MNHN-Tree-Tools: Supplement

1 Introduction

MNHN-Tree-Tools is a new suite of tools that allow us to cluster sequences and build phylogenetic trees from datasets containing nucleic and experimentally, amino acid sequences. MNHN-Tree-Tools are based on an iterative, adaptive version of the *density-based algorithm for discovering clusters in large spatial databases with noise* (DBSCAN) [4] algorithm and are further optimized to take advantage of Single Instruction Multiple Data (SIMD) instructions, Graphics Processing Units (GPU)s, multi-threading and multi-node High Performance Computing (HPC) architectures. Working around and adapting the well known DBSCAN clustering algorithm allows us to gain insights into the evolution of sequences and to build phylogenetic dependency graphs ultimately leading to trees.

In this document we provide a detailed overview of the algorithm. We present how we evaluate the performances of our tool using two available annotated datasets: α satellites in humans [19] and the Tree of Life (ToL) from the SILVA project which uses 16S/18S ribosomal RNA as an Operational Taxonomic Unit (OTU) [14]. Finally we provide further insights into the introduced methods by applying them on computer generated sequences resulting from virtual evolution simulations.

2 Algorithm

2.1 Motivation and aims

Many applications in natural history require a researcher to group related sequences into "families". In order to do this we first have to provide a clear definition of a sequence family and its representation. Afterwards we outline how we can automatically identify a sequence family in an algorithmic fashion.

We first define a sequence space $\mathbb{K}_S$ to be an finite dimensional field. We then define a distance measure between two sequences d as a function that yields only positive real results. Hence,

$$d : \mathbb{K}_S \times \mathbb{K}_S \to \mathbb{R}^+. \tag{1}$$

We propose that a family of sequences, which arises from an evolutionary process is represented by a density peak in such a sequence space, a confined region $R(\rho)$ where local sequence distances are smaller than on the surface $S(R(\rho))$. ρ being the density limit for R to form a connected region, and thus the density on the surface:

$$\forall r \in S(R(\rho)) : \rho(r) = \rho, \tag{2}$$

and within the regions:

$$\forall r \in R(\rho) : \rho(r) \geq \rho, \tag{3}$$

where $r \in \mathbb{K}_S$ represents a sequence in our sequence space. The DBSCAN algorithm can find ensembles for a given density defined by the number of sequences to be found in a neighbourhood defined by the radius ϵ. With the

volume V of the ϵ-ball:

$$\rho(\text{minpts}, \epsilon) = \frac{\text{minpts}}{V(\epsilon)}, \tag{4}$$

dependent on the measure d in equation (1) DBSCAN finds connected regions, clusters of sequences, $R_1(\rho) \dots R_n(\rho)$ with a density $\geq \rho(\text{minpts}, \epsilon)$. As dense regions shall be embedded into less dense regions we further sketch that for $l, m \leq n$ and for an arbitrary set of chosen found regions $R(\rho)_l \dots R(\rho)_m$ there exists a radius $\epsilon' > \epsilon$ and hence, a density $\rho'(\text{minpts}, \epsilon') < \rho(\text{minpts}, \epsilon)$ so that $R(\rho)_l \dots R(\rho)_m \in R'(\rho')$, with $R'(\rho')$ being a region, cluster of sequences, found by DBSCAN with input parameters minpts and ϵ'. Reasoning from the above shows that clusters can only be discovered, grown or merged as the density in equation (4) is decreased while the ϵ is increased. From the sketch outlined it directly follows that for two discovered regions $R_1(\rho) = \{r_1, \dots, r_k\}$ and $R_2(\rho) = \{r_u, \dots, r_v\}$, holding sequences r_i only the following outcomes, at decreased density ρ' and hence increased ϵ', compared to a prior situation can exist:

1. A new region $R_3(\rho')$ is formed and the regions $R_1(\rho)$ and $R_2(\rho)$ merge and hence:

   $$R_3(\rho') = [R_1(\rho) \cup R_2(\rho)] \cup R_+(\rho'), \tag{5}$$

   where R_+ has either no elements or hold elements not found in R_1 or R_2.

2. The regions expand:

   $$\begin{aligned} R_{1a}(\rho') &= R_1(\rho) \cup R_{1+}(\rho'), \\ R_{2a}(\rho') &= R_2(\rho) \cup R_{2+}(\rho'), \end{aligned} \tag{6}$$

   with R_{1+} and R_{2+} holding either no sequences or sequences that are were not found in R_1 or R_2.

Further it is clear that, at such a decrease in density, new regions independent from R_1 and R_2 can be discovered.

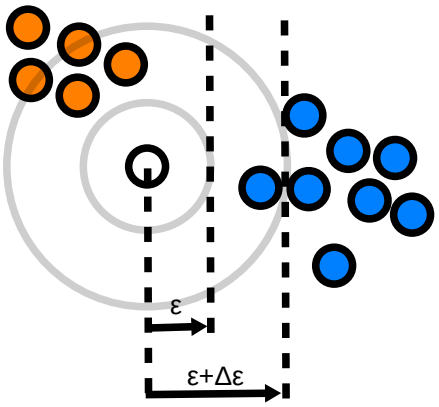

Figure 1: **Sequence families in the sequence space are recovered by the DBSCAN algorithm** As ϵ increases by $\Delta\epsilon$ the density that is required to form a cluster decreases and the blue and orange cluster are merged.

A tree of related sequences can be built performing DBSCAN clustering runs for different density values and by merging related clusters as shown in figure 1. Such a tree can be used to guide phylogenetics as we will discuss later in section 2.2. The rational behind figure 1 is that sequence families form dense ensembles in sequence space as outlined by the blue and orange sequence families. A DBSCAN run with a well chosen ϵ can, together with an appropriate minpts value, distinguish both families and attribute them to separate clusters. Performing a successive DBSCAN run with an increased epsilon $\epsilon + \Delta\epsilon$ value captures less dense connected ensembles at a same minpts value and therefore, assuming that $\Delta\epsilon$ is large enough, both the orange and blue families are attributed to a single family. Hence, we can build a phylogenetic tree from density connected clusters found by successive DBSCAN runs with increased ϵ parameters and a cluster comparison step as outlined in figure 4.

We illustrated the way our algorithm captures and builds trees in figure 2. In the left image of figure 2 a typical sequence space encountered by our algorithm is shown. Highlighted are three dense areas, orange, purple and green regions that are representing high sequence count and conservation. The purple and green regions are embedded into a less dense blue region. Further a disconnected red region of about the same density as the blue region is shown. In the right image the algorithm starts at a small ϵ and will at first detect the very dense green, purple and orange clusters representing the rightmost leaves of our tree. As in the following steps ϵ increases, less dense regions are discovered and the blue and red families, that incorporate the previous detected clusters, are found. The green and purple clusters are embedded into the blue cluster and tree branches are formed accordingly. In the left section of the right image finally a single cluster representing the whole dataset is shown. This is the root of the tree.

2.2 Evolutionary families and statistical clusters

For improved clarity we introduce the distinction between *evolutionary families* and *statistical clusters*:

- **Evolutionary family**: Represents an ensemble of proximal sequences that have emerged from an evolutionary process. Hence, sequences that are descending from a previously amplified (multiplied) sequence, and are mutations (a diffusion) of thereof. As such we can further define subfamilies which correspond to amplifications of such mutations and the mutations of such amplifications. The chain of these families yields a hierarchical structure of families and subfamilies. These amplification events are hard to grasp or due to the high mutation rate over a long past and because of the underlying rise in entropy not statistically accessible.
- **Statistical cluster**: Is an ensemble of sequences that cluster together for a given density. Herein we use a density and monotony based description. A cluster is the ensemble of sequences around a local density maximum in sequence space. All sequences around this maximum shall belong to the same cluster as long as the density surrounding this point is either monotonically decreasing or stable. Statistical clusters can as such be captured using the DBSCAN algorithm.

As such, a relationship between statistical clusters found within a dataset and evolutionary families does not hold for all imaginable evolutionary families. We nevertheless stipulate, and show using simulations, that the statistical clusters found, have an inherent relationship to amplification-mutation (multiplication-diffusion) based models of evolution such as the Jukes and Cantor 1969 (JC69) model [10] or derived more recent models [5, 8, 12, 17, 18]. The capture of evolutionary families in statistical clusters is further discussed in figure 3.

2.3 Distance measures between sequences

We implemented our algorithm using two different distance measures, d in equation (1). The first one is a distance measure based on the k-mer representation of the sequences followed by principal component analysis (PCA). [2]. The second one is a custom implementation of the Smith Waterman distance [15].

2.3.1 K-mer based distance

To compute the k-mer based distance we first transform the input sequences into frequen-

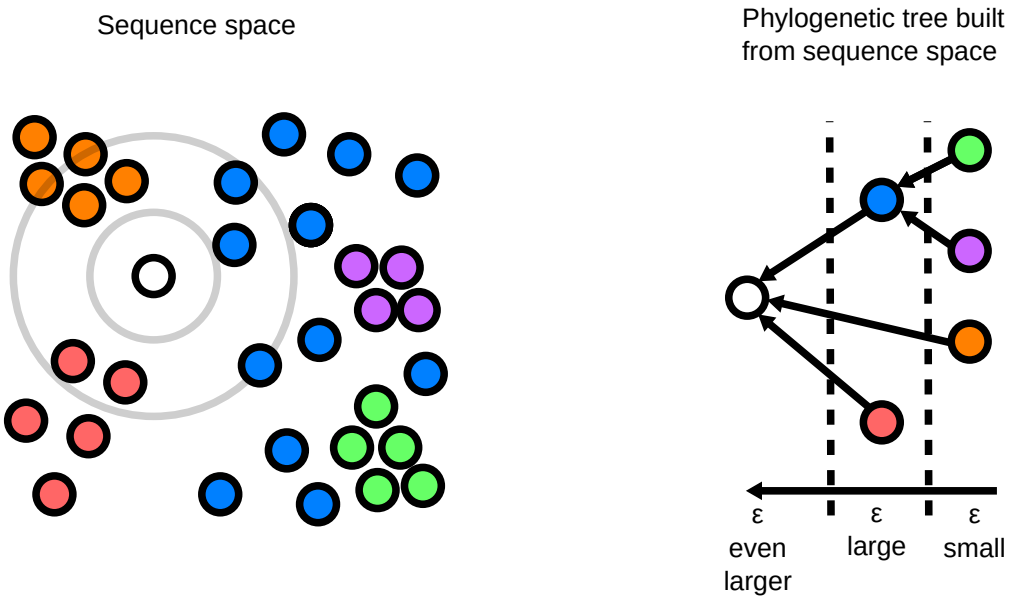

Figure 2: **An illustration motivating our algorithm:** Left: An ensemble of sequences forming clusters at various densities. Orange, green and purple are found at high densities whereas the blue and red clusters are found at lower densities. The blue cluster encompasses the green and purple points. Right: A sequence tree built from the correspondences between clusters found at various density values.

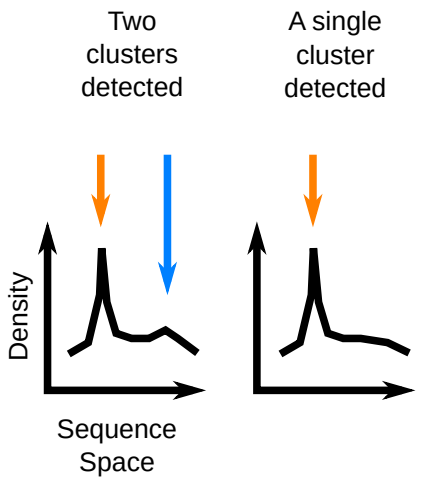

Figure 3: The discernibly of statistical sequence clusters in sequence space $\mathbb{K}_S$: The figure on the left contains two sequence clusters while the figure on the right contains only a single cluster of sequences as no further surface $S(R(\rho))$ can be found that is able to clearly delimit R for the a second peak.

cies of occurring sub-sequences of size k and perform PCA on the vectors obtained. The distance is then evaluated by calculating the L_2 distance in a reduced subspace spun by the principal components. In order for the DBSCAN algorithm to perform in an efficient manner we suggest to project k-mer frequency vectors onto the first 7 principal components, though the researcher can adapt this value to his own information capture / curse of dimensionality tradeoff. For completeness direct L_1 and L_2 k-mer based distances without PCA were also implemented.

Calculations operating on the PCA subspace are the fastest due to the reduced dimensionality. These calculations further offer the advantage of a PCA based feature selection. This is outlined by the application of our algorithm on simulated datasets in section 4.1.

2.3.2 Smith Waterman distance

Besides the above outlined k-mer and PCA based distance we have implemented a Smith Waterman based distance measure [15]. The algorithm was implemented in a straight forward fashion as follows: We compute the Smith-Waterman Matrix [15] by initializing the first row and first column with zeros and apply the following recurrence relation:

$$\begin{aligned} c_1 &= M(j-1, i-1) \\ &+ \text{xch}(A[i], B[j]), \\ c_2 &= M(j, i-1) - 3, \\ c_3 &= M(j-1, i) - 3, \\ M(i,j) &= \max(c_1, c_2, c_3), \end{aligned} \tag{7}$$

where $M(i,j)$ represents the Smith-Waterman matrix, $A[i]$ and $B[i]$ the nucleotide of sequence A and B at position i respectively. *xch*, the nucleotide exchange penalty, yields 4 if the $A[i], B[j]$ hold the same nucleotide base and -4 otherwise. We evaluate a theoretical maximum F of the matrix by taking the longer sequences of A and B and multiplying the length by four:

$$F = 4\text{maxlength}(A, B), \tag{8}$$

and define the distance between the sequences A and B to be:

$$D(A, B) = F - \max(M(i, j)). \tag{9}$$

$D(A, B)$ resolves to 0 if the sequences A and B are identical and to a positive distance otherwise. As the evaluation of the Smith Waterman distance is computationally expensive we implemented it in OpenCL [16] allowing for execution on GPUs. Our algorithm can further make use of the Message Passing Interface (MPI) library [6] to distribute the workloads across high performance computing (HPC) cluster nodes.

2.4 Algorithm details

With the theory at hand we implemented an algorithm that allows us to detect statistical clusters of various densities. In varying the density ρ by modifying the ϵ parameter of the DBSCAN algorithm and comparing clusters obtained from such different runs we implemented a tree building algorithm as outlined in figure 4. As pointed out by the figure the algorithm requires three input parameters. ϵ the initial epsilon neighborhood radius for the DBSCAN algorithm, $\Delta\epsilon$ the increase of ϵ in every step and minpts the minimal number of points to be found in an epsilon neigbourhood to either extend or create a cluster. n in figure 4 represents the number of clusters found by the current DBSCAN run, n(prev) the number of clusters from the previous run and t the step number and hence, how often DBSCAN has been run. The algorithm starts with a small intial ϵ value in order to find very dense clusters of nucleic sequences and hence, clusters that are highly conserved in sequence. In increasing the ϵ of DBSCAN in successive runs, the clusters are built from less and less conserved sequences, and basically fusion from layer to layer, until DBSCAN just detects a single cluster, the root of the tree.

2.5 Tree Building and Cluster Comparison

As outlined above, our procedure to build a tree from a set of sequences relies on three parameters. We need to specify the values for ϵ_o, the smallest *radius* defining the neighborhood that the DBSCAN algorithm uses to search for neighboring points, $\Delta\epsilon$, the increment to this *radius* from one step to the next, and minpts, the minimum number of points that has to reside inside an epsilon neighborhood for cluster formation or expansion.

Going back to figure 3, it is straightforward to see that for the distribution on the left there exists an ϵ, minpts combination so that both families form their own clusters in a DBSCAN clustering approach. It is further noticeable that by increasing ϵ such that $\epsilon_1 > \epsilon$ that the density defined by our ϵ_1, minpts couple has decreased to a point where the DBSCAN algorithm will only detect a single cluster containing both clusters of the prior ϵ, minpts couple.

With this knowledge at hand, our algorithm starts at ϵ_0, stores the clusters, increases ϵ_0 by $i\Delta\epsilon$ until it finds less clusters than in the previous step. In this way obtain an ensemble of ϵ_i for $i = [0, n]$ such that the number of clusters for ϵ_i is bigger than the number of clusters for ϵ_{i+1} until at ϵ_n the DBSCAN algorithm only finds a single cluster. In order to build our tree now we see all these clusters as nodes, and connect a node of a layer of clusters corresponding to ϵ_{i+1} to a node of a layer of clusters ϵ_i if it the node at ϵ_{i+1} contains at least 80% of the sequences of the node at ϵ_i. This whole iterative algorithm is highlighted in figure 4.

3 Application on experimental datasets

3.1 Description of the datasets

We tried *MNHN-Tree-Tools* on two experimental datasets: α-satellite repeats from the Human Genome (hg38) as annotated by Uralsky et al. [19] and a genetic barcoding dataset that contains the 16S/18S ribosomal RNA sequence for many species from the SILVA ToL project [14].

3.1.1 Human α-satellite sequences

Uralsky et al. have published a set of annotations to the α-satellite sequences found in the human genome (hg38) [19]. These annotations were performed using the PERCON tool [11]. Curious to see how our density

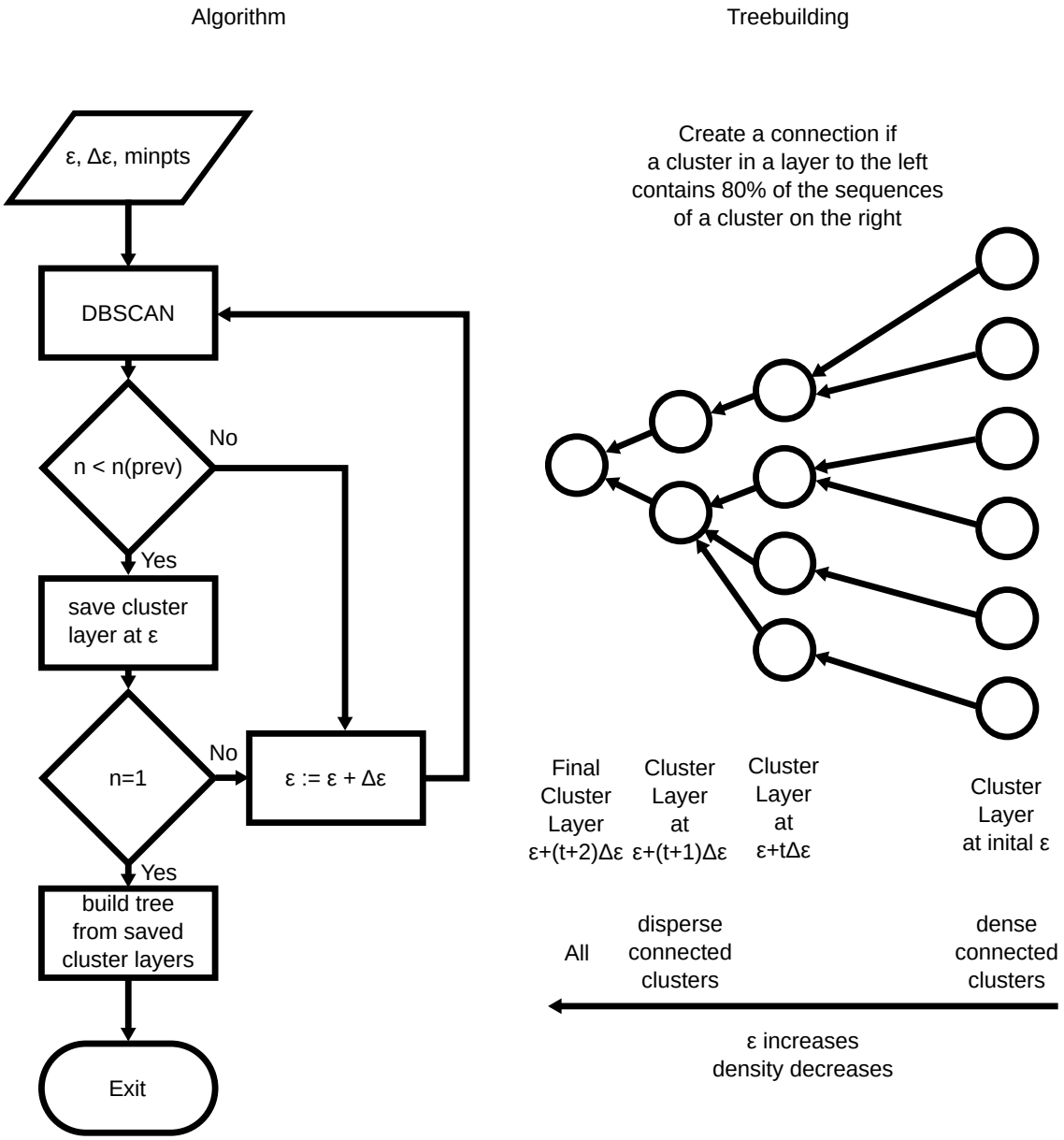

Figure 4: **Algorithmic details:** Left: Algorithm flow chart outlining the different steps of our algorithm. Right: The tree building procedure that links the clusters for different ϵ values together forming a tree.

based approach would classify, detect families in such a dataset we downloaded all annotated sequences. The obtained DNA sequences were transformed into 5-mer frequency vectors. PCA was performed and the 5-mer vectors were projected onto the 7 largest, in terms of associated eigenvalues, principal components. Our tree building algorithm was performed on this 7 dimensional subspace, with the parameters: initial $\epsilon = 0.2$, $\Delta\epsilon = 0.001$ and minpts $= 10$.

MNHN-Tree-Tools yielded a full "phylogenetic" tree that differentiated major well known classifications of α-satellite sequence groups. Figure 5 outlines the full tree obtained as well as colored branches according to original annotations found in the dataset which follow the notation originally proposed by Alexandrov et al. [1].

Most interestingly our results go beyond this initial classification and highlight a fine grained detail of subfamilies, under the herein mentioned definition, that constitute the families under the classic definition.

3.1.2 16S/18S RNA sequences from the Tree of Life

In order to test our algorithm on *barcoding* applications we obtained the SILVA [14] ToL dataset. The dataset contains 16S/18S RNA sequences provided with a detailed annotation that we extracted directly from the provided FASTA file. From these annotations the first six levels were taken into account. We built a tree from these annotations beginning with *Archeae, Bacteria, Eukaryota* on the first level and going down to 13413 different clades found in the sixth and last level that we considered. The resulting tree is shown in figure 6. We used MNHN-Tree-Tools to build a tree from the dataset. Hence, the sequences of SILVA [14] dataset were transformed into k-mer vectors and PCA was performed. The k-mer vectors were projected onto the first 7 principal components and our algorithm has been applied on the resulting data. The parameters used were initial $\epsilon = 0.1$, $\Delta\epsilon = 0.05$ and minpts $= 3$. The resulting tree is highlighted in figure 7.

3.2 Tree quality determination using known partitions

Comparing two partitioned datasets is an inherent difficult task. Several measures such as the F-measure [3], Jaccard [9] or Fowles-Mallows index [7] have been proposed to compare two partitions of a set of points. In our case the system is governed by an increased complexity as we are handling partitions that are incomplete as points below a certain chosen density for a stage in our tree are not attributed to any cluster by the DBSCAN algorithm. An additional difficulty that we face is that a single tree contains numerous partition layers (or levels). We thus need to compare the partitions found for each of these layers with a the given ground truth partition.

In order to evaluate the qualities of our trees we define:

- **Pure clusters**: Clusters that only contain sequences which belong to the a single cluster in the ground truth set.
- **Impure clusters:** Clusters which contain sequences that are a mixture of sequences from different clusters in the ground truth set.

With such a definition we are naturally interested in the following questions:

1. How many pure and impure clusters do we count?
2. Does the total number of detected clusters corresponds to the number of clusters in the ground truth dataset?

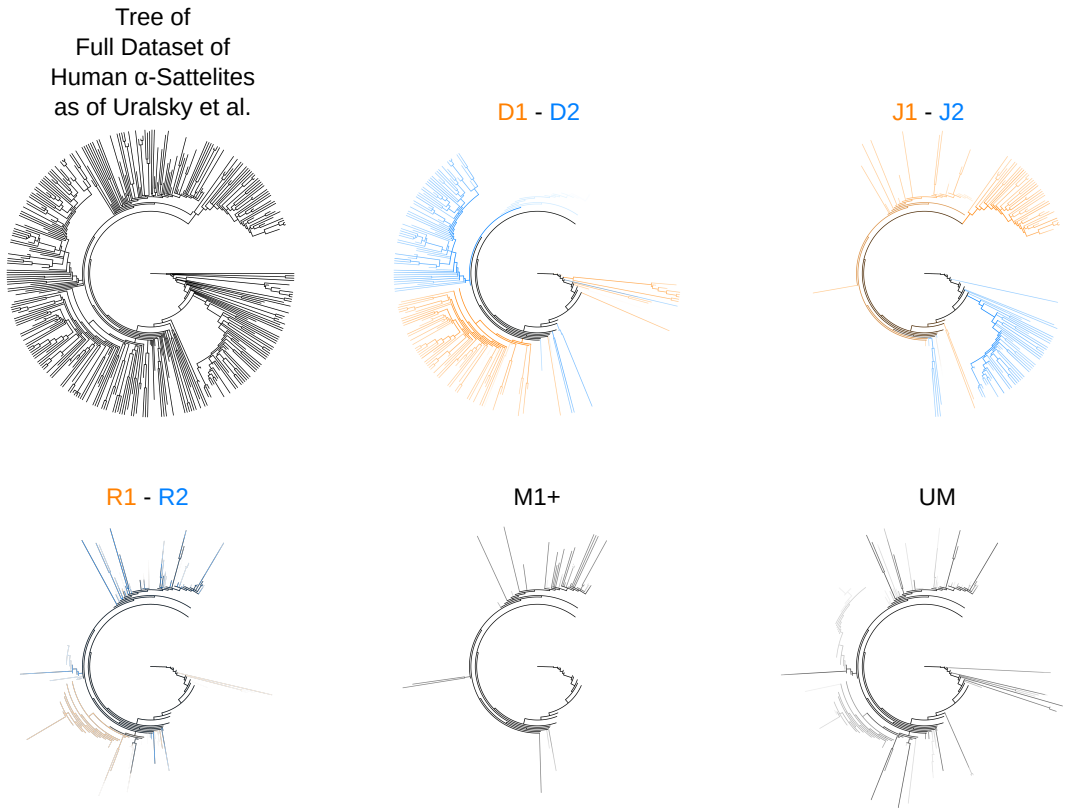

Figure 5: **MNHN-Tree-Tools applied on human α-satellite sequences:** The figure shows six sub-trees. We outline the performance of our algorithm to classify α-satellite sequences found in the human genome as annotated by the track provided by Uralsky et al. [19] and highlight the full tree in the upper row on the left. For the following panels the number of sequences that are grouped in each branch of the tree is indicated by the opacity of each line on a logarithmic scale from white, indicating few sequences, to full opaqueness, indicating all sequences. Additionally we color-annotated sequences families using the original annotation [19]. We outline how the highly conserved families D1, D2, J1 and J2 are clearly distinguishable and how older families such as R1 and R2, M1+ and UM are found to be spread out across branches and closer to the root of the tree. Families W1 to W5 and XM are not outlined in separate trees.

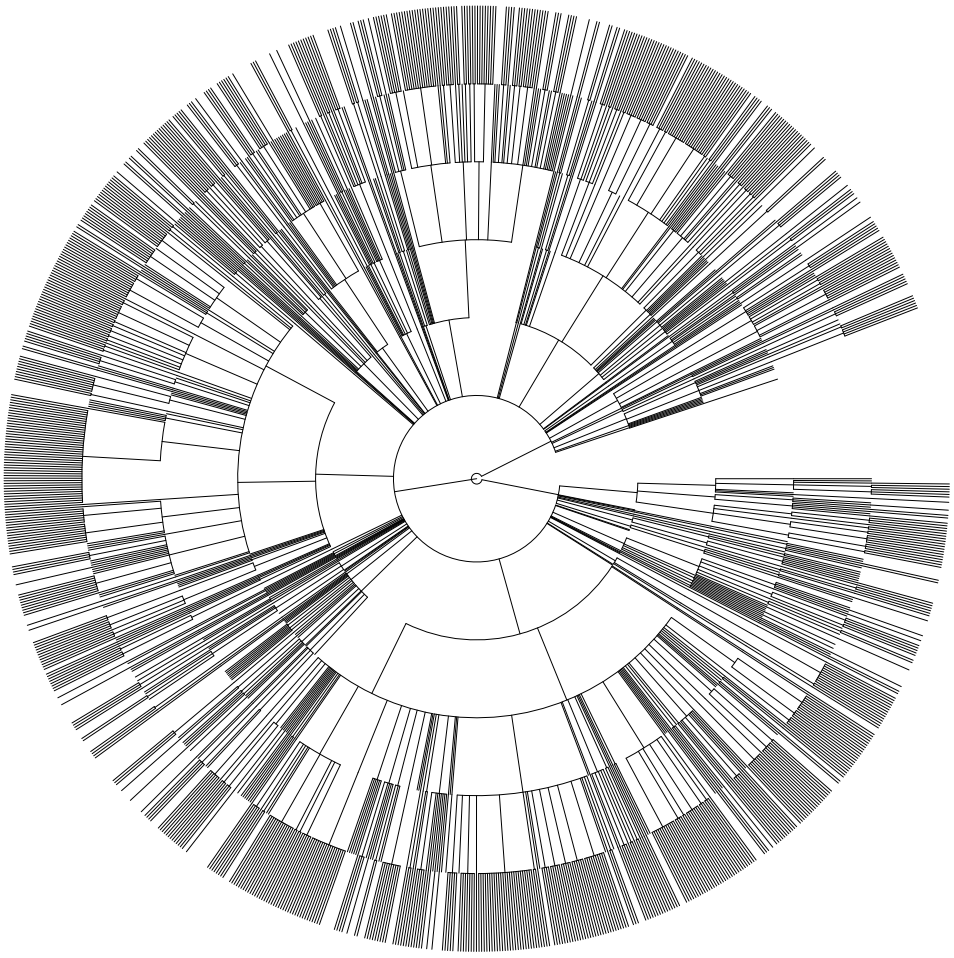

Figure 6: **Representation of the tree derived from the annotations in the Silva dataset**

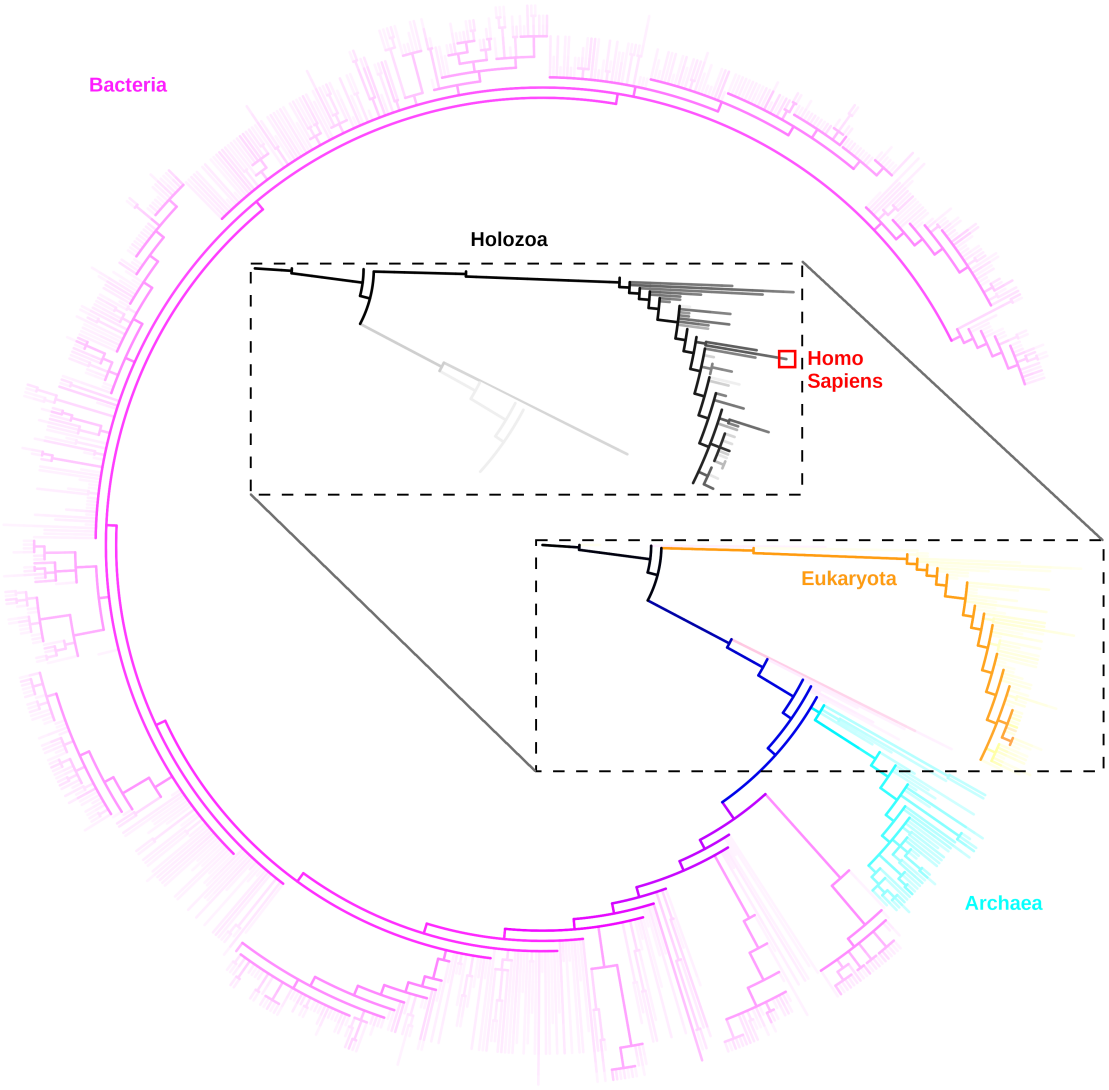

Figure 7: **Tree inferred by MNHN-Tree-Tools applied on the SILVA dataset:** Colors in the figure correspond to archeae, bacteria and eukaryota. Additionally we zoomed into the holozoa clade highlighting the branch containing the *Homo Sapiens* species.

3. In the case where we find impure clusters, how much of a mixture of clusters from the ground truth dataset are they?

To cover all these points we define the following indices to describe the quality of a tree and its clusters. l being the layer under scrutinization in the tree:

1. **The "impures over total" index**: Answers the first question in yielding the ratio of impure over the total number of clusters:
$$I_l = \frac{n_{l,\text{imp}}}{n_l}, \tag{10}$$
where I_l in the limits is either, 0: all clusters are pure, or 1: all clusters are impure.

2. **The "number of clusters correspondence" index:** Responds to the second question. One of the important advantages of using a tree based clustering technique over traditional algorithms is that one can find a tree layer for which the number of clusters corresponds closely to the number of clusters in a ground truth dataset. This index is also important as the dataset might exhibit for instance a high number of pure but small clusters with many more partitions as one might expect and hence, an unwanted result. As such we define:
$$S_l = \frac{|n_l - n_{\text{target}}|}{n_{\text{target}}}, \tag{11}$$
where n_l is the number of clusters in layer l and n_{target} the number of clusters in the ground truth dataset. Hence, the optimal values might be close to 0.

3. **The "pureness" index** covers the third point. This index is computed only taking impure clusters found in the dataset into account. It was largely modeled around the idea of recall as it is used in traditional machine learning. As we use the DBSCAN algorithm, and as such neglect datapoints in regions below a defined density at different tree layers as we search for partitions, we had to create this index, as to our knowledge, traditional indices can not handle partitions of different data sizes. The last term of equation (12) is similar to a classical recall which has been adapted to measure the amount of ground truth clusters mixture found in each impure cluster. We define:
$$P_l = \frac{1}{n_{l,\text{imp}}} \sum_{i=1}^{n_{l,\text{imp}}} \sum_{k=1}^{n_{\text{target}}} \frac{f_{l,i,k}^{<50\%}}{C_{l,i}}, \tag{12}$$
where l represents the layer of a tree, $n_{l,\text{imp}}$ the number of impure clusters, n_{target} the number of partitions in the original truth dataset, $f_{l,i,k}^{<50\%}$ the number of elements belonging to partition k of the truth dataset in the impure cluster indexed by i in layer l, if and only if this number represents less than 50% of the elements in the impure cluster. Otherwise $f_{l,i,k}^{<50\%}$ represents the total number of elements in the impure cluster minus the elements in this cluster that belong to the partition of the truth dataset that is indexed by k. Finally $C_{l,i}$ represents the total number of elements in the impure cluster in layer l indexed by i. As the last term is similar to the widely used classical recall function, we could also state that this formula describes an average of recalls computed for each clusters and adapted to our problem. $\frac{f_{l,i,k}^{<50\%}}{C_{l,i}}$ is equal to 1 if a detected cluster is a perfect mixture of ground truth clusters, and is lower than 1 otherwise. What we call a perfectly mixed cluster a cluster that contains half of the elements of one ground truth cluster and the other half of the elements of another ground truth cluster, or three thirds of three different ground truth clusters, etc. The more *pure*, i.e.

the less perfectly mixed, a cluster is the more this value tends towards 0. An illustrative example of how the pureness index is calculated is shown in figure 8.

3.3 Quality evaluation of *MNHN-Tree-Tools* results by comparison to ground truth

3.3.1 SILVA tree ground truth against *MNHN-Tree-Tools*

We start by comparing the different layers of the SILVA tree as shown in figure 6 to the tree inferred by *MNHN-Tree-Tools* shown in figure 7.

As already outlined in section 3.1.2 we investigated the first 6 layers of annotations found in the FASTA file provided with the SILVA dataset [14]. We removed all clusters containing less than 10 sequences from our SILVA ground truth tree. This removal yields for the first 6 layers the following number of resting clusters: 3, 102, 266, 658, 1278 and 3231 clusters. Knowing that layer 6 without such a suppression contains 13413 different unique annotations we did not consider this layer in the following analyses. The SILVA tree built from expert annotations provides the ground truth to which we compare the tree inferred by *MNHN-Tree-Tools* using the quality indices outlined in section 3.2. From the inferred tree, we also removed all clusters containing less than 10 sequences. The removal of clusters with less than 10 sequences in both the ground truth and detected sets allows us to drastically speed up the calculation of our quality measurement indices. The first 5 layers of the SILVA tree ground truth were compared with different tree layers obtained at different ϵ values and hence, densities inferred using our adaptive *MNHN-Tree-Tools* approach. The results of this comparison are outlined in table 1.

3.3.2 SWARM2 ground truth against *MNHN-Tree-Tools*

In the next step we compared our results obtained by *MNHN-Tree-Tools* to the clustering given by the SWARM2 [13] tool using the three quality indices outlined in section 3.2. In this case the partition provided by SWARM2 provides the ground truth for the comparison, and the entire tree inferred by *MNHN-Tree-Tools* is compared to this ground truth. The ground truth partition derived by SWARM2 contains 9540 clusters. Clusters that are smaller than 10 sequences have been removed. The results for the quality comparison to SWARM2 is outlined in table 2.

3.3.3 SILVA tree ground truth against SWARM2

To make the above comparison more interesting we not only calculated how well *MNHN-Tree-Tools* reproduces in its tree layers the clusters found by SWARM2, but also used our quality measurement indices (c.f. section 3.2) comparing the original SILVA expert annotations to SWARM2 results. As such the first 5 layers of the SILVA tree as built from the experts annotations were compared to the clustering of SWARM2. The results of this comparison are outlined in table 3.

3.3.4 Quality evaluation results

Comparing table 1 and 3, we clearly see one advantage of our multilayer tree approach. SWARM2 only yields a single partition with a number clusters that does not match to any of the ground truth layers. This is an expected result as the primary goal of SWARM2 is to detect single species Operational Taxonomic Units (OTUs). *MNHN-Tree-Tools* by contrast finds a sweet spot yielding approximately as much clusters as each of the initial layers at one of its layers as shown in table 1.

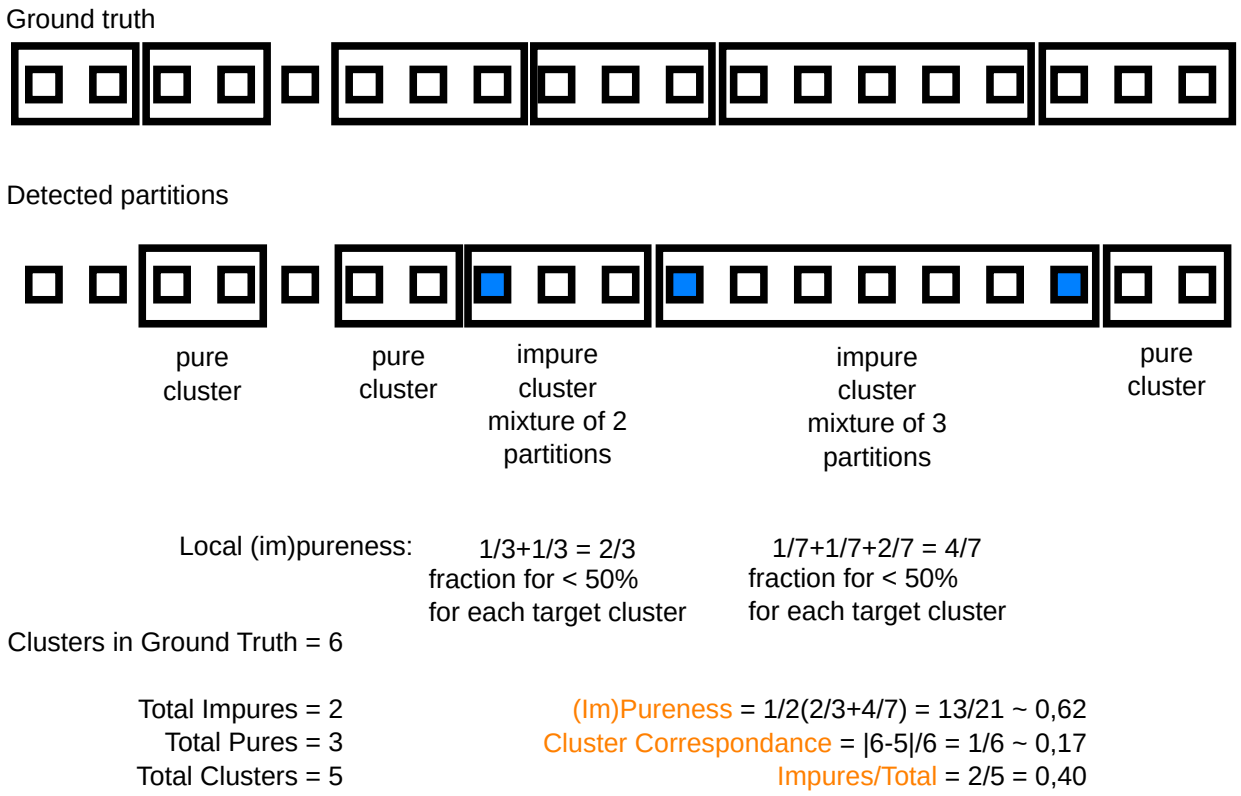

Figure 8: **Illustration on the use of our validation indices:** Two partitions, the ground truth and a detected partition, a single tree layer, are compared. The number of clusters is 6 and 5 respectively. Hence, the cluster correspondence yields 0.17. Further we count 3 pure clusters and 2 impure clusters obtaining an impures over total index is thus $2/5$. For the two impure clusters, the "pureness" is respectively $2/3$ and $4/7$. The "pureness" index is thus 0.62.
.

ϵ	Pureness	No CC	Imp/Total	ϵ	Pureness	No CC	Imp/Total
Layer 1				Layer 2			
4.90	0.22	1.33	0.14	2.00	0.28	0.42	0.17
5.35	0.22	1.00	0.17	2.05	0.28	0.28	0.17
5.45	0.22	0.67	0.20	2.10	0.31	0.18	0.16
5.50	0.22	0.67	0.20	2.15	0.36	0.07	0.15
5.70	0.22	0.33	0.25	2.20	0.33	0.01	0.15
5.95	0.22	0.00	0.33	2.25	0.36	0.04	0.16
6.05	0.22	0.00	0.33	2.30	0.48	0.15	0.13
7.20	0.22	0.67	1.00	2.35	0.44	0.26	0.14
				2.40	0.41	0.31	0.16
Layer 3				Layer 4			
1.45	0.31	0.84	0.18	1.25	0.36	0.69	0.28
1.50	0.31	0.53	0.17	1.30	0.34	0.43	0.26
1.55	0.34	0.29	0.17	1.35	0.33	0.17	0.26
1.60	0.33	0.16	0.19	1.40	0.31	0.06	0.25
1.65	0.32	0.04	0.19	1.45	0.31	0.24	0.23
1.70	0.32	0.19	0.19	1.50	0.31	0.37	0.22
1.75	0.28	0.29	0.18	1.55	0.33	0.47	0.21
1.80	0.30	0.39	0.16				
Layer 5							
0.10	0.88	0.21	0.03				
0.25	0.86	0.22	0.03				
0.30	0.82	0.18	0.03				
0.75	0.29	0.07	0.20				
0.80	0.30	0.15	0.23				
0.85	0.32	0.22	0.26				
0.90	0.33	0.30	0.29				

Table 1: **Comparison between annotated groups in different layers of the SILVA dataset and MNHN-Tree-Tools results:** Pureness, Number of Cluster Correspondence, and number of Impure over Total number of cluster is shown. Note that Pureness only reflects the Pureness of Impure clusters.

The sweet spots for each ground truth layer are in detail outlined in table 1. They are found around the regions where the middle column, which highlights the Cluster Correspondence index outlined in equation (11), tends towards 0. Nevertheless, the minimum alone of this Cluster Correspondence index is not sufficient. Considering the two other indices, shown in equations (10) and (12), the optimal *MNHN-Tree-Tools*-tree layer that corresponds best to the expert annotation might be in a close region around the best Cluster Correspondence index result where all three indices approach small values. As expected we identify that inner layers of the original SILVA annotation are found around at larger ϵ values and hence the clusters are less dense connected than the outer layers where the sweet spots, where all three indices approach small values, occur at smaller ϵ values and as such at denser regions. We further see that the found statistical clusters in specific layers found by *MNHN-Tree-Tools* closely correspond to the ground truth. We refer the reader to our hands on example outlined in figure 8 that describes the indices shown in tables 1, 2 and 3 in detail. Figure 8 should further aid the reader in evaluating these indices.

ϵ	Pureness	No CC	Imp/Total
2.10	0.31	0.18	0.16
2.15	0.36	0.07	0.15
2.20	0.33	0.02	0.15
2.25	0.36	0.04	0.17
2.30	0.48	0.15	0.13
2.35	0.44	0.26	0.14
2.40	0.41	0.31	0.16
2.45	0.46	0.41	0.12
2.50	0.52	0.52	0.11
2.55	0.48	0.54	0.10

Table 2: **Reproduction of SWARM2 clustering of the SILVA dataset by MNHN-Tree-Tools:** Pureness, Number of Cluster Correspondence, and number of Impure over Total number of clusters is shown. Note that Pureness only reflects the Pureness of Impure clusters.

Layer	Pureness	No CC	Imp/Total
1	0.14	1.00	0.33
2	0.11	0.99	0.50
3	0.10	0.97	0.42
4	0.13	0.93	0.39
5	0.17	0.87	0.41

Table 3: **SWARM2 accuracy on the SILVA Tree:** Shown are: Layer of the SILVA tree (ground truth), Pureness, Number of Cluster Correspondence, and number of Impure over the Total number of clusters is shown. Note that Pureness only reflects the Pureness of Impure clusters.

4 Simulation and Investigation of Properties

4.1 Simulation: Outlining Differences in Distance Measures

We compared the different distance measurements, L1, L2, Smith-Waterman like [15] and L2 projected on a 7 principal components subspace, by applying MNHN-Tree-Tools and its algorithms on simulated datasets.

As such we developed a sequence generator that provides datasets with the following properties: All sequences are 100 bases pairs long.

There are 10000 sequences in total in the generated dataset. We use a pseudo number generator together with a variable seed value to create an initial sequence. This sequence is then copied 1000 times. After this copying process the ensemble of 1000 sequences undergoes X single nucleotide mutations. Creating a second *evolutionary family* (c.f. section 2.2) a sequence is randomly selected from the 1000 sequences and again copied 1000 times. The new ensemble of 2000 sequences now undergoes $2X$ single nucleotide mutations. Again a sequence is chosen from the last 1000 sequences copied to create 3000 sequences which then undergo $3X$ mutations. The entire procedure is repeated until 10000 sequences are generated. The whole approach should yield 10 *evolutionary families* consisting of 1000 sequences each. We remark that this approach further yields us older sequences corresponding to the first batch of 1000 sequences while the youngest sequences are found in the last batch of 1000 sequences.

Using a custom in house built verification tool, which we ship as a part of MNHN-Tree-Tools, that allows us to check whether our algorithm is able to find clusters containing exactly 1000±20% sequences corresponding to the 10 *evolutionary families*, created by the procedure described above, in at least one layer of the resulting tree, we are able to verify the capture of simulated *evolutionary families* in *statistical clusters*.

We created the following set of simulations:

- **Set of simulations:** 200 datasets starting with 20 different seeds and setting the mutation rate X to be of either: 10, 20, 50, 100, 200, 500, 1000, 2000, 5000 or 10000. Generating this dataset we further assured that at least one mutation happens between the template sequences that generates each batch of 1000 sequences in order to create new families and not family extensions.

The datasets were processed with the adaptive clustering approach of MNHN-Tree-Tools. In the case of k-mer / PCA based distance measures the input values: $\epsilon = 0.4, \Delta\epsilon = 0.05$ and minpts = 3 were used. For Smith Waterman based tree evaluations we resorted to the input values: $\epsilon = 5$, $\Delta\epsilon = 1$ and minpts = 3. The best result where most simulated *evolutionary families* were correctly recovered as *statistical clusters* were obtained as we used the L2 norm in a 7 dimensional subspace spun by principal components obtained from a 5-mer representation of the dataset. A change prior PCA to a 4-mer or 6-mer representation did not yield significant better results. Results for L1 or L2 distances directly applied without performing PCA on the k-mer vectors yielded significantly weaker results capturing less *evolutionary families* as *statistical clusters*. Smith-Waterman distance based results proved to find less of the 10 supposed clusters than a L2-PCA based approach. The feature selection advantage that PCA provides is the crucial factor here. Results comparing the L2-PCA based approach with the Smith-Waterman approach are outlined in figure 9. Figure 9 reports the number of *evolutionary families* identified as *statistical clusters*. In the k-mer / PCA based evaluations projections of k-mers down to the first 7 principal components was performed. For different k values evaluating k-mers we see that 4-mers detect fewer clusters at mutation rates $X > 1000$ as shown in figure 9. No improvement of efficiency in using 6-mers over 5-mers has been found for the sequence lengths simulated.

4.2 Simulation: Trees, Diffusion, Distances

The simulations outlined in the previous section provided us with the important information on family capture under different distance

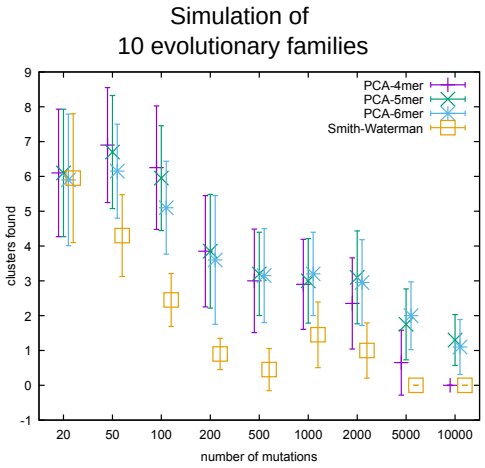

Figure 9: **Differences in distance measures:** The capture of 10 simulated evolutionary families at different mutations rates X for different k-mer lengths and PCA based sequences spaces as well as for a Smith Waterman based sequence space.

measures, and have shown us the clear advantage provided by the feature selection of PCA.

In this section we are going further and show how our algorithm clearly distinguishes proximal families whose consensus sequence only differs by a limited number of mutated base pairs. As such we performed 400 simulations of 10000 sequences that are 100 bases pairs long. The 10000 sequences are composed of two families of 5000 sequences that undergo for the first 5000 sequences N mutations and for the second half of 5000 sequences $2N$ mutations. The two families of 5000 sequences were initialized with two template sequences that differ in d nucleotides. The following parameters were chosen for the simulations: $N = [500, 1000, 2000, 10000, 20000, 50000]$ and $d = [1, 2, 3, 4]$. 20 simulated datasets from pseudo random templates were created for each combination of N and d, totalling 400. These input sets were treated with our adaptive clustering algorithm and hence, the sequences where converted to 5-mer frequency vectors. PCA was performed on these 5-mer vectors and our adaptive clustering algorithm ran using the L2 norm applied on the 7 dimensional subspace spun by the 7 principal components with the 7 largest corresponding eigenvalues. Trees from the results were built. For this run input values of $\epsilon = 0.4$, $\Delta\epsilon = 0.05$ and minpts $= 3$ were chosen. The trees highlight their number of total sequences in a logarithmic gradient from white to dark, further the two different families where colored blue and orange, and are clearly visible as the two darkest traits in most of the trees. Black areas in the tree are clusters that combine sequences of both families. The results are highlighted in figure 10. From this figure we see that the algorithm excels in the upper left area as the two families almost immediately separate at the root of the tree. The algorithm begins fail to clearly distinguish the two generated families in the lower right section as the sequences of both evolutionary fam-

ilies do not form clear distinguishable branches anymore, marking the limit of the approach presented herein.

References

[1] Alexandrov, I. A., Mitkevich, S. P., and Yurov, Y. B. The phylogeny of human chromosome specific alpha satellites. *Chromosoma 96*, 6 (Jul 1988), 443–453.

[2] Chatterji, S., Yamazaki, I., Bai, Z., and Eisen, J. A. Compostbin: A dna composition-based algorithm for binning environmental shotgun reads. In *Annual International Conference on Research in Computational Molecular Biology* (2008), Springer, pp. 17–28.

[3] Chinchor, N. The statistical significance of the muc-4 results. In *Proceedings of the 4th Conference on Message Understanding* (USA, 1992), MUC4 '92, Association for Computational Linguistics, p. 30–50.

[4] Ester, M., Kriegel, H.-P., Sander, J., and Xu, X. A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining* (1996), KDD'96, AAAI Press, p. 226–231.

[5] Felsenstein, J. Evolutionary trees from dna sequences: A maximum likelihood approach. *Journal of Molecular Evolution 17*, 6 (Nov 1981), 368–376.

[6] Forum, M. P. Mpi: A message-passing interface standard. Tech. rep., USA, 1994.

[7] Fowlkes, E. B., and Mallows, C. L. A method for comparing two hierarchical clusterings. *Journal of the American Statistical Association 78*, 383 (1983), 553–569.

[8] Hasegawa, M., Kishino, H., and Yano, T.-a. Dating of the human-ape splitting by a molecular clock of mitochondrial dna. *Journal of Molecular Evolution 22*, 2 (Oct 1985), 160–174.

[9] Jaccard, P. Étude comparative de la distribution florale dans une portion des alpes et des jura. *Bulletin del la Société Vaudoise des Sciences Naturelles 37* (1901), 547–579.

[10] Jukes, T. H., and Cantor, C. R. Chapter 24 - evolution of protein molecules. In *Mammalian Protein Metabolism*, H. MUNRO, Ed. Academic Press, 1969, pp. 21 – 132.

[11] Kazakov, A. E., Shepelev, V. A., Tumeneva, I. G., Alexandrov, A. A., Yurov, Y. B., and Alexandrov, I. A. Interspersed repeats are found predominantly in the "old" satellite families. *Genomics 82*, 6 (2003), 619 – 627.

[12] Kimura, M. Estimation of evolutionary distances between homologous nucleotide sequences. *Proceedings of the National Academy of Sciences 78*, 1 (1981), 454–458.

[13] Mahé, F., Rognes, T., Quince, C., de Vargas, C., and Dunthorn, M. Swarm v2: highly-scalable and high-resolution amplicon clustering. *PeerJ 3* (2015), e1420.

[14] Munoz, R., Yarza, P., Ludwig, W., Euzéby, J., Amann, R., Schleifer, K.-H., Oliver Glöckner, F., and Rosselló-Móra, R. Release ltps104 of the all-species living tree. *Systematic and*

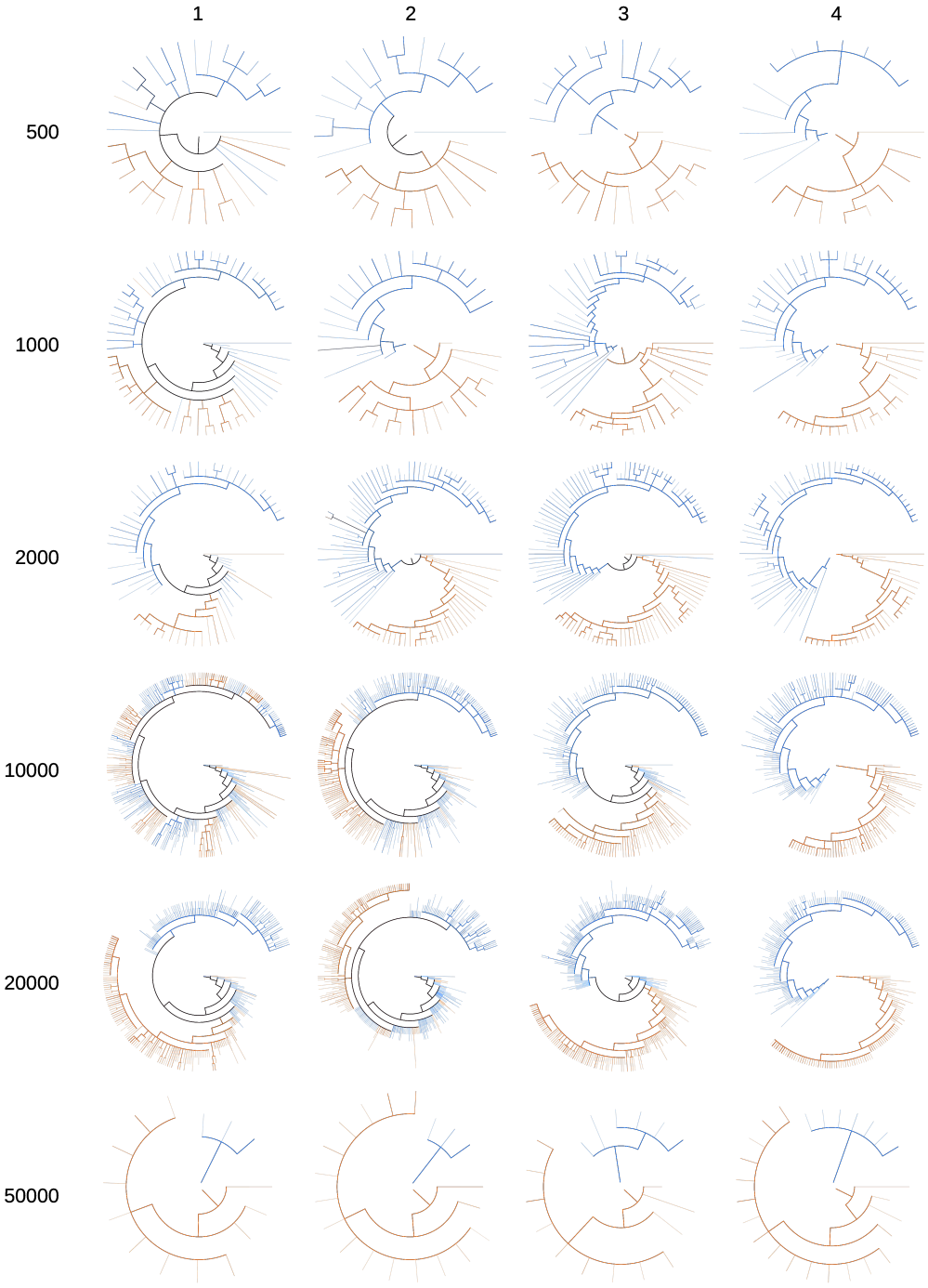

Figure 10: **Simulated evolutionary diffusion and cluster distance:** From top to bottom the N number of mutations, and from left to right the difference in nucleotides d between the generated clusters. Two clusters are generated with a distance d different nucleotides. The first undergoes $2N$ mutations to simulate an older group, the second undergoes N mutations to simulate the younger group. The older group is highlighted in orange while the younger group is highlighted in blue. Overlapping impure clusters are presented in black. The opacity of the stroke highlights the logarithm of the number of sequences in a cluster of the tree.

Applied Microbiology 34, 3 (2011), 169 – 170.

[15] Smith, T., and Waterman, M. Identification of common molecular subsequences. *Journal of Molecular Biology 147*, 1 (1981), 195 – 197.

[16] Stone, J. E., Gohara, D., and Shi, G. Opencl: A parallel programming standard for heterogeneous computing systems. *Computing in Science Engineering 12*, 3 (2010), 66–73.

[17] Tamura, K. Estimation of the number of nucleotide substitutions when there are strong transition-transversion and G+C-content biases. *Molecular Biology and Evolution 9*, 4 (07 1992), 678–687.

[18] Tamura, K., and Nei, M. Estimation of the number of nucleotide substitutions in the control region of mitochondrial DNA in humans and chimpanzees. *Molecular Biology and Evolution 10*, 3 (05 1993), 512–526.

[19] Uralsky, L., Shepelev, V., Alexandrov, A., Yurov, Y., Rogaev, E., and Alexandrov, I. Classification and monomer-by-monomer annotation dataset of suprachromosomal family 1 alpha satellite higher-order repeats in hg38 human genome assembly. *Data in Brief 24* (2019), 103708.