cap c_l||\omega_i \cap c_j| > 0$) and zero otherwise. Then the taxonomic incompatibility is defined as:

$$TI(\Omega, C) = \sum_{l=1}^{d-1} \sum_{j=i+1}^{d} max\{0, \sum_{i=1}^{k} S_{\omega_i}(c_l, c_j) - 1\}. \quad (4)$$

Notice that, this measure does not penalize paraphyletic clusters but only penalizes clustering that is strictly incompatible with a phylogentic tree, i.e. polyphyletic clusters.

All three measures (purity, NMI and taxonomic incompatibility) are very sensitive to both the number of clusters and the variance in cluster size. For example, if all sequences are assigned to different clusters or if all sequences are assigned to the same cluster, the taxonomic incompatibility is, by definition, zero. In general, taxonomic incompatibility has the potential to be highest, and NMI has the potential to be lowest, when there is an intermediate amount of clusters of equal size. With high variance in cluster size there is less potential for generating clusters with high taxonomic incompatibility of low NMI. To address these issues and to allow fair comparison when numbers of clusters and variance in cluster sizes vary, we calculated the *relative purity*, *relative NMI* and *relative incompatibility*. We calculate these measures by scaling them relative to their expected values under random assignments given the number of clusters and the cluster sizes. We estimate *relative NMI* by dividing the raw NMI score by the average NMI of 10 clusterings, in which sequences have been assigned at random with equal probability to clusters, such that the cluster sizes are the same as the cluster sizes produced in the original clustering. We use the same procedure to convert the purity measure into *relative purity* and the taxonomic incompatibility measure into *relative incompatibility*.

3 Results

To assess performance of these clustering methods on random samples of divergent nucleotide sequences, we used 100 random samples of 10 000 sequences from three metabarcode reference databases [16, 18S and Cytochrome Oxidase I (COI)] from the CALeDNA project Curd *et al.* (2019). We were unable to perform clustering of these datasets using SpClust because the program did not complete in a feasible amount of time (we allowed for a week of computational time to complete the clusterings of 100 random samples for each method). In the main figures, we display the results against UCLUST, but because of the high number of clusters and high Coefficient of Variation of cluster sizes from CD-HIT (see Supplementary Figs S8, S15 and S16), we display the CD-HIT results in the Supplementary Material. For CD-HIT, we used the lowest possible similarity threshold, which is 80%, to attempt to create clusters of similar sizes to AncestralClust.

We used the COI database to explore the relationships between taxonomic incompatibility and the number of clusters (Supplementary Fig. S1), taxonomic incompatibility and the Coefficient of Variation (Supplementary Fig. S2) and taxonomic incompatibility and both the number of clusters and the Coefficient of Variation (Supplementary Fig. S3). Notice that, taxonomic incompatibility increases as the number of clusters increases and taxonomic incompatiblity decreases as the Coefficient of Variation decreases. Also, the relationships between *relative incompatibility* and the number of clusters (Supplementary Fig. S4), *relative incompatibility* and the Coefficient of Variation (Supplementary Fig. S5) and *relative incompatibility* and both the number of clusters and the Coefficient of Variation (Supplementary Fig. S6) similarly show the same increasing or decreasing trends for species, genus and family taxonomic levels. This shows that comparisons of methods cannot be done fairly without also referencing differences in number of clusters and variance in cluster sizes inferred by the different methods.

We compared AncestralClust against UCLUST using *relative NMI* and the Coefficient of Variation for species (Fig. 2), genus, family, order, class and phylum levels (Supplementary Fig. S7) for the 16S, 18S and COI metabarcoding datasets. We used $r = 750$

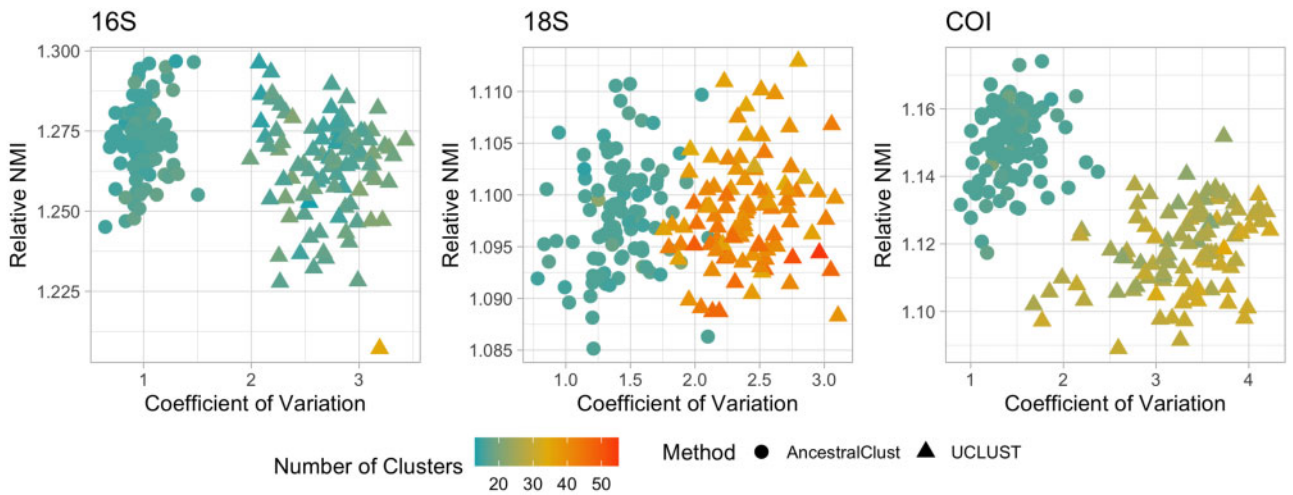

Fig. 2. *Relative NMI* at the species level against Coefficient of Variation for AncestralClust and UCLUST for 100 samples of 10 000 randomly chosen 16S, 18S and COI reference sequences from the CALeDNA Project (Curd *et al.*, 2019). The similarity threshold for UCLUST was 0.58. For AncestralClust, we used 750 initial random sequences with 15 initial clusters

random initial sequences, which is 7.5% of the total number of sequences in each sample, and $b = 15$ initial clusters. The choice of r and b is described in Supplementary Appendix A. Results for CD-HIT (Supplementary Fig. S8) show that CD-HIT creates hundreds of clusters for every barcode with a high Coefficient of Variation and tends to have a lower *relative NMI* than AncestralClust. Notice in Figure 2 that, *relative NMI* tends to be higher with a lower coefficient of variation for AncestralClust across all barcodes. This suggests, that for these divergent eDNA sequences, AncestralClust provides clusterings that are more even in size and that are more consistent with conventional taxonomic assignment. We also measured *relative purity* and *relative incompatibility* and Coefficient of Variation using AncestralClust, UCLUST and CD-HIT for the same datasets under the same running conditions. Notice in Figures 3 and 4, AncestralClust tends to create balanced clusters with higher *relative purities* and lower *relative taxonomic incompatibilities* compared to UCLUST at all taxonomic levels. For *relative incompatibility* for metabarcode 16S (Supplementary Fig. S9), AncestralClust performs noticeably better than UCLUST at the species and genus levels but at the family, order, class and phylum levels it has either the same or slightly more taxonomic incompatibility, but with substantially lower Coefficient of Variation of the cluster sizes. Also, at the species, genus and family levels, there is a clear negative correlation between UCLUST *relative incompatibility* and Coefficient of Variation. This illustrates the observation that clusterings with a higher variance in cluster sizes tend to generate lower taxonomic incompatibility. For *relative purity* for 16S, AncestralClust has noticeably higher *relative purities* than UCLUST at every taxonomic level (Supplementary Fig. S10).

At the order, class and phylum levels for metabarcode 18S AncestralClust tends to have less *relative incompatibility* (Supplementary Fig. S11) but not at the species, genus or family level. For *relative purity* of 18S, AncestralClust shows higher *relative purities* than UCLUST at the family, order, class and phylum levels but lower *relative purities* at the species and genus levels (Supplementary Fig. S12). The reason for this difference in relative performance between low and high taxonomic levels is that when d, the number of taxonomic groups, approaches N, the total number of sequences in a sample, the performance measured become increasingly sensitive to the value of k. As defined by Equation 1, when $d = N$, purity takes the value k/N. So at lower taxonomic levels with large values of d, methods that defines a high number of clusters (large value of k) will tend to have higher purity (Equation 1). The same effect is observed for taxonomic incompatibility (Equation 4). When $d = N$ taxonomic incompatibility is 0. Thus, as d approaches N, taxonomic incompatibility becomes increasingly sensitive to values of k. This sensitivity to k is observed in the raw values of purity (Supplementary Fig. S13) and incompatibility (Supplementary Fig. S14) where the average number of species, genera and families in a sample is high (8369.1, 5129.9 and 2524.2, respectively), while the average number of phyla in a sample is low (50.4). 16S and COI have fewer taxonomic groups than 18S at every taxonomic level and thus are less sensitive to k.

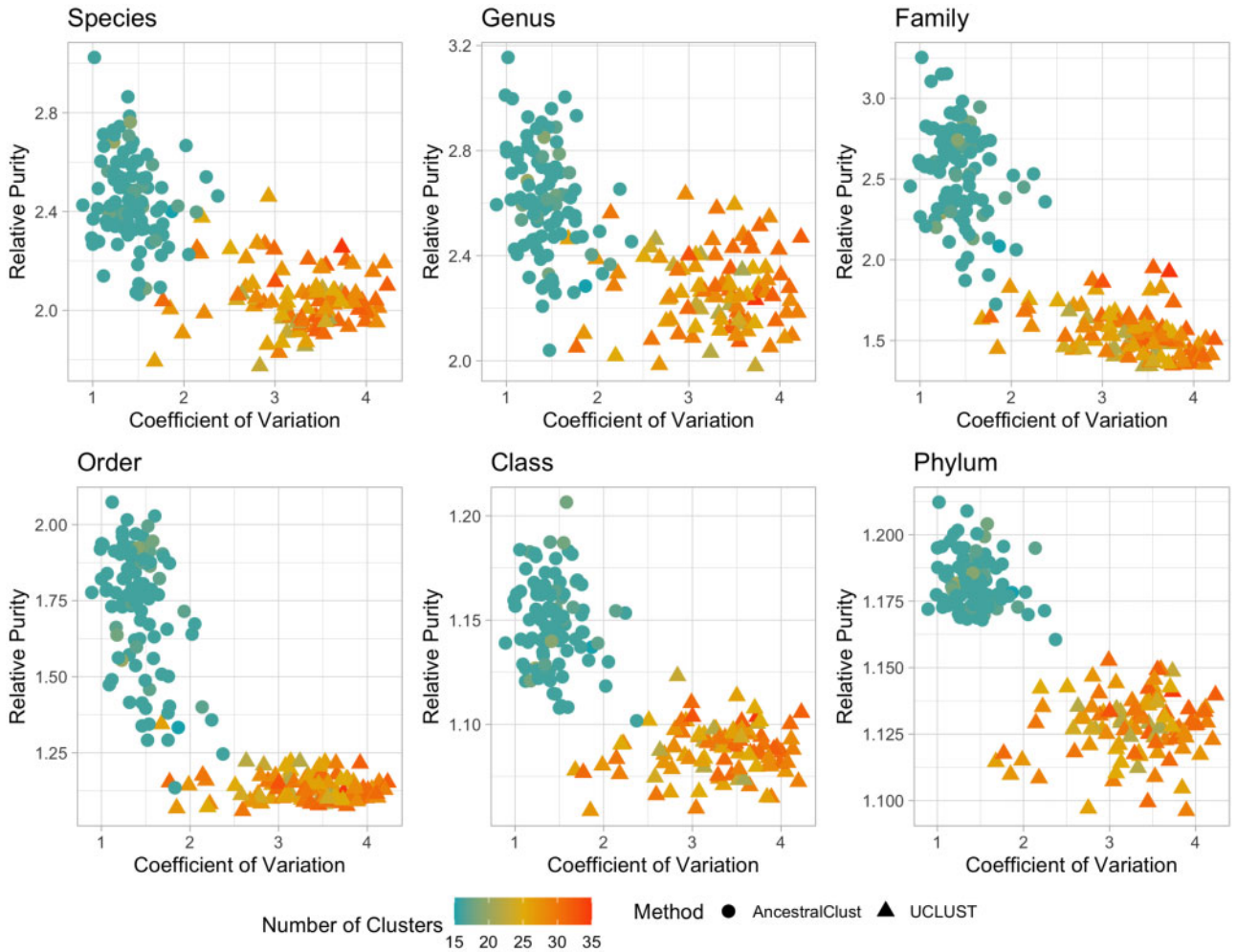

Fig. 3. *Relative purity* against coefficient of variation for AncestralClust and UCLUST for 100 samples of 10 000 randomly chosen COI reference sequences. COI reference sequences are from the CALeDNA Project (Curd *et al.*, 2019). The similarity threshold for UCLUST was 0.58. For AncestralClust, we used 750 initial random sequences with 15 initial clusters

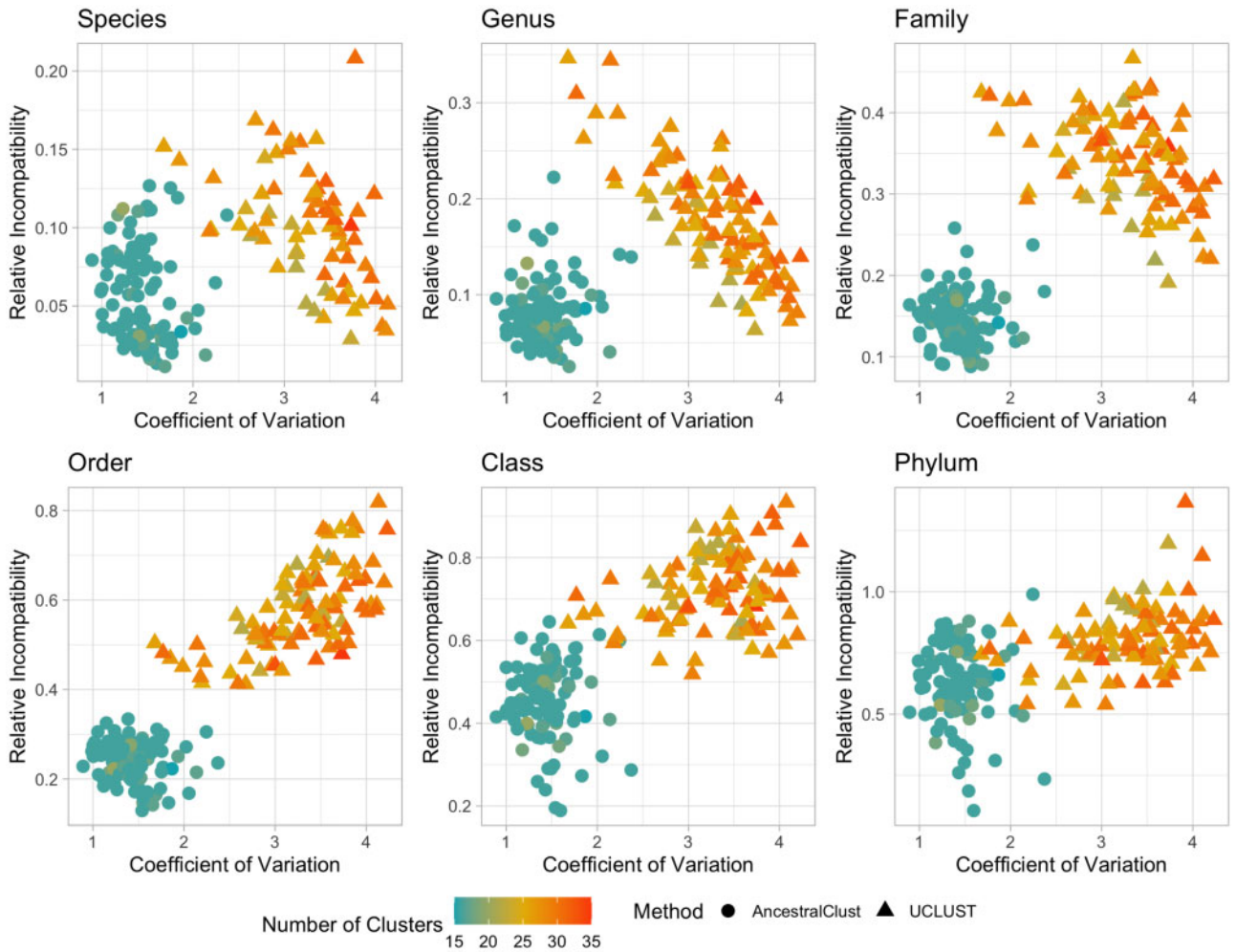

Fig. 4. Relative incompatibility against coefficient of variation for AncestralClust and UCLUST for 100 samples of 10 000 randomly chosen COI reference sequences. COI reference sequences are from the CALeDNA Project (Curd *et al.*, 2019). The similarity threshold for UCLUST was 0.58. For AncestralClust, we used 750 initial random sequences with 15 initial clusters

For CD-HIT, *relative purities* of 16S, 18S and COI tends to be similar or lower than AncestralClust at every taxonomic level (Supplementary Fig. S15). In addition, *relative incompatibility* of 16S and 18S tends to be similar or higher than AncestralClust at every taxonomic level (Supplementary Fig. S16). For COI, *relative incompatibility* tends to be higher or similar at species, genus and family levels, but lower at order, class and phylum levels, but with substantially higher Coefficient of Variation in cluster size.

Next, we analyzed two datasets with different properties: one dataset of divergent species from the same gene and another dataset of six paralogous genes from species of the same phylum. In the first dataset, we expect the sequences to cluster according to species. In the second dataset, we expect the sequences to cluster according to genes. The first dataset contained 13 043 sequences from the COI CaleDNA database from 11 divergent species that were from seven different phyla and 11 different classes and the second dataset contained sequences from six different genes from taxonomically

Table 1. Comparisons of clustering methods using 13 043 COI sequences from 11 different species

Method	No. of clusters	Time (s)	Mem (MB)	*Relative purity* (species)	*Relative incompat.* (species)	*Relative NMI* (species)	Coeff. of Var.
AncestralClust	11	293.2	19.3	3.6271	0	551.09	0.8574
UCLUST	11	<1	9.9	3.1204	0.0182	474.63	0.8300
CD-HIT	24	5.86	43.9	3.6180	0	241.66	1.2031
SpClust (fast)	1	152046.5	2678.9	1	0	1	—
SpClust (moderate)	1	188172.9	6457.6	1	0	1	—
SpClust (maxPrecision)	1	189577.1	6452.5	1	0	1	—

Note: *The* list of species can be found in Supplementary Table S1. *Relative purity*, *relative incompatibility* and *relative NMI* were calculated at the taxonomic rank of species. For UCLUST, the identity thresholds were chosen to force the expected 11 number of clusters. For CD-HIT, the lowest possible identity was chosen which is 0.8. In the case of SpClust, Coefficient of Variation cannot be calculated for one cluster. SpClust clusters were created with version 2.

Table 2. Comparisons of clustering methods using 39 sequences from six paralogous genes from Matar *et al.* (2019)

Method	No. of clusters	Time (s)	Memory (MB)	*Relative purity*	*Relative NMI*	Coeff. of Var.
AncestralClust	6	370.3	412.0	1.7619	1.8660	0.3982
UCLUST (id = 0.4)	6	1	15.4	1.3810	1.5667	0.5396
UCLUST (id = 0.6)	19	1	20.1	2.1538	1.4379	0.7166
UCLUST (id = 0.8)	29	1.9	20.4	1.6923	1.1717	0.4565
CD-HIT (id = 0.8)	31	0.48	39.9	1.2308	1.0950	0.4574
SpClust (fast)	4	44.6	166.2	1.4167	2.2463	0.8432
SpClust (moderate)	4	112.5	166.1	1.6818	2.4335	0.6453
SpClust (max precision)	4	570.1	166.0	1.6818	2.9449	0.6809

Note: 'id' refers to the identity threshold used. We used identity thresholds of 0.4, 0.6 and 0.8 for UCLUST. We used precision levels of fast, moderate and maximum for SpClust using version 1 since version 2 only produced one cluster for all modes.

similar species. First, we compared all methods using 13 043 COI sequences from the 11 different species (Table 1). Ideally, we expect these sequences to form 11 different clusters, each including all the sequences from one species. We chose identity thresholds to enforce the expected number of clusters for each method. We were unable to form 11 clusters using CD-HIT because the program does not allow clustering of sequences with identity thresholds $< 80\%$ at default parameters. For SpClust, we used the three precision modes (fast, moderate and maxPrecision) available for the method. In this analysis, AncestralClust achieved a perfect clustering (the raw purity was 1 and *relative incompatibility* was 0) and it had the second lowest memory usage, although it was the second slowest. CD-HIT also had a raw purity of 1 but formed more than twice the number of clusters than expected. UCLUST was one of the fastest methods and used the least amount of memory but had the second lowest *relative purity* with the third highest *relative NMI* values. SpClust only identified one cluster, with a computational time of $\sim$2 days. In comparison, AncestralClust took $\sim$5 minutes and UCLUST used $<$ 1 s.

Next, we analyzed 'genomic set 1' from Matar *et al.* (2019), which consists of 39 sequences from six homologous genes (FCER1G, S100A1, S100A6, S100A8, S100A12 and SH3BGRL3 in Table 2). We expect these sequences to form six clusters. We varied the identity thresholds for UCLUST using thresholds 0.4, 0.6 and 0.8. For CD-HIT, we used the lowest identity threshold available on default parameters which is 0.8. Since this dataset contained six different genes, we calculated *relative NMI* and *relative purity* using genes as the categories instead of taxonomy, and did not use taxonomic incompatibility as an accuracy measure. Only AncestralClust and UCLUST produced the expected number of clusters, and among the methods and parameters that created the expected number of clusters, AncestralClust had the highest *relative purity* value. AncestralClust was the second slowest method and had the highest memory requirements due to the wavefront algorithm alignment, which is $\mathcal{O}(ns)$ in running time and $\mathcal{O}(s^2)$ in memory requirements, where n is the read length and s is the alignment score. Since alignments were performed using six different genes that were longer than 1.5 kb (the average sequence length was 2387.9 bp and the longest sequence was 5363 bp), this resulted in high values of n and s. SpClust had the highest *relative NMI* but lower *relative purity* than AncestralClust for all precision modes, however, it failed to produce the expected number of clusters and found fewer clusters with a higher Coefficient of Variation than AncestralClust, making the results difficult to compare.

We measured the time and memory requirements of AncestralClust using datasets containing 100, 1000, 10 000 and 100 000 sequences from both the 16S and the COI reference database for 1 and 8 CPUs (Supplementary Fig. S23). We created 'low' divergence datasets by randomly choosing one sequence from the database and selecting all of its nearest sequences based on taxonomy information. We also created 'high' divergence datasets which were created by randomly choosing sequences from the database that are from different phyla. We used the wavefront algorithm to investigate whether there was a substantial increase of memory requirements with the 'high' divergence dataset given that the alignment algorithm is quadratic with respect to the alignment score. Unsurprisingly, the 'high' divergence datasets had the longest running time (Supplementary Fig. S23A) and consumed the most memory (Supplementary Fig. S23B). However, there were not substantially large differences in running times and memory usage between 'high' and 'low' divergent datasets. In addition, the run time was significantly reduced by using more CPUs. The use of more CPUs also did not substantially increase the memory requirements.

4 Conclusions

We developed a phylogenetic-based clustering method, AncestralClust, specifically to cluster divergent metabarcode sequences. We performed a comparative study between AncestralClust and widely used clustering programs UCLUST and CD-HIT, and for divergent sequences, SpClust. UCLUST is substantially faster than AncestralClust and should be the preferred method if computational speed is the main concern (i.e. quick clustering of a large amount of raw sequencing reads from next-generation sequencing technologies for error correction). However, AncestralClust tends to form clusters of more even size with lower *relative taxonomic incompatibility* and higher *relative NMI* and *relative purity* than other methods, for the relatively divergent sequences analyzed here. We recommend the use of AncestralClust when sequences are divergent, especially if a relatively even clustering is also desirable, for example for various divide-and-conquer approaches where computational speed of downstream analyses increases faster than linearly with cluster size.

Funding

This work used the Extreme Science and Engineering Discovery Environment (XSEDE) Bridges system at the Pittsburgh Supercomputing Center through allocation BIO180028 and was supported by National Institutes of Health R01GM138634-01.

Data availability

AncestralClust is available at https://github.com/lpipes/AncestralClust and the data underlying this manuscript are available at https://doi.org/10.5281/zenodo.5602364.

Conflict of Interest: none declared.

References

Balaban,M. *et al.* (2019) Treecluster: clustering biological sequences using phylogenetic trees. *PLoS One*, **14**, e0221068.

Bonder,M.J. *et al.* (2012) Comparing clustering and pre-processing in taxonomy analysis. *Bioinformatics*, **28**, 2891–2897.

Chen,Q. *et al.* (2018) Comparative analysis of sequence clustering methods for deduplication of biological databases. *J. Data Inf. Qual.*, **9**, 1–27.

Curd,E.E. *et al.* (2019) Anacapa toolkit: an environmental DNA toolkit for processing multilocus metabarcode datasets. *Methods Ecol. Evol.*, **10**, 1469–1475.

Edgar,R.C. (2010) Search and clustering orders of magnitude faster than blast. *Bioinformatics*, **26**, 2460–2461.

Farris,J.S. (1972) Estimating phylogenetic trees from distance matrices. *Am. Nat.*, **106**, 645–668.

Fu,L. *et al.* (2012) Cd-hit: accelerated for clustering the next-generation sequencing data. *Bioinformatics*, **28**, 3150–3152.

Ghodsi,M. *et al.* (2011) Dnaclust: accurate and efficient clustering of phylogenetic marker genes. *BMC Bioinformatics*, **12**, 1–11.

Huang,Y. *et al.* (2010) CD-HIT Suite: a web server for clustering and comparing biological sequences. *Bioinformatics*, **26**, 680–682.

Jukes,T.H. *et al.* (1969) Evolution of protein molecules. *Mammalian Protein Metab.*, **3**, 21–132.

Lassmann,T. (2020) Kalign 3: multiple sequence alignment of large datasets. *Bioinformatics*, **36**, 1928-1929.

Li,W. *et al.* (2001) Clustering of highly homologous sequences to reduce the size of large protein databases. *Bioinformatics*, **17**, 282–283.

Marco-Sola,S. *et al.* (2020) Fast gap-affine pairwise alignment using the wavefront algorithm. *Bioinformatics*, 1–8.

Matar,J. *et al.* (2019) Spclust: towards a fast and reliable clustering for potentially divergent biological sequences. *Comput. Biol. Med.*, **114**, 103439.

Needleman,S.B. and Wunsch,C.D. (1970) A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J. Mol. Biol.*, **48**, 443–453.

Nilsson,R.H. *et al.* (2006) Taxonomic reliability of DNA sequences in public sequence databases: a fungal perspective. *PLoS One*, **1**, e59.

Ratnasingham,S. and Hebert,P.D. (2007) Bold: the barcode of life data system (http://www. barcodinglife. org). *Mol. Ecol. Notes*, **7**, 355–364.

Rusch,D.B. *et al.* (2007) The sorcerer ii global ocean sampling expedition: northwest Atlantic through eastern tropical pacific. *PLoS Biol.*, **5**, e77.

Saitou,N. and Nei,M. (1987) The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Mol. Biol. Evol.*, **4**, 406–425.

Schoch,C.L. *et al.* (2020) NCBI taxonomy: a comprehensive update on curation, resources and tools. *Database*, **2020**, baaa062.

Schütze,H. *et al.* Introduction to Information Retrieval. Cambridge University Press, Cambridge, UK, 2008.

Stamatakis,A. (2014) Raxml version 8: a tool for phylogenetic analysis and post-analysis of large phylogenies. *Bioinformatics*, **30**, 1312–1313.

Yang,Z. (2014) *Molecular Evolution: A Statistical Approach*. Oxford University Press, Oxford, UK.

Zheng,W. *et al.* (2019) A parallel computational framework for ultra-large-scale sequence clustering analysis. *Bioinformatics*, **35**, 380–388.

Zou,Q. *et al.* (2018) Sequence clustering in bioinformatics: an empirical study. *Brief. Bioinformatics*, **21**, 1–10.