Journal of Machine Learning Research 19 (2018) 1-43 Submitted 3/18; Published 8/18

An Efficient and Effective Generic Agglomerative Hierarchical Clustering Approach

Julien Ah-Pine julien.ah-pine@univ-lyon2.fr
University of Lyon, Lyon 2
ERIC EA3083
5 avenue Pierre Mendès France
69676 Bron Cedex, France

Editor: Maya Gupta

Abstract

We introduce an agglomerative hierarchical clustering (AHC) framework which is generic, efficient and effective. Our approach embeds a sub-family of Lance-Williams (LW) clusterings and relies on inner-products instead of squared Euclidean distances. We carry out a constrained bottom-up merging procedure on a sparsified normalized inner-product matrix. Our method is named SNK-AHC for Sparsified Normalized Kernel matrix based AHC. SNK-AHC is more scalable than the classic dissimilarity matrix based AHC. It can also produce better results when clusters have arbitrary shapes. Artificial and real-world benchmarks are used to exemplify these points. From a theoretical standpoint, SNK-AHC provides another interpretation of the classic techniques which relies on the concept of weighted penalized similarities. The differences between group average, Mcquitty, centroid, median and Ward, can be explained by their distinct averaging strategies for aggregating clusters inter-similarities and intra-similarities. Other features of SNK-AHC are examined. We provide sufficient conditions in order to have monotonic dendrograms, we elaborate a stored data matrix approach for centroid and median, we underline the diagonal translation invariance property of group average, Mcquitty and Ward and we show to what extent SNK-AHC can determine the number of clusters.

Keywords: Agglomerative hierarchical clustering, Lance-Williams formula, Kernel methods, Scalability, Manifold learning.

1. Introduction

Clustering is the process of discovering homogeneous groups among a set of objects. There are many clustering methods and one way to differentiate one approach from another one is by the classification scheme they are based upon. On the one hand, flat clustering provides a partition of the elements. On the other hand, hierarchical clustering outputs a set of nested partitions. The latter classification type is represented by a binary tree named dendrogram.

Hierarchical clustering presents several advantages compared to flat clustering. Firstly, a dendrogram is more informative than a single partition because it provides more insights about the relationships between objects and clusters. Secondly, there is no requirement to set the number of clusters *a priori* unlike most of flat clustering techniques.

In this paper, we focus on hierarchical clustering methods. There are two kinds of procedures: agglomerative and divisive. The former type builds the dendrogram in a bottom-up

fashion whereas the latter case uses a top-down approach. We focus on *Agglomerative Hierarchical Clustering (AHC)*. Suppose we are given a pairwise dissimilarity matrix between the elements we want to cluster. The AHC bottom-up procedure initializes a trivial partition composed of singletons then, iteratively merges the two closest clusters until all items are grouped together.

In any AHC method, after each merge, it is required to compute the dissimilarity measure between the newly formed group and other existing clusters. In fact, there are as many AHC methods as dissimilarity measures. Despite the great number of approaches found in the literature, Lance and Williams (1967) proposed a parametric formula (LW formula) that generalizes a lot of them.

The bottom-up strategy described above with the LW formula form the usual stored *Dissimilarities*[1] *based AHC (D-AHC)* framework. Due to its simplicity and flexibility, it has been studied in many research works, implemented in many programming languages and successfully applied in many domains.

However, D-AHC suffers from important scalability issues since, with respect to the number of objects, it has a quadratic memory complexity and a cubic time complexity. These drawbacks severely limit the application of D-AHC to very large data sets.

In this context, our work aims at designing an AHC approach that is equivalent to D-AHC but which can be extended in order to reduce the computational costs. Furthermore, the approach we define is able to take into account the natural geometry of the data. It is thus an unsupervised approach for manifold learning as well.

In a nutshell, the contributions of the paper are the following ones:

- We focus on a sub-part of the LW formula and we establish a more general model which relies on inner-products instead of squared Euclidean distances. In this case, we need two parametric recurrence equations instead of one. Since our model relies on inner-products, it encompasses Reproducing Kernel Hilbert Spaces (RKHS) through the use of kernel functions. This first model called *Kernel matrix based Agglomerative Hierarchical Clustering (K-AHC)* can be viewed as a kind of "dual" of D-AHC when squared Euclidean distances are used as dissimilarities.

- In the usual D-AHC framework, the geometric techniques centroid, median and Ward can be carried out by using data matrices instead of distance matrices. On the contrary, the graph methods group average and Mcquitty do not enjoy such a property. We show that K-AHC enables a stored data[2] matrix approach for the two latter schemes.

- The median and centroid schemes can suffer from pathological behaviors because they can produce reversals in the dendrogram. This phenomenon appears when at an iteration, the dissimilarity value between two clusters becomes lower than the dissimilarity values observed at previous iterations. Median and centroid are said to provide non-monotonic dendrograms. In fact, Ward can be seen as a modification of

1. The terms "stored dissimilarities/similarities" and "stored data" approaches were coined by Anderberg (1973). The former one means that the input is the pairwise proximity matrix whereas the latter one indicates that the input is the data matrix where objects are described by a set of attributes.
2. Defined in the previous footnote.

centroid which enables solving the non-monotonicity issue of the latter scheme. In the same spirit, we introduce a new scheme called w-median which solves the non-monotonicity problem of the median technique.

- We propose to project the data points on an hypersphere and to shift them in order to obtain non-negative inner-products values. As a result, we obtain a *Normalized Kernel (NK) matrix* which can also be interpreted as a similarity matrix satisfying several conditions such as maximal self-similarity. In this case, we can interpret our model in terms of *weighted penalized similarities* and we show that the main differences between classic techniques rely on distinct averaging operations of inter-similarities and of intra-similarities as well.

- Given a NK matrix, we can apply sparsification procedures in order to remove non-relevant similarity relationships between objects. The resulting output is called *Sparsified Normalized Kernel (SNK) matrix* and it can be viewed as the weighted adjacency matrix of a sparse similarity graph. Then, we apply K-AHC on a SNK matrix but with the *constraint* that two clusters can be merged together providing that they have a non-null inter-similarity value. Our approach is called *Sparsified Normalized Kernel matrix based AHC (SNK-AHC)*. SNK-AHC has much lower computational costs compared to K-AHC and D-AHC, both in terms of memory and running time. Moreover, the sparsification enables capturing the intrinsic geometry of the data.

- Unlike a NK matrix, a SNK matrix is not positive semi-definite. Therefore, from a general perspective, SNK-AHC can not be interpreted from a geometrical point of view unlike K-AHC. Nevertheless, we show that in the particular cases of group average, Mcquitty and Ward, SNK-AHC still implicitly acts in an Hilbert space. This is due to the fact that these schemes are invariant with respect to any translation of the diagonal of the SNK matrix.

- By interpreting SNK-AHC in the framework of graph theory, we demonstrate that the bottom-up procedure emulates the same kinds of operations employed in order to determine the connected components of an undirected graph. As a result, we show that SNK-AHC can automatically determine the number of clusters when the latter ones are seen as connected components of a similarity graph.

- We illustrate the aforementioned properties of K-AHC and SNK-AHC on two artificial data sets. In addition, we show the superiority of SNK-AHC over D-AHC on two real-world benchmarks. Our experimental results confirm that SNK-AHC is much more scalable than the classic D-AHC. Last but not least, SNK-AHC can also outperform D-AHC in terms of clustering quality. In fact, in many cases, our approach is both more efficient and more effective than D-AHC.

The remainder of the paper is organized as follows. In section 2, we introduce the notations and some useful definitions. In section 3, we review the basics of D-AHC and of the LW formula. Then, in section 4, we introduce our K-AHC model by establishing an inner-product based expression that embeds the LW sub-equation we are interested in. Several features of K-AHC are examined as well. Afterward, we present SNK-AHC and

study its properties in section 5. Section 6 is dedicated to the experiments which are carried out on both artificial and real-world data sets. After having introduced our approach and exhibited its properties, we present and discuss in section 7 related research works. Finally, in section 8, we conclude the paper and we sketch future work as well.

2. Notations and Definitions

The set of objects (or elements or items or points) to cluster is denoted by $\mathbb{O}$ and $|\mathbb{O}|$ represents its cardinal. We suppose throughout the paper that $|\mathbb{O}|= n$. The usual AHC algorithm takes as input a pairwise dissimilarity matrix.

Definition 1 (Dissimilarity matrix) *A pairwise dissimilarity matrix of elements in $\mathbb{O}$ is denoted $\mathbf{D}$. Given $|\mathbb{O}|= n$, $\mathbf{D}$ is a square matrix of order n satisfying the following conditions:*

$$\left\{ \begin{array}{ll} \mathbf{D}_{ab} \geq 0, \quad \forall a,b \in \mathbb{O} & \textit{(non-negativity)} \\ \mathbf{D}_{ab} = \mathbf{D}_{ba}, \quad \forall a,b \in \mathbb{O} & \textit{(symmetry)} \end{array} \right. , \quad \forall a,b \in \mathbb{O}$$

Let $2^{\mathbb{O}}$ denote the set of subsets of $\mathbb{O}$. The AHC procedure builds a set of nested partitions of $\mathbb{O}$. We denote by the letters a,b,c,d,e,f any singletons (or objects) of $\mathbb{O}$, whereas i,j,k,l,m correspond to any item (or clusters) in $2^{\mathbb{O}}$. The cardinal of k is denoted $|k|$. Given k and l, their fusion (or merge or union) is denoted by (kl).

The AHC algorithm is an iterative procedure with $n-1$ steps. We denote by $\mathbb{T} = \{1,2,\ldots,n-1\}$ the set of iterations and use t to designate any of its elements.

Let $\mathbb{C}^t$ denote the set of existing clusters at iteration t. It is a partition of $\mathbb{O}$ with $n-t+1$ subsets. We denote by $\mathbf{D}^t$, the dissimilarity matrix of clusters in $\mathbb{C}^t$. It is thus a symmetric square matrix of order $n-t+1$ satisfying the conditions given in Definition 1.

The AHC bottom-up algorithm produces a set of nested partitions represented by a tree-diagram called dendrogram.

Definition 2 (Dendrogram) *A dendrogram representing a hierarchical clustering of $\mathbb{O}$ is denoted D. Given $|\mathbb{O}|= n$, D is a rooted binary tree with n leaves and $2n-1$ nodes in total. Each node i corresponds to a subset of objects. Any two distinct nodes i and j of D are such that $i \subset j$ or $j \subset i$ or $i \cap j = \emptyset$. Besides, each node i is assigned a non-negative value called the height denoted by $H(i)$.*

Since any node in a dendrogram represents a cluster, we adopt the same notations as precised above: a,b,c,d,e,f are nodes that designate singletons only, while i,j,k,l,m correspond to subsets of $\mathbb{O}$.

As an illustration, we give in Figure 1 an example of a dendrogram of the set $\mathbb{O} = \{a,b,c,d,e,f\}$.

Definition 3 (Monotonic dendrogram) *A dendrogram D is monotonic if and only if $i \subset j \Leftrightarrow H(i) \leq H(j)$, for any two distinct nodes i and j.*

The dendrogram represented in Figure 1 is monotonic. Indeed, the height of larger nodes are higher than smaller ones and any path from a leaf to the root has no reversal.

The following definition is used to compare dendrograms.

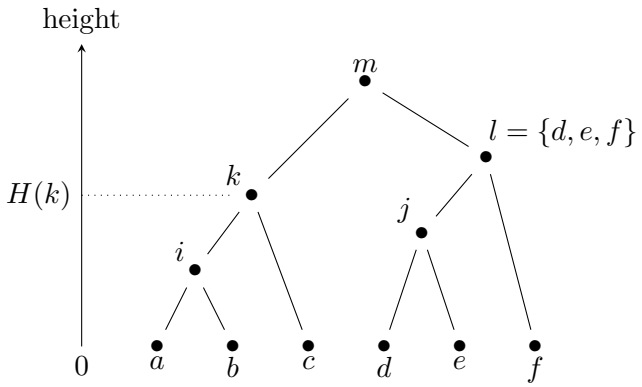

Figure 1: Illustration of a dendrogram.

Definition 4 (Sequence of merges) *A sequence of merges representing a hierarchical clustering of $\mathbb{O}$ is denoted M. Given $|\mathbb{O}|= n$, $M = (m_1, \ldots, m_{n-1})$ is a sequence of $n-1$ couples of disjoint subsets of elements of $\mathbb{O}$. For all $t \in \mathbb{T}$, $m_t = (m_t^1, m_t^2) \in \{(i,j) \in 2^{\mathbb{O}} \times 2^{\mathbb{O}}, i \cap j = \emptyset\}$. Any two elements m_s and m_t of M satisfy the following condition: if $s < t$ then $(m_s^1 \cup m_s^2) \subset (m_t^1 \cup m_t^2)$ or $(m_s^1 \cup m_s^2) \cap (m_t^1 \cup m_t^2) = \emptyset$.*

Whether an AHC technique produces a monotonic dendrogram or not, it always groups two clusters into a larger one at each iteration. If we consider the pairs of clusters that are fused at each step of the AHC bottom-up procedure (regardless the height values) then, it is clear that there is a one-to-one correspondence between dendrograms and sequences of merges.

For example, the sequence of merges in correspondence with the dendrogram provided in Figure 1 is $M = \{(\{a\},\{b\}), (\{d\},\{e\}), (\{a,b\},\{c\}), (\{d,e\},\{f\}), (\{a,b,c\},\{d,e,f\})\}$.

The following definition is used to establish the equivalence between two different AHC algorithms.

Definition 5 (Equivalent dendrograms) *Two dendrograms D and D' of a set of objects $\mathbb{O}$ are equivalent if their respective sequence of merges M and M' are identical.*

Eventually, we introduce the definition of a similarity matrix.

Definition 6 (Similarity matrix) *A pairwise similarity matrix of elements in $\mathbb{O}$ is denoted $\mathbf{S}$. Given $|\mathbb{O}|= n$, $\mathbf{S}$ is a square matrix of order n satisfying the following conditions:*

$$\begin{cases} \mathbf{S}_{ab} \geq 0 \ \textit{(non-negativity)} \\ \mathbf{S}_{ab} = \mathbf{S}_{ba} \ \textit{(symmetry)} \\ \mathbf{S}_{aa} \geq \mathbf{S}_{ab} \ \textit{(maximal self-similarity)} \end{cases}, \quad \forall a, b \in \mathbb{O}$$

The maximal self-similarity condition states that an object a can not be more similar to any other object b but to itself (except if $a = b$).

3. D-AHC : Dissimilarity based AHC

In this section, we review the basic concepts of AHC. Firstly, we introduce the usual LW formula based on dissimilarities. We also provide more detailed explanations about the bottom-up fusion mechanism that builds the dendrogram. Secondly, we review another equivalent way to express the LW formula. This formulation relies on a weighted version of the dissimilarities. In fact, the framework we introduce thereafter, is inspired from this latter expression.

3.1 The LW Formula and the Bottom-up Procedure

For $t = 1$, we initialize $\mathbb{C}^1$ to the set of n singletons with null height values and we set $\mathbf{D}^1 = \mathbf{D}$, the given dissimilarity matrix. Then, at each iteration $t \in \mathbb{T}$, D-AHC merges the[3] couple of clusters (k, l) that satisfies:

$$(k, l) = \underset{(i,j)\in\mathbb{C}^t\times\mathbb{C}^t, i\neq j}{\arg\min} \mathbf{D}^t_{ij} \tag{1}$$

Clusters k and l are fused into (kl) and the dendrogram D is amended with a new node whose height value $H((kl))$ is set to $\mathbf{D}^t_{kl}$. The new partition $\mathbb{C}^{t+1}$ is updated as follows:

$$\mathbb{C}^{t+1} = \mathbb{C}^t \setminus \{k, l\} \cup \{(kl)\} \tag{2}$$

Next, the dissimilarity values between the new cluster (kl) and the other clusters $m \in \mathbb{C}^{t+1}$ have to be computed in order to determine $\mathbf{D}^{t+1}$.

Several schemes were proposed and among the most famous ones we can cite: single linkage, complete linkage, group average (also named UPGMA[4]), Mcquitty (also named WPGMA[5]), centroid (also named UPGMC[6]), median (also named WPGMC[7]) and Ward. The first four techniques are known as graph methods whereas the three latter ones are named geometric methods.

Despite these numerous dissimilarity measures, the LW equation introduced in (Lance and Williams, 1967), is a parametric updating formula that generalizes all aforementioned cases. It is defined as follows:

$$\begin{aligned}\mathbf{D}^{t+1}_{(kl)m} = \alpha'(k,l,m)\mathbf{D}^t_{km} + \alpha'(l,k,m)\mathbf{D}^t_{lm} + \beta'(k,l,m)\mathbf{D}^t_{kl} + \gamma'|\mathbf{D}^t_{km} - \mathbf{D}^t_{lm}|,\\ \forall t \in \mathbb{T}, \forall m \in \mathbb{C}^{t+1}, m \neq (kl)\end{aligned} \tag{3}$$

where γ' is a scalar and α', β' are functions from the set of triples of disjoint subsets of $\mathbb{O}$ to $\mathbb{R}$.

In Table 1, we review the particular definitions of α', β' and γ' for the methods cited above. In this table, observe that:

- For all schemes, β' is symmetric in its two first arguments unlike α'.

3. Note that there might be several couples of clusters as solutions to (1) which could result in different dendrograms.
4. Unweighted Pair Group Method with Arithmetic mean
5. Weighted Pair Group Method with Arithmetic mean
6. Unweighted Pair Group Method Centroid
7. Weighted Pair Group Method Centroid

Method	$\alpha'(k,l,m)$	$\beta'(k,l,m)$	γ'
Single link.	$1/2$	0	$-1/2$
Complete link.	$1/2$	0	$1/2$
Group aver.	$\frac{\lvert k\rvert}{\lvert k\rvert+\lvert l\rvert}$	0	0
Mcquitty	$1/2$	0	0
Centroid	$\frac{\lvert k\rvert}{\lvert k\rvert+\lvert l\rvert}$	$-\frac{\lvert k\rvert\lvert l\rvert}{(\lvert k\rvert+\lvert l\rvert)^2}$	0
Median	$1/2$	$-1/4$	0
Ward	$\frac{\lvert k\rvert+\lvert m\rvert}{\lvert k\rvert+\lvert l\rvert+\lvert m\rvert}$	$-\frac{\lvert m\rvert}{\lvert k\rvert+\lvert l\rvert+\lvert m\rvert}$	0

Table 1: Particular settings of the LW formula (4).

- Except the Ward method, α' and β' do not depend on a third argument.
- Whatever the triple (k,l,m), α' is constant for single linkage, complete linkage, Mcquitty and median.
- Likewise, β' is constant for single linkage, complete linkage, group average, Mcquitty and median.
- Concerning γ', it is non-null only for single linkage and complete linkage.

In the rest of the paper, we only consider the sub-family of LW clusterings that satisfies $\gamma = 0$. This rules out the single and complete linkage techniques. In fact, these two latter schemes are peculiar since they reduce to the min and max operators respectively. Due to their specific features, single and complete linkages can be addressed using special algorithms (Gower and Ross, 1969; Sibson, 1973; Defays, 1977).

Consequently, we are interested in the following LW sub-formula in what follows:

$$\mathbf{D}^{t+1}_{(kl)m} = \alpha'(k,l,m)\mathbf{D}^t_{km} + \alpha'(l,k,m)\mathbf{D}^t_{lm} + \beta'(k,l,m)\mathbf{D}^t_{kl}, \quad \forall t \in \mathbb{T}, \forall m \in \mathbb{C}^{t+1}, m \neq (kl) \tag{4}$$

To wrap up this sub-section, we provide in Algorithm 1 the pseudo-code of D-AHC using the previous LW sub-formula.

3.2 An Equivalent Dissimilarity Based LW Sub-formula

Henceforth, we suppose that any object $a \in \mathbb{O}$ can be represented as a vector $\mathbf{x}^a$ in an Hilbert space $\mathcal{H}$. Moreover, we assume that dissimilarities are given by squared Euclidean distances. Thus, the general term of $\mathbf{D}$ is:

$$\mathbf{D}_{ab} = \|\mathbf{x}^a - \mathbf{x}^b\|^2, \quad \forall a, b \in \mathbb{O} \tag{5}$$

In this context, we review another dissimilarity based LW sub-formula which is equivalent to (4) and Table 1. Indeed, if objects are vectors in an Hilbert space then, the centroid

Algorithm 1: General procedure of D-AHC.

Input: $\mathbf{D}$ a dissimilarity matrix, an AHC method
Output: D a dendrogram
Initialize D with n leaves;
Set $\mathbf{D}^1 = \mathbf{D}$;
for $t = 1, \ldots, n-1$ **do**
  Find the pair of clusters (k, l) according to (1);
  Merge (k, l) into (kl) and update D;
  Compute $\mathbf{D}^{t+1}$ by applying (4) with the corresponding AHC method parameters values given in Table 1.
end

and Ward update equations can be expressed in terms of cluster representatives (see for e.g. (Murtagh and Contreras, 2012; Müllner, 2011)). Let $\mathbf{g}^i$ be the mean vector of cluster i:

$$\mathbf{g}^i = \frac{1}{|i|} \sum_{a \in i} \mathbf{x}^a, \quad i \in 2^{\mathbb{O}} \tag{6}$$

Then, for two clusters $i, j \in \mathbb{C}^t$, the dissimilarity used by the centroid scheme is:

$$\mathbf{D}^t_{ij} = \|\mathbf{g}^i - \mathbf{g}^j\|^2, \quad \forall t \in \mathbb{T} \tag{7}$$

Regarding the Ward approach, it is in fact, a weighted version of centroid since we have for the former scheme:

$$\mathbf{D}^t_{ij} = \frac{|i||j|}{|i|+|j|} \|\mathbf{g}^i - \mathbf{g}^j\|^2, \quad \forall t \in \mathbb{T}, \forall i, j \in \mathbb{C}^t \tag{8}$$

The following D-AHC iterative procedure is equivalent to Algorithm 1 (see for e.g. (Murtagh and Contreras, 2012)). For $t = 1$, let $\mathbf{D}^1$ be the input dissimilarity matrix $\mathbf{D}$ of squared Euclidean distances between data points. At each iteration, the pair (k, l) which gives the minimum *weighted dissimilarity* is merged:

$$(k, l) = \underset{(i,j) \in \mathbb{C}^t \times \mathbb{C}^t, i \neq j}{\arg\min} p(i, j) \mathbf{D}^t_{ij} \tag{9}$$

where p is a function from $\{(i, j) \in 2^{\mathbb{O}} \times 2^{\mathbb{O}}, i \cap j = \emptyset\}$, the set of pairs of disjoint subsets of $\mathbb{O}$, to $\mathbb{R}$. The definition of p for each dissimilarity scheme is provided in Table 2.

After each merge the dissimilarity matrix is updated as follows:

$$\mathbf{D}^{t+1}_{(kl)m} = \alpha(k, l)\mathbf{D}^t_{km} + \alpha(l, k)\mathbf{D}^t_{lm} + \beta(k, l)\mathbf{D}^t_{kl}, \quad \forall t \in \mathbb{T}, \forall m \in \mathbb{C}^{t+1}, m \neq (kl) \tag{10}$$

where, in this modeling, α and β are set functions whose definitions are also given in Table 2.

It is important to mention that, in the case of Ward, the set functions α and β do not depend on the third argument unlike α' and β'. Consequently, we can globally consider α and β as two-place set functions which only depend on the two clusters being fused at each iteration. More formally, α and β are functions from $\{(i, j) \in 2^{\mathbb{O}} \times 2^{\mathbb{O}}, i \cap j = \emptyset\}$ to $\mathbb{R}$

Method	$\alpha(k,l)$	$\beta(k,l)$	$p(i,j)$
Group aver.	$\frac{\lvert k\rvert}{\lvert k\rvert+\lvert l\rvert}$	0	1
Mcquitty	1/2	0	1
Centroid	$\frac{\lvert k\rvert}{\lvert k\rvert+\lvert l\rvert}$	$-\frac{\lvert k\rvert\lvert l\rvert}{(\lvert k\rvert+\lvert l\rvert)^2}$	1
Median	1/2	$-1/4$	1
Ward	$\frac{\lvert k\rvert}{\lvert k\rvert+\lvert l\rvert}$	$-\frac{\lvert k\rvert\lvert l\rvert}{(\lvert k\rvert+\lvert l\rvert)^2}$	$\frac{\lvert i\rvert\lvert j\rvert}{\lvert i\rvert+\lvert j\rvert}$
W-Median	1/2	$-1/4$	$\frac{\lvert i\rvert\lvert j\rvert}{\lvert i\rvert+\lvert j\rvert}$

Table 2: Particular settings of the LW sub-formula (10) in the model defined by (9).

As mentioned previously and by observing (7) and (8), Ward can be interpreted as a weighted version of centroid. Similarly, we introduce a *weighted version of median (w-median)*. The parameters of this new method are defined in the last row of Table 2. W-median set functions α and β are the same as median. It is the set function p which is different: instead of a uniform weight, w-median uses the same weight function as Ward. As we shall demonstrate later on, the w-median method provides monotonic dendrograms unlike the median technique.

4. K-AHC: Kernel Matrix based AHC

We have supposed that the items are represented in an Hilbert space $\mathcal{H}$ and so far, squared Euclidean distances have been used to represent the proximity relationships between points. Henceforth, we also use the underlying inner-product of $\mathcal{H}$.

In this section, we start by introducing a model that generalizes the LW sub-equation (10). Our framework relies on inner-products and it amounts to a *Kernel matrix based AHC (K-AHC)*. Thereafter, we introduce several properties of our model. We provide sufficient conditions for a technique expressed in our modeling to provide monotonic dendrograms. Furthermore, we design a stored data matrix approach that generalizes to group average and Mcquitty schemes.

4.1 Inner-product Based LW Sub-formula

Let $\langle .,. \rangle$ denotes the inner-product of $\mathcal{H}$. The geometrical data representation we assume in this work is formally stated as follows:

$$\left\{ \begin{array}{l} \mathbf{S}_{ab} = \langle \mathbf{x}^a, \mathbf{x}^b \rangle \\ \mathbf{D}_{ab} = \mathbf{S}_{aa} + \mathbf{S}_{bb} - 2\mathbf{S}_{ab} \end{array} \right. , \quad \forall a, b \in \mathbb{O} \tag{C1}$$

This implies that the matrix $\mathbf{S}$ is a *kernel (or Gram) matrix* and satisfies:

$$\left\{ \begin{array}{ll} \mathbf{S}_{ab} = \mathbf{S}_{ba}, \quad \forall a, b \in \mathbb{O} & \text{(symmetry)} \\ \mathbf{S} \succeq 0 & \text{(positive semi-definite)} \end{array} \right. \tag{11}$$

This data representation encompasses Reproducing Kernel Hilbert Spaces (RKHS). Accordingly, our approach is a kernel method (see for e.g. (Scholkopf and Smola, 2001)) which

can benefit from a large spectrum of kernel functions in order to address diverse manifold learning problems, that is K-AHC is able to detect groups of items with arbitrary shapes.

In contrast to the LW sub-formula, our model requires two equations to update the $\mathbf{S}$ matrix: one for the *off-diagonal* elements and one for the *on-diagonal* entries.

Suppose that, for $t = 1$, $\mathbf{S}$ is the input matrix of our procedure: $\mathbf{S}^1 = \mathbf{S}$. Assume that, at iteration $t \in \mathbb{T}$, clusters k and l are merged together. In our model, $\mathbf{S}^{t+1}$ is updated according to the two following recurrence equations:

$$\mathbf{S}^{t+1}_{(kl)m} = \mathfrak{a}(k,l)\mathbf{S}^t_{km} + \mathfrak{a}(l,k)\mathbf{S}^t_{lm}, \quad \forall t \in \mathbb{T}, \forall m \in \mathbb{C}^{t+1}, m \neq (kl) \tag{12a}$$

$$\mathbf{S}^{t+1}_{(kl)(kl)} = \mathfrak{b}(k,l)\mathbf{S}^t_{kl} + \mathfrak{c}(k,l)\mathbf{S}^t_{kk} + \mathfrak{c}(l,k)\mathbf{S}^t_{ll}, \quad \forall t \in \mathbb{T} \tag{12b}$$

where $\mathfrak{a}$, $\mathfrak{b}$ and $\mathfrak{c}$ are functions from $\{(i,j) \in 2^{\mathbb{O}} \times 2^{\mathbb{O}}, i \cap j = \emptyset\}$ to $\mathbb{R}$.

Similarly to D-AHC, we assume that the updated matrix is symmetric all along the procedure and thus, $\mathbf{S}^t_{m(kl)} = \mathbf{S}^t_{(kl)m}, \forall t \in \mathbb{T}, \forall m \in \mathbb{C}^{t+1}$.

For all $t \in \mathbb{T}$, let $\mathbf{\Lambda}^t$ be the square matrix of order $n - 1 + t$ with general term:

$$\mathbf{\Lambda}^t_{ij} = \mathbf{S}^t_{ij} - \frac{1}{2}(\mathbf{S}^t_{ii} + \mathbf{S}^t_{jj}), \quad \forall t \in \mathbb{T}, \forall i,j \in \mathbb{C}^t \tag{13}$$

$\mathbf{\Lambda}^t$ compares each couple of clusters in $\mathbb{C}^t$ and plays a core role in our approach. The following result establishes the connection between the LW sub-equation (10) on the one hand, and our approach (13), (12a), (12b) on the other hand.

Lemma 7 *Let $\{\mathbf{D}^t\}_{t\in\mathbb{T}}$ and $\{\mathbf{\Lambda}^t\}_{t\in\mathbb{T}}$ be the sequences of square matrices with input elements $\mathbf{D}$ and $\mathbf{S}$ and subsequent elements defined by* (10) *and* (13), (12a),(12b)*, respectively. Suppose that the related set functions α, β on the one hand and $\mathfrak{a}$, $\mathfrak{b}$, $\mathfrak{c}$ on the other hand, are such that:*

$$\begin{aligned} \mathfrak{a} &= \alpha \\ \mathfrak{b} &= -2\beta \\ \mathfrak{c} &= \alpha + \beta \end{aligned}$$

Then, under (C1) *and if $\mathfrak{a}(k,l) + \mathfrak{a}(l,k) = 1, \forall k,l \in 2^{\mathbb{O}}$, it holds:*

$$\mathbf{\Lambda}^t_{ij} = -\frac{1}{2}\mathbf{D}^t_{ij}, \quad \forall t \in \mathbb{T}, \forall i,j \in \mathbb{C}^t, i \neq j$$

Proof Under (C1), it is clear that for $t = 1$:

$$\mathbf{\Lambda}^1_{ab} = \mathbf{S}^1_{ab} - \frac{1}{2}(\mathbf{S}^1_{aa} + \mathbf{S}^1_{bb}) = -\frac{1}{2}\mathbf{D}^1_{ab}, \quad \forall a,b \in \mathbb{O}$$

We assume that the property is true for t: $\mathbf{\Lambda}^t_{ij} = \mathbf{S}^t_{ij} - \frac{1}{2}(\mathbf{S}^t_{ii} + \mathbf{S}^t_{jj}) = -\frac{1}{2}\mathbf{D}^t_{ij}, \forall i,j \in \mathbb{C}^t, i \neq j$. Then, let us prove that it is true for $t+1$ as well. Let k and l be the two clusters that are fused at iteration t. We replace in (12a) and (12b) the set functions $\mathfrak{a}$, $\mathfrak{b}$ and $\mathfrak{c}$ with α, -2β and $\alpha + \beta$ respectively. It comes:

$$\mathbf{S}^t_{(kl)m} = \alpha(k,l)\mathbf{S}^t_{km} + \alpha(l,k)\mathbf{S}^t_{lm}$$

$$\mathbf{S}^t_{(kl)(kl)} = -2\beta(k,l)\mathbf{S}^t_{kl} + (\alpha(k,l) + \beta(k,l))\mathbf{S}^t_{kk} + (\alpha(l,k) + \beta(l,k))\mathbf{S}^t_{ll}$$

Method	$\mathfrak{a}(k,l)$	$\mathfrak{b}(k,l)$	$\mathfrak{c}(k,l)$	$\mathfrak{p}(i,j)$
Group average	$\frac{\lvert k\rvert}{\lvert k\rvert+\lvert l\rvert}$	0	$\frac{\lvert k\rvert}{\lvert k\rvert+\lvert l\rvert}$	1
Mcquitty	1/2	0	1/2	1
Centroid	$\frac{\lvert k\rvert}{\lvert k\rvert+\lvert l\rvert}$	$\frac{2\lvert k\rvert\lvert l\rvert}{(\lvert k\rvert+\lvert l\rvert)^2}$	$\frac{\lvert k\rvert^2}{(\lvert k\rvert+\lvert l\rvert)^2}$	1
Median	1/2	1/2	1/4	1
Ward	$\frac{\lvert k\rvert}{\lvert k\rvert+\lvert l\rvert}$	$\frac{2\lvert k\rvert\lvert l\rvert}{(\lvert k\rvert+\lvert l\rvert)^2}$	$\frac{\lvert k\rvert^2}{(\lvert k\rvert+\lvert l\rvert)^2}$	$\frac{\lvert i\rvert\lvert j\rvert}{\lvert i\rvert+\lvert j\rvert}$
W-Median	1/2	1/2	1/4	$\frac{\lvert i\rvert\lvert j\rvert}{\lvert i\rvert+\lvert j\rvert}$

Table 3: Particular settings in our model defined by (12a), (12b) and (14).

Next, assuming $\alpha(k,l)+\alpha(l,k)=1$, $\forall k,l\in\mathbb{C}^t$, we have:

$$\mathbf{S}^t_{mm}=(\alpha(k,l)+\alpha(l,k))\mathbf{S}^t_{mm}$$

If we put all these ingredients into (13) for $t+1$, $i=(kl)$ and $j=m$; regroup terms with respect to α and β; replace $\mathbf{S}^t_{ii}+\mathbf{S}^t_{jj}-2\mathbf{S}^t_{ij}$ with $\mathbf{D}^t_{ij}$, for all $i,j\in\{k,l,m\}, i\neq j$; then we have, $\forall m\in\mathbb{C}^{t+1}$:

$$\begin{aligned}\mathbf{\Lambda}^{t+1}_{(kl)m}&=-\frac{1}{2}\left(\alpha(k,l)\mathbf{D}^t_{km}+\alpha(l,k)\mathbf{D}^t_{lm}+\beta(k,l)\mathbf{D}^t_{kl}\right)\\&=-\frac{1}{2}\mathbf{D}^{t+1}_{(kl)m}\end{aligned}$$

■

Next, we introduce our approach which also proceeds in a bottom-up manner. However, unlike D-AHC, K-AHC performs a *maximum search* at each iteration. Indeed, after having initialized a dendrogram D with n leaves, for each $t\in\mathbb{T}$, K-AHC fuses the pair of clusters (k,l) that satisfies:

$$(k,l)=\underset{(i,j)\in\mathbb{C}^t\times\mathbb{C}^t, i\neq j}{\arg\max}\ \mathfrak{p}(i,j)\mathbf{\Lambda}^t_{ij} \tag{14}$$

where $\mathfrak{p}$ is also a real-valued function whose domain is the set of pairs of disjoint subsets of $\mathbb{O}$.

At iteration t, clusters k and l are merged into (kl). The latter subset is represented by a new node in D and its "height" value is $H((kl))=\mathfrak{p}(k,l)\mathbf{\Lambda}^t_{kl}$. $\mathbb{C}^{t+1}$ is updated similarly to (2) and $\mathbf{S}^{t+1}$ is determined from $\mathbf{S}^t$ by applying (12a) and (12b).

In order to clearly state in our model the schemes under study, we provide in Table 3 the definitions of their respective set functions $\mathfrak{a}$, $\mathfrak{b}$, $\mathfrak{c}$ and $\mathfrak{p}$. Furthermore, we summarize in Algorithm 2 the K-AHC procedure.

From Lemma 7 and assuming $\mathfrak{p}=p$, it is clear that Algorithm 1 and Algorithm 2 merge the same[8] couple of clusters at each iteration. Therefore, they have equivalent dendrograms (see Definition 5). The only difference is that the dendrogram provided by K-AHC assigns

8. Assuming that if there are several (but same) solutions to (9) and (14), both algorithms pick the same pair.

Algorithm 2: General procedure of K-AHC.

Input: $\mathbf{S}$ a kernel matrix, an AHC method
Output: D a dendrogram
Initialize D with n leaves;
Set $\mathbf{S}^1 = \mathbf{S}$;
for $t = 1, \ldots, n-1$ **do**
  Find the pair of clusters (k, l) according to (14) with the corresponding AHC method parameters values given in Table 3 ;
  Merge (k, l) into (kl) and update D;
  Compute $\mathbf{S}^{t+1}$ by applying (12a) and (12b) with the corresponding AHC method parameters values given in Table 3.
end

to each node a "height" value which equals minus one half times the height value assigned to the same node of the dendrogram obtained by D-AHC. As a consequence, we have the following result.

Theorem 8 *Suppose that the conditions in Lemma 7 are satisfied. Suppose in addition, that p in (9) and $\mathfrak{p}$ in (14) are the same. Then, Algorithm 1 and Algorithm 2 provide equivalent dendrograms.*

Note that, since for all techniques listed in Table 3, $\mathfrak{a}(k,l) + \mathfrak{a}(l,k) = 1, \forall k, l \in 2^{\mathbb{O}}$ and $p = \mathfrak{p}$ then, for all particular schemes we examine, K-AHC is equivalent to D-AHC.

4.2 Monotonicity

In hierarchical clustering, it is important to know whether a method can provide reversals while building the dendrogram. Indeed, in practice, non-monotonic dendrograms can be difficult to interpret and are thus undesirable.

In the classic AHC framework described by Algorithm 1, a technique provides a monotonic dendrogram if and only if the following condition holds:

$$\mathbf{D}^{t+1}_{(kl)m} \geq \mathbf{D}^{t}_{kl}, \quad \forall t \in \mathbb{T}, \forall k, l, m \in \mathbb{C}^t \tag{15}$$

Milligan (1979), provided sufficient conditions for a method expressed in the usual LW equation (4), to output a monotonic dendrogram. It has to satisfy the following relationships:

$$\begin{cases} \alpha'(k,l,m), \alpha'(l,k,m) \geq 0 \\ \alpha'(k,l,m) + \alpha'(l,k,m) + \beta'(k,l,m) \geq 1 \\ (\gamma' \geq 0) \vee (\gamma' \leq 0 \wedge |\gamma'| \geq \alpha'(k,l,m), \alpha'(l,k,m)) \end{cases}, \quad \forall k, l, m \in 2^{\mathbb{O}} \tag{16}$$

In our approach described in Algorithm 2, the "height" value is rather a *depth value* since it varies in the opposite way than in Algorithm 1. Consequently, the monotonicity definition given previously translates as follows in the K-AHC case:

$$\mathfrak{p}((kl), m)\mathbf{\Lambda}^{t+1}_{(kl)m} \leq \mathfrak{p}(k, l)\mathbf{\Lambda}^{t}_{kl}, \quad \forall t \in \mathbb{T}, \forall k, l, m \in \mathbb{C}^t \tag{17}$$

We give below sufficient conditions for a method expressed in our model to give monotonic dendrograms.

Proposition 9 *Let* $\{\mathbf{\Lambda}^t\}_{t\in\mathbb{T}}$ *be the sequence of square matrices with input element* $\mathbf{S}$ *and subsequent elements defined by* (13), (12a), (12b). *Suppose that the set functions* $\mathfrak{a}, \mathfrak{b}, \mathfrak{c}$ *and* $\mathfrak{p}$ *satisfy:*

$$\begin{cases} \mathfrak{a}(k,l), \mathfrak{b}(k,l), \mathfrak{c}(k,l), \mathfrak{p}(k,l) \geq 0 \\ \mathfrak{a}(k,l) + \mathfrak{a}(l,k) = 1 \\ \mathfrak{b}(k,l) - \mathfrak{b}(l,k) = 0 \\ \mathfrak{c}(k,l) - \mathfrak{a}(k,l) + \frac{1}{2}\mathfrak{b}(k,l) = 0 \\ \frac{\mathfrak{a}(k,l)}{\mathfrak{p}(k,m)} + \frac{\mathfrak{a}(l,k)}{\mathfrak{p}(l,m)} - \frac{\mathfrak{b}(k,l)}{2\mathfrak{p}(k,l)} \geq 0 \\ \mathfrak{p}((kl),m)\left(\frac{\mathfrak{a}(k,l)}{\mathfrak{p}(k,m)} + \frac{\mathfrak{a}(l,k)}{\mathfrak{p}(l,m)} - \frac{\mathfrak{b}(k,l)}{2\mathfrak{p}(k,l)}\right) \geq 1 \end{cases} , \quad \forall k,l,m \in 2^{\mathbb{O}}$$

Then, under (C1), $\{\mathfrak{p}\mathbf{\Lambda}^t\}_{t\in\mathbb{T}}$ *satisfies* (17).

Proof From the definition of $\mathbf{\Lambda}^t$ given in (13), it comes:

$$\mathbf{\Lambda}^{t+1}_{(kl)m} = \mathfrak{a}(k,l)\mathbf{S}^t_{km} + \mathfrak{a}(l,k)\mathbf{S}^t_{lm} - \frac{1}{2}(\mathfrak{b}(k,l)\mathbf{S}^t_{kl} + \mathfrak{c}(k,l)\mathbf{S}^t_{kk} + \mathfrak{c}(l,k)\mathbf{S}^t_{ll} + \mathbf{S}^t_{mm})$$

By using $\mathfrak{c}(k,l) = \mathfrak{a}(k,l) - \frac{1}{2}\mathfrak{b}(k,l)$ and $\mathfrak{a}(k,l) + \mathfrak{a}(l,k) = 1$, we obtain:

$$\begin{aligned}\mathbf{\Lambda}^{t+1}_{(kl)m} = \mathfrak{a}(k,l)\mathbf{S}^t_{km} + \mathfrak{a}(l,k)\mathbf{S}^t_{lm} - \frac{1}{2}\Bigg(&\mathfrak{b}(k,l)\mathbf{S}^t_{kl} + (\mathfrak{a}(k,l) - \frac{1}{2}\mathfrak{b}(k,l))\mathbf{S}^t_{kk} \\ &+ (\mathfrak{a}(l,k) - \frac{1}{2}\mathfrak{b}(l,k))\mathbf{S}^t_{ll} + (\mathfrak{a}(k,l) + \mathfrak{a}(l,k))\mathbf{S}^t_{mm}\Bigg)\end{aligned}$$

Then, by assuming $\mathfrak{b}(k,l) - \mathfrak{b}(l,k) = 0$ and by regrouping terms with respect to $\mathfrak{a}$ and $\mathfrak{b}$, we get:

$$\mathbf{\Lambda}^{t+1}_{(kl)m} = \mathfrak{a}(k,l)\mathbf{\Lambda}^t_{km} + \mathfrak{a}(l,k)\mathbf{\Lambda}^t_{lm} - \frac{1}{2}\mathfrak{b}(k,l)\mathbf{\Lambda}^t_{kl}$$

Next, since k and l are the clusters that have been merged at iteration t, according to (14), we have $\mathfrak{p}(k,l)\mathbf{\Lambda}^t_{kl} \geq \mathfrak{p}(k,m)\mathbf{\Lambda}^t_{km}, \mathfrak{p}(l,m)\mathbf{\Lambda}^t_{lm}$. Therefore, it holds:

$$\begin{aligned}\mathbf{\Lambda}^{t+1}_{(kl)m} &\leq \mathfrak{a}(k,l)\frac{\mathfrak{p}(k,l)}{\mathfrak{p}(k,m)}\mathbf{\Lambda}^t_{kl} + \mathfrak{a}(l,k)\frac{\mathfrak{p}(k,l)}{\mathfrak{p}(l,m)}\mathbf{\Lambda}^t_{kl} - \frac{1}{2}\mathfrak{b}(k,l)\mathbf{\Lambda}^t_{kl} \\ &\leq \left(\frac{\mathfrak{a}(k,l)}{\mathfrak{p}(k,m)} + \frac{\mathfrak{a}(l,k)}{\mathfrak{p}(l,m)} - \frac{\mathfrak{b}(k,l)}{2\mathfrak{p}(k,l)}\right)\mathfrak{p}(k,l)\mathbf{\Lambda}^t_{kl}\end{aligned}$$

Under (C1), it is easy to see that $\mathbf{\Lambda}^1_{ab} \leq 0, \forall a,b \in \mathbb{O}$. Then, by assuming that $\mathfrak{p}(k,l) \geq 0$ and $\frac{\mathfrak{a}(k,l)}{\mathfrak{p}(k,m)} + \frac{\mathfrak{a}(l,k)}{\mathfrak{p}(l,m)} - \frac{\mathfrak{b}(k,l)}{2\mathfrak{p}(k,l)} \geq 0$, $\forall k,l,m \in 2^{\mathbb{O}}$, it is easy to prove by induction that $\mathbf{\Lambda}^t_{ij} \leq 0, \forall t \in \mathbb{T}, \forall i,j \in \mathbb{C}^t$, using the inequality above.

Besides, by multiplying the same previous inequality by $\mathfrak{p}((kl),m)$, we obtain an upper bound for $\mathfrak{p}((kl),m)\mathbf{\Lambda}^{t+1}_{(kl)m}$:

$$\mathfrak{p}((kl),m)\mathbf{\Lambda}^{t+1}_{(kl)m} \leq \mathfrak{p}((kl),m)\left(\frac{\mathfrak{a}(k,l)}{\mathfrak{p}(k,m)} + \frac{\mathfrak{a}(l,k)}{\mathfrak{p}(l,m)} - \frac{\mathfrak{b}(k,l)}{2\mathfrak{p}(k,l)}\right)\mathfrak{p}(k,l)\mathbf{\Lambda}^t_{kl}$$

Finally, since $\mathbf{\Lambda}_{kl}^t, \mathbf{\Lambda}_{(kl)m}^{t+1} \leq 0$ as stated previously, in order to have $\mathfrak{p}((kl),m)\mathbf{\Lambda}_{(kl)m}^{t+1} \leq \mathfrak{p}(k,l)\mathbf{\Lambda}_{kl}^t$, it is sufficient that:

$$\mathfrak{p}((kl),m)\left(\frac{\mathfrak{a}(k,l)}{\mathfrak{p}(k,m)}+\frac{\mathfrak{a}(l,k)}{\mathfrak{p}(l,m)}-\frac{\mathfrak{b}(k,l)}{2\mathfrak{p}(k,l)}\right) \geq 1$$

■

It is easy to check that all six studied techniques described in Table 3 satisfy the first fifth conditions in Proposition 9. However, unlike group average, Mcquitty and Ward, the methods centroid and median do not satisfy the last condition. These three former schemes are known to be monotonic. Regarding w-median, the new technique we introduced at the end of sub-section 3.2, we have the following property.

Proposition 10 *The w-median scheme is monotonic.*

Proof If we apply the w-median parameters values given in Table 3 to the last condition of Proposition 9, the left-hand side of the inequality reads:

$$\frac{(|k|+|l|)|m|}{|k|+|l|+|m|}\left(\frac{|k|+|m|}{2|k||m|}+\frac{|l|+|m|}{2|l||m|}-\frac{|k|+|l|}{4|k||l|}\right)$$

By developing this equation and after some manipulations, we obtain the following equivalent expression:

$$1+\frac{|m|}{4(|k|+|l|+|m|)}\left(\frac{|l|}{|k|}+\frac{|k|}{|l|}-2\right)$$

Let $r=\frac{\max(|k|,|l|)}{\min(|k|,|l|)}$ being a non-negative rational number. The term in parenthesis becomes $\left(r+\frac{1}{r}-2\right)$. It is easy to see that $r+\frac{1}{r}\geq 2$ and thus $\left(\frac{|l|}{|k|}+\frac{|k|}{|l|}-2\right)\geq 0$, which completes the proof. ■

4.3 Stored Data Matrix Approach Based on K-AHC

Let q be the dimension of the Hilbert space $\mathcal{H}$. Suppose that q is finite and let $\mathbf{X}$ be the data matrix of size $n \times q$ where each row represents a vector.

So far, we have assumed a stored proximity matrix approach where the input of the algorithms is either $\mathbf{D}$ or $\mathbf{S}$ which are both of size $O(n^2)$. However, it can be useful to operate on the data matrix $\mathbf{X}$ instead of $\mathbf{D}$ or $\mathbf{S}$. Indeed, if n is too large, it could be inefficient, or even impossible, to store the whole proximity matrix on a single machine.

If n is very large but q is much lower than n then, the *stored data matrix* approach can be carried out. It takes as input the data matrix $\mathbf{X}$, computes the dissimilarities between vectors on the fly, finds the pair of clusters to merge, represents the new cluster by a representative vector in $\mathcal{H}$ and repeat these latter steps for $t \in \mathbb{T}$. Note that such an approach allows alleviating the storage complexity but not the computational one, since, in the worst case, the proximities between all pairs of objects need to be evaluated.

In order to carry out the stored data approach, any dissimilarity scheme needs to be formulated in terms of representative vectors in $\mathcal{H}$ (see for e.g. (Murtagh and Contreras, 2012)).

We already pointed out, in sub-section 3.2, that the centroid and Ward methods can be performed by using mean vectors. Note that we can determine the latter representative vectors in an iterative fashion. For $t = 1$, we set $\mathbf{g}^a = \mathbf{x}^a, \forall a \in \mathbb{O}$. For $t > 1$, if k and l are the clusters that are fused then, the mean vector $\mathbf{g}^{(kl)}$ can be computed as follows:

$$\mathbf{g}^{(kl)} = \frac{|k|}{|k|+|l|}\mathbf{g}^k + \frac{|l|}{|k|+|l|}\mathbf{g}^l \tag{18}$$

The median and w-median schemes can also be carried out in a similar way. In these cases, clusters are represented by mid-points. Let $\mathbf{g}^i$ denote the representative vector of cluster $i \in 2^{\mathbb{O}}$. For $t = 1$, all objects are considered as singletons and we set again $\mathbf{g}^a = \mathbf{x}^a, \forall a \in \mathbb{O}$. Then, for $t > 1$, the newly formed cluster (kl) is given as follows, for median and w-median:

$$\mathbf{g}^{(kl)} = \frac{1}{2}\mathbf{g}^k + \frac{1}{2}\mathbf{g}^l \tag{19}$$

Then, for the median and w-median methods, the dissimilarity values satisfy:

$$\mathbf{D}^t_{ij} = \|\mathbf{g}^i - \mathbf{g}^j\|^2, \quad \forall t \in \mathbb{T}, \forall i, j \in \mathbb{C}^t, i \neq j \tag{20}$$

For the graph methods group average and Mcquitty, there is no such equivalent way to express their dissimilarity update equation using representative points using the usual LW sub-equation 4. Yet, our approach makes it possible to obtain such a property for these two latter schemes.

In order to introduce this property, let us first discuss the stored data matrix approach for geometric schemes in our inner-product based modeling. We have the following results.

Proposition 11 *Let $\mathbf{g}^i = \frac{1}{|i|}\sum_{a\in i}\mathbf{x}^a, \forall i \in 2^{\mathbb{O}}$. Then, for the centroid and Ward schemes, it holds:*

$$\mathbf{S}^t_{ij} = \langle \mathbf{g}^i, \mathbf{g}^j \rangle, \quad \forall t \in \mathbb{T}, \forall i, j \in \mathbb{C}^t$$

Proof Since $\mathbf{S}$ is a kernel matrix, $\mathbf{S}^1_{ab} = \langle \mathbf{g}^a, \mathbf{g}^b \rangle, \forall a, b \in \mathbb{C}^1$. Assume that $\mathbf{S}^t_{ij} = \langle \mathbf{g}^i, \mathbf{g}^j \rangle, \forall i, j \in \mathbb{C}^t$ holds for t and let us prove that it holds for $t + 1$ as well. The proof simply uses the linear property of the inner-product. Concerning $\mathbf{S}^{t+1}_{(kl)m}$, we have:

$$\begin{aligned}
\mathbf{S}^{t+1}_{(kl)m} &= \mathfrak{a}(k,l)\mathbf{S}^t_{km} + \mathfrak{a}(l,k)\mathbf{S}^t_{lm} \\
&= \frac{|k|}{|k|+|l|}\langle \mathbf{g}^k, \mathbf{g}^m \rangle + \frac{|l|}{|k|+|l|}\langle \mathbf{g}^l, \mathbf{g}^m \rangle \\
&= \frac{|k|}{|k|+|l|}\langle \frac{1}{|k|}\sum_{a\in k}\mathbf{x}^a, \mathbf{g}^m \rangle + \frac{|l|}{|k|+|l|}\langle \frac{1}{|l|}\sum_{b\in l}\mathbf{x}^b, \mathbf{g}^m \rangle \\
&= \langle \frac{1}{|k|+|l|}(\sum_{a\in k}\mathbf{x}^a + \sum_{b\in l}\mathbf{x}^b), \mathbf{g}^m \rangle \\
&= \langle \mathbf{g}^{(kl)}, \mathbf{g}^m \rangle
\end{aligned}$$

Regarding $\mathbf{S}^{t+1}_{(kl)(kl)}$, we have:

$$\begin{aligned}
\mathbf{S}^{t+1}_{(kl)(kl)} &= \mathfrak{b}(k,l)\mathbf{S}^t_{kl} + \mathfrak{c}(k,l)\mathbf{S}^t_{kk} + \mathfrak{c}(l,k)\mathbf{S}^t_{ll} \\
&= \frac{2|k||l|}{(|k|+|l|)^2}\langle \mathbf{g}^k, \mathbf{g}^l\rangle + \frac{|k|^2}{(|k|+|l|)^2}\langle \mathbf{g}^k, \mathbf{g}^k\rangle \\
&+ \frac{|l|^2}{(|k|+|l|)^2}\langle \mathbf{g}^l, \mathbf{g}^l\rangle \\
&= \langle \frac{1}{|k|+|l|}(\sum_{a\in k}\mathbf{x}^a + \sum_{b\in l}\mathbf{x}^b), \frac{1}{|k|+|l|}(\sum_{a\in k}\mathbf{x}^a + \sum_{b\in l}\mathbf{x}^b)\rangle \\
&= \langle \mathbf{g}^{(kl)}, \mathbf{g}^{(kl)}\rangle
\end{aligned}$$

■

Proposition 12 *Let $\mathbf{g}^a = \mathbf{x}^a, \forall a \in \mathbb{O}$ and $\forall t \in \mathbb{T}$, if k and l are merged, let $\mathbf{g}^{(kl)}$ be defined by* (19)*. Then, for the median and w-median schemes, it holds:*

$$\mathbf{S}^t_{ij} = \langle \mathbf{g}^i, \mathbf{g}^j\rangle, \quad \forall t \in \mathbb{T}, \forall i,j \in \mathbb{C}^t$$

Proof The proof is similar than for Proposition 11. ■

Propositions 11 and 12 states that for centroid, Ward, median and w-median, both off-diagonal and on-diagonal entries of $\mathbf{S}^t$ can be determined by inner-products of representative vectors. As far as group average and Mcquitty techniques are concerned, this latter property is only valid for off-diagonal elements of $\mathbf{S}^t$. Indeed, in Table 3, it is clear that group average and Mcquitty respectively have the same weights vectors $\{\mathfrak{a}(k,l), \mathfrak{a}(l,k)\}$ than centroid (or Ward) and median (or w-median). Regarding the on-diagonal entries, we can actually compute these values for group average and Mcquitty efficiently, providing that we initially store, in an extra vector, the squared norm of each element vector. Let $\mathbf{s}$ be the vector of size n with general term:

$$\mathbf{s}_a = \langle \mathbf{x}^a, \mathbf{x}^a\rangle, \quad \forall a \in \mathbb{O} \tag{21}$$

Let $\mathbf{s}^1 = \mathbf{s}$ and for $t = 2, \ldots, n-1$, let $\mathbf{s}^t$ be a vector of size $n-t+1$ whose component $\mathbf{s}^t_i$ is associated to cluster $i \in \mathbb{C}^t$. At each iteration $t \in \mathbb{T}$, suppose that k and l are the clusters that are merged then, $\mathbf{s}^{t+1}_{(kl)}$ is determined[9] using the following recurrence formula:

$$\mathbf{s}^{t+1}_{(kl)} = \mathfrak{c}(k,l)\mathbf{s}^t_k + \mathfrak{c}(l,k)\mathbf{s}^t_l, \quad \forall t \in \mathbb{T} \tag{22}$$

where $\mathfrak{c}$ is the set function defined for group average and Mcquitty in Table 3.

Then, it is easy to check the following property.

Proposition 13 *Let $\mathbf{s}^1_a = \langle \mathbf{x}^a, \mathbf{x}^a\rangle, \forall a \in \mathbb{O}$ and $\forall t \in \mathbb{T}$, if k and l are merged, let $\mathbf{s}^t_{(kl)}$ be defined by* (22)*. Then, for the group average and Mcquitty schemes, it holds:*

$$\mathbf{S}^t_{ii} = \mathbf{s}^t_i, \quad \forall t \in \mathbb{T}, \forall i \in \mathbb{C}^t$$

All previously discussed results are summarized in Table 4 and Algorithm 3 which provide a K-AHC based stored data matrix approach for all six methods we examine.

9. Note that similarly to $\mathbf{D}^t$ or $\mathbf{S}^t$, $\mathbf{s}^t$ loses one dimension at each iteration.

Method	$\mathbf{S}_{ij}^t, i \neq j$	$\mathbf{S}_{ii}^t$	$\mathfrak{p}(i,j)$	$\mathbf{g}^{(kl)}$	$\mathbf{s}_{(kl)}^{t+1}$
Group average	$\langle \mathbf{g}^i, \mathbf{g}^j \rangle$	$\mathbf{s}_i^t$	1	$\frac{\lvert k\rvert}{\lvert k\rvert+\lvert l\rvert}\mathbf{g}^k + \frac{\lvert k\rvert}{\lvert k\rvert+\lvert l\rvert}\mathbf{g}^l$	$\frac{\lvert k\rvert}{\lvert k\rvert+\lvert l\rvert}\mathbf{s}_k^t + \frac{\lvert k\rvert}{\lvert k\rvert+\lvert l\rvert}\mathbf{s}_l^t$
Mcquitty	$\langle \mathbf{g}^i, \mathbf{g}^j \rangle$	$\mathbf{s}_i^t$	1	$\frac{1}{2}\mathbf{g}^k + \frac{1}{2}\mathbf{g}^l$	$\frac{1}{2}\mathbf{s}_k^t + \frac{1}{2}\mathbf{s}_l^t$
Centroid	$\langle \mathbf{g}^i, \mathbf{g}^j \rangle$	$\langle \mathbf{g}^i, \mathbf{g}^i \rangle$	1	$\frac{\lvert k\rvert}{\lvert k\rvert+\lvert l\rvert}\mathbf{g}^k + \frac{\lvert k\rvert}{\lvert k\rvert+\lvert l\rvert}\mathbf{g}^l$	NA
Median	$\langle \mathbf{g}^i, \mathbf{g}^j \rangle$	$\langle \mathbf{g}^i, \mathbf{g}^i \rangle$	1	$\frac{1}{2}\mathbf{g}^k + \frac{1}{2}\mathbf{g}^l$	NA
Ward	$\langle \mathbf{g}^i, \mathbf{g}^j \rangle$	$\langle \mathbf{g}^i, \mathbf{g}^i \rangle$	$\frac{\lvert i\rvert\lvert j\rvert}{\lvert i\rvert+\lvert j\rvert}$	$\frac{\lvert k\rvert}{\lvert k\rvert+\lvert l\rvert}\mathbf{g}^k + \frac{\lvert k\rvert}{\lvert k\rvert+\lvert l\rvert}\mathbf{g}^l$	NA
W-Median	$\langle \mathbf{g}^i, \mathbf{g}^j \rangle$	$\langle \mathbf{g}^i, \mathbf{g}^i \rangle$	$\frac{\lvert i\rvert\lvert j\rvert}{\lvert i\rvert+\lvert j\rvert}$	$\frac{1}{2}\mathbf{g}^k + \frac{1}{2}\mathbf{g}^l$	NA

Table 4: Particular settings in the stored data matrix based on K-AHC and defined by (14), (13) and representative vectors updates.

Algorithm 3: General procedure of the K-AHC based stored data matrix approach.

Input: $\mathbf{X}$ a data matrix, an AHC method
Output: D a dendrogram

Initialize D with n leaves;
Set $\mathbf{g}^a = \mathbf{x}^a, \forall a \in \mathbb{O}$;
Set $\mathbf{s}_a = \langle \mathbf{x}^a, \mathbf{x}^a \rangle, \forall a \in \mathbb{O}$, if appropriate;
for $t = 1, \ldots, n-1$ **do**
    Compute the inner-product matrix of representative vectors;
    Find the pair of clusters (k, l) according to (14) and (13) with the corresponding AHC method definitions given in Table 4 ;
    Merge (k, l) into (kl) and update D;
    Compute the representative vector $\mathbf{g}^{(kl)}$ by applying the corresponding AHC method formula given in Table 4;
    Compute $\mathbf{s}_{(kl)}^{t+1}$ by applying the corresponding AHC method formula given in Table 4, if appropriate.
end

5. SNK-AHC: Sparsified Normalized Kernel Matrix Based AHC

Another important property of K-AHC is that it offers a way to address the scalability issues of stored proximity matrix based AHC procedures. Our main idea can be stated as follows. Given the distance matrix $\mathbf{D}$, it is reasonable to assume that pairs of items whose distance measure is high are unlikely to be grouped together at an early stage. Consequently, in the goal of reducing the storage complexity, these values could be discarded and we may replace them with zero in order to have a sparse $\mathbf{D}$ matrix. However, this is not sound since a zero distance measure would mean that points are identical while they are far away[10]. In order to avoid this drawback, we propose to use the inner-product matrix $\mathbf{S}$ instead, as we shall explain in what follows.

We now introduce our approach called *Sparsified Normalized Kernel matrix based AHC (SNK-AHC)*. Firstly, we introduce the normalization procedure which transform a kernel matrix $\mathbf{S}$ so that it has a constant diagonal and non-negative values. This preliminary step makes it possible to interpret the inner-product matrix $\mathbf{S}$ in terms of similarities (see Definition 6). Next, we present the sparsification procedure which aims at thresholding $\mathbf{S}$ by setting to zero the lowest values. After this, we introduce the SNK-AHC algorithm and its properties. In particular, we study an interesting feature of group average, Mcquitty and Ward methods and we show in what context, SNK-AHC is able to determine the number of clusters.

5.1 Normalized Kernel Matrix

In our perspective, the term *Normalized Kernel (NK) matrix* designates a kernel matrix with a constant diagonal and non-negative terms. In other words, we assume that the points belong to the intersection between an hypersphere and the positive quadrant of $\mathcal{H}$:

$$\mathbf{S}_{aa} = \mathbf{S}_{bb}, \quad \forall a, b \in \mathbb{O} \tag{C2}$$

$$\mathbf{S}_{ab} \geq 0, \quad \forall a, b \in \mathbb{O} \tag{C3}$$

If the kernel matrix $\mathbf{S}$ does not have a constant diagonal then, we can always apply the cosine normalization (or any generalization proposed in (Ah-Pine, 2010)):

$$\mathbf{S}_{ab} \leftarrow \frac{\mathbf{S}_{ab}}{\sqrt{\mathbf{S}_{aa}\mathbf{S}_{bb}}}, \quad \forall a, b \in \mathbb{O} \tag{23}$$

Next, let v be the minimal value in $\mathbf{S}$. If $v < 0$ then we propose to perform a simple translation in order to obtain non-negative values:

$$\mathbf{S}_{ab} \leftarrow \mathbf{S}_{ab} + |v|, \quad \forall a, b \in \mathbb{O} \tag{24}$$

It is worth noting that such a translation does not change the results of Algorithm 2. In fact, the procedure is invariant under any positive linear transformation of the $\mathbf{S}$ matrix for all six schemes we are interested in.

10. Note that we could have replaced these values with a constant which would have been the maximal distance value but in this case, we would have lost the sparsity property.

Proposition 14 *Suppose that the set functions* $\mathfrak{a}, \mathfrak{b}, \mathfrak{c}$ *satisfy:*

$$\begin{cases} \mathfrak{a}(k,l) + \mathfrak{a}(l,k) = 1 \\ \mathfrak{b}(k,l) + \mathfrak{c}(k,l) + \mathfrak{c}(l,k) = 1 \end{cases}, \quad \forall k,l,m \in 2^{\mathbb{O}}$$

Then, Algorithm 2 provides equivalent dendrograms for input similarity matrices $\mathbf{S}$ *and* $\mathbf{T} = u\mathbf{S} + v\mathbf{1}_n$*, with* $u > 0$*,* $v \in \mathbb{R}$ *and* $\mathbf{1}_n$ *being the square matrix of order* n *filled with 1.*

Proof Let $\mathbf{T}$ be a linear transformation of $\mathbf{S}$ with general term $\mathbf{T}_{ab} = u\mathbf{S}_{ab} + v$ where $u > 0$. For $t = 1$, we can write $\mathbf{T}^1_{ab} = u\mathbf{S}^1_{ab} + v$. Assume that $\mathbf{T}^t_{ij} = u\mathbf{S}^t_{ij} + v, \forall i,j \in \mathbb{C}^t$ and let us prove that $\mathbf{T}^{t+1}_{ij} = u\mathbf{S}^{t+1}_{ij} + v, \forall i,j \in \mathbb{C}^{t+1}$. Let k and l be the clusters that are fused at iteration t. First, we need to prove that $\mathbf{T}^{t+1}_{(kl)m} = u\mathbf{S}^{t+1}_{(kl)m} + v, \forall m \in \mathbb{C}^{t+1}, m \neq (kl)$. We have:

$$\begin{aligned} \mathbf{T}^{t+1}_{(kl)m} &= \mathfrak{a}(k,l)\mathbf{T}^t_{km} + \mathfrak{a}(l,k)\mathbf{T}^t_{lm} \\ &= \mathfrak{a}(k,l)(u\mathbf{S}^t_{km} + v) + \mathfrak{a}(l,k)(u\mathbf{S}^t_{lm} + v) \\ &= u(\mathfrak{a}(k,l)\mathbf{S}^t_{km} + \mathfrak{a}(l,k)\mathbf{S}^t_{lm}) + v(\mathfrak{a}(k,l) + \mathfrak{a}(l,k)) \\ &= u\mathbf{S}^{t+1}_{(kl)m} + v, \end{aligned}$$

providing that $\mathfrak{a}(k,l) + \mathfrak{a}(l,k) = 1$.

Next, we prove that $\mathbf{T}^{t+1}_{(kl)(kl)} = u\mathbf{S}^{t+1}_{(kl)(kl)} + v$. Indeed, we have:

$$\begin{aligned} \mathbf{T}^{t+1}_{(kl)(kl)} &= \mathfrak{b}(k,l)\mathbf{T}^t_{kl} + \mathfrak{c}(k,l)\mathbf{T}^t_{kk} + \mathfrak{c}(l,k)\mathbf{T}^t_{ll} \\ &= \mathfrak{b}(k,l)(u\mathbf{S}^t_{kl} + v) + \mathfrak{c}(k,l)(u\mathbf{S}^t_{kk} + v) + \mathfrak{c}(l,k)(u\mathbf{S}^t_{ll} + v) \\ &= u(\mathfrak{b}(k,l)\mathbf{S}^t_{kl} + \mathfrak{c}(k,l)\mathbf{S}^t_{kk} + \mathfrak{c}(l,k)\mathbf{S}^t_{ll}) + v(\mathfrak{b}(k,l) + \mathfrak{c}(k,l) + \mathfrak{c}(l,k)) \\ &= u\mathbf{S}^{t+1}_{(kl)(kl)} + v \end{aligned}$$

providing that $\mathfrak{b}(k,l) + \mathfrak{c}(k,l) + \mathfrak{c}(l,k) = 1$.

Denote respectively $\{\mathbf{\Lambda}^t\}_{t\in\mathbb{T}}$ and $\{\mathbf{\Delta}^t\}_{t\in\mathbb{T}}$, the sequences of square matrices defined by (13), (12a) and (12b), and obtained when $\mathbf{S}$ and $\mathbf{T} = u\mathbf{S} + v\mathbf{1}_n$ are the input kernel matrices respectively. We have, $\forall t \in \mathbb{T}, \forall i,j \in \mathbb{C}^t$:

$$\begin{aligned} \mathbf{\Delta}^t_{ij} &= \mathbf{T}^t_{ij} - \frac{1}{2}(\mathbf{T}^t_{ii} + \mathbf{T}^t_{jj}) \\ &= u\mathbf{S}^t_{ij} + v - \frac{1}{2}(u\mathbf{S}^t_{ii} + v + u\mathbf{S}^t_{jj} + v) \\ &= u(\mathbf{S}^t_{ij} - \frac{1}{2}(\mathbf{S}^t_{ii} + \mathbf{S}^t_{jj})) \\ &= u\mathbf{\Lambda}^t_{ij} \end{aligned}$$

Since $u > 0$ then, $\arg\max_{(i,j)\in\mathbb{C}^t, i\neq j} \mathfrak{p}(i,j)\mathbf{\Delta}^t_{ij} = \arg\max_{(i,j)\in\mathbb{C}^t, i\neq j} \mathfrak{p}(i,j)\mathbf{\Lambda}^t_{ij}, \forall t \in \mathbb{T}$. ■

Moreover, let us remind that a positive linear transformation of the terms of a positive semi-definite matrix provides a positive semi-definite matrix. Therefore, $\mathbf{S}$ remains a kernel matrix after (24) is carried out.

Henceforth, we assume that $\mathbf{S}$ is a NK matrix. In fact, such a matrix enjoys a double interpretation. On the one hand, it gives the inner-products of points represented (on an

hypersphere) in an Hilbert space. On the other hand, it can be seen as a similarity matrix satisfying the conditions[11] given in Definition 6.

As a consequence, in the rest of the paper, NK matrix and similarity matrix are terms that we use interchangeably.

5.2 Penalized Similarities: Aggregated Inter-similarities *Versus* Aggregated Intra-similarities

Supposing (C1), (C2) and (C3), we discuss a new interpretation of K-AHC based on similarities. Equation (14) is a maximum search over the set of couples of clusters in $\mathbb{C}^t$. The quality of a pair (i,j) depends on $\mathbf{\Lambda}^t_{ij}$ defined in (13) which is the difference between $\mathbf{S}^t_{ij}$ and the arithmetic mean of $\mathbf{S}^t_{ii}$ and $\mathbf{S}^t_{jj}$. Let us call $\mathbf{S}^t_{ij}$, the *inter-similarity* between clusters i and j, and $\mathbf{S}^t_{ii}$, the *intra-similarity* of cluster i. For the couple of clusters (i,j), $\mathbf{\Lambda}^t_{ij}$ can be seen as their inter-similarity value *penalized* by the arithmetic mean of their respective intra-similarities. According to (13), $\mathbf{\Lambda}^t_{ij}$ is great if the inter-similarity is high *and* the intra-similarities are low. Consequently, i and j are more likely to be merged together if their inter-similarity is high enough with respect to their intra-similarities.

In this context, it is important to formally state the properties of the set functions defining the six schemes we deal with. From Table 3, we can observe that for all methods:

$$\begin{cases} \mathfrak{a}(k,l), \mathfrak{b}(k,l), \mathfrak{c}(k,l) \geq 0 \\ \mathfrak{a}(k,l) + \mathfrak{a}(l,k) = 1 \\ \mathfrak{b}(k,l) + \mathfrak{c}(k,l) + \mathfrak{c}(l,k) = 1 \end{cases}, \quad \forall k,l \in 2^{\mathbb{O}} \tag{25}$$

Therefore, $\{\mathfrak{a}(k,l), \mathfrak{a}(l,k)\}$ and $\{\mathfrak{b}(k,l), \mathfrak{c}(k,l), \mathfrak{c}(l,k)\}$ can be seen as *weight vectors* and we interpret (12a) and (12b) as *averages* of inter-similarities and intra-similarities respectively. From this viewpoint, the differences between the techniques can be understood from their distinct *averaging strategies*.

In order to have a more precise view of the differences between the six methods, let us take an example. Suppose $\mathbb{O} = \{a,b,c,d,e,f,g\}$ and at iteration $t=4$, $\mathbb{C}^4 = \{k,l,m\}$ with $k = \{a,b,c\}$, $l = \{d,e\}$, $m = \{f,g\}$. Assume that (k,l) is the couple of clusters to be merged. In Figure 2, we illustrate this situation using the input similarity matrix $\mathbf{S}$. The different elements involved in (12a) and (12b) are shown. They correspond to rectangular blocks for inter-similarities $(\mathbf{S}^4_{kl}, \mathbf{S}^4_{lm}, \mathbf{S}^4_{kl})$ and to square blocks for intra-similarities $(\mathbf{S}^4_{kk}, \mathbf{S}^4_{ll}, \mathbf{S}^4_{mm})$.

The inter-similarity $\mathbf{S}^5_{(kl)m}$ is an average of $\mathbf{S}^4_{km}$ and $\mathbf{S}^4_{lm}$ represented by dashed line blocks $k \times m$ and $l \times m$. Mcquitty, median and w-median assign the same weight $1/2$ to both terms whereas group average, centroid and Ward, assign weights which depend on one of the blocks side length.

The intra-similarity $\mathbf{S}^5_{(kl)(kl)}$ is an average of $\mathbf{S}^4_{kl}$, $\mathbf{S}^4_{kk}$ and $\mathbf{S}^4_{ll}$ which are highlighted by a dotted line block and two solid line blocks respectively. Since $\mathbf{S}^4$ is symmetric, it is equivalent to consider that $\mathbf{S}^5_{(kl)(kl)}$ depends on $\mathbf{S}^4_{kl}$, $\mathbf{S}^4_{lk}$, $\mathbf{S}^4_{kk}$ and $\mathbf{S}^4_{ll}$. We can see that these four elements depict a partition of the block $(k \cup l) \times (k \cup l)$. For geometric methods, all four sub-blocks contribute to $\mathbf{S}^5_{(kl)(kl)}$. In the median and w-median schemes, all sub-blocks are assigned a uniform weight of $1/4$ which amounts to an unweighted mean. Regarding centroid

11. Note that in this context, the maximal self-similarity property is due to the Cauchy-Schwartz inequality.

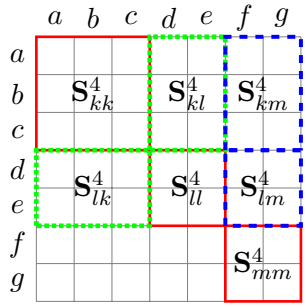

Figure 2: Illustration of inter-similarities and intra-similarities.

and Ward, the weights are distributed with respect to the blocks area. Conversely, for graph methods group average and Mcquitty, $\mathbf{S}^5_{(kl)(kl)}$ only depends on $\mathbf{S}^4_{kk}$ and $\mathbf{S}^4_{ll}$. Consequently, it is not difficult to see that only the on-diagonal terms of $\mathbf{S}$ are involved in the computation of the clusters intra-similarity for these two latter schemes. This observation was already underlined in sub-section 4.3.

5.3 Sparsified Normalized Kernel Matrix

In order to cope with the storage complexity of K-AHC, we sparsify the NK matrix by removing the lowest similarity values. We obtain a *Sparsified Normalized Kernel (SNK) matrix*.

The first sparsification procedure we introduce is a simple thresholding operator[12] based on a real parameter $\theta \geq 0$.

$$\mathbf{S}_{ab} \leftarrow \begin{cases} \mathbf{S}_{ab} & \text{if } \mathbf{S}_{ab} \geq \theta \\ 0 & \text{otherwise} \end{cases}, \quad \forall a, b \in \mathbb{O} \tag{26}$$

Note that under (C2), we have:

$$\begin{aligned} \mathbf{D}_{ab} &= \mathbf{S}_{aa} + \mathbf{S}_{bb} - 2\mathbf{S}_{ab} \\ &= 2(w - \mathbf{S}_{ab}), \quad \forall a, b \in \mathbb{O} \end{aligned}$$

where $\mathbf{S}_{aa} = w, \forall a \in \mathbb{O}$.

Thereby, the entries which correspond to the greatest values in $\mathbf{S}$ are exactly the same entries in $\mathbf{D}$ having the lowest values.

The second sparsification approach is based on the nearest neighbors. Let $\mathbb{NN}_k(a)$ be the set of the k elements closest to a according to $\mathbf{S}$ (or $\mathbf{D}$ equivalently). Then, we define:

$$\mathbf{S}_{ab} \leftarrow \begin{cases} \mathbf{S}_{ab} & \text{if } b \in \mathbb{NN}_k(a) \text{ or } a \in \mathbb{NN}_k(b) \\ 0 & \text{otherwise} \end{cases}, \quad \forall a, b \in \mathbb{O} \tag{27}$$

We point out that for each $a \in \mathbb{O}$, the number of non-null values in the similarity profile $\{\mathbf{S}_{ab}\}_{b \in \mathbb{O}}$ is lower bounded by k. Apart from the k closest items to a in $\mathbb{NN}_k(a)$, an item

12. When using distances, this sparsification procedure is equivalent to the epsilon-neighborhood method.

$c \notin \mathbb{NN}_k(a)$ could have a in its k nearest neighbors in which case $\mathbf{S}_{ca}$ but also $\mathbf{S}_{ac}$ would be non-null. Consequently, if $k = round(n/2)$ for instance, then $\mathbf{S}$ memory usage is not necessarily divided by 2, but by a factor which is lower or equal to 2.

Besides, it should be clear that determining the exact k nearest neighbors graph basically takes $O(n^2)$ time. However, there are different ways to speed-up this procedure (see for e.g. (Franti et al., 2006) and references therein).

Observe that after (26) or (27) is performed, the sparsified similarity matrix $\mathbf{S}$ is no longer positive semi-definite. Thereby, the geometric context we have assumed so far does not hold for a SNK matrix. Nonetheless, as we shall show in sub-section 5.5, three out of the six techniques are not concerned with this issue.

5.4 Performing K-AHC on a SNK Matrix in an Efficient and Effective Manner

We carry out K-AHC on a SNK matrix $\mathbf{S}$. However, owing to the distinct interpretations we can give to $\mathbf{S}$, as exposed in sub-section 5.2, we propose some substantial modifications to Algorithm 2 that lead to interesting properties.

In the stored proximities based AHC algorithms D-AHC or K-AHC, the bottleneck that causes an heavy computational cost is the search for the pair of clusters to fuse. This operation is carried out over the set of all possible pairs in $\mathbb{C}^t \times \mathbb{C}^t$ which has $O(n^2)$ cost. Since there are $n-1$ iterations, the overall time complexity is thus $O(n^3)$.

In our case, $\mathbf{S}$ is a sparse similarity matrix and we introduce the following subset:

$$\mathbb{S} = \{(a,b) \in \mathbb{O} \times \mathbb{O}, \mathbf{S}_{ab} > 0\} \tag{28}$$

Likewise, during the course of the bottom-up merging mechanism, we determine at each iteration, the following subsets:

$$\mathbb{S}^t = \{(i,j) \in \mathbb{C}^t \times \mathbb{C}^t, \mathbf{S}^t_{ij} > 0\}, \quad \forall t \in \mathbb{T} \tag{29}$$

Note that $\mathbb{S}^{t+1}$ can be easily updated from $\mathbb{S}^t$.

In contrast to K-AHC, for each $t \in \mathbb{T}$, SNK-AHC searches for the pair to merge among the elements in $\mathbb{S}^t$ only. Accordingly, we replace (14) with:

$$(k,l) = \underset{(i,j)\in\mathbb{S}^t, i\neq j}{\arg\max}\ \mathfrak{p}(i,j)\mathbf{\Lambda}^t_{ij} \tag{30}$$

Therefore, whatever the value $\mathfrak{p}(i,j)\mathbf{\Lambda}^t_{ij}$, two clusters i and j can not be merged together if they share no similarity at all. SNK-AHC is thus a *constrained AHC procedure*.

The SNK-AHC pseudo-code is given in Algorithm 4. It also describes a bottom-up algorithm but unlike K-AHC, the dendrogram grows as long as $\mathbb{S}^t \neq \emptyset$. As a consequence, the output of Algorithm 4 is not a tree in general but a *forest*. We investigate this point further in sub-section 5.6.

It is clear that restricting the search to $\mathbb{S}^t$ makes it possible to obtain a much more scalable dendrogram building procedure.

Proposition 15 *Let z be the number of non-null entries in $\mathbf{S}$ after the sparsification step in Algorithm 4 has been performed. Then, the bottom-up procedure of Algorithm 4 has $O(z)$ storage complexity and $O(nz)$ processing time complexity.*

Algorithm 4: General procedure of SNK-AHC.

Input: $\mathbf{S}$ a kernel matrix, a sparsification method, an AHC method
Output: D a dendrogram

if *the diagonal of* $\mathbf{S}$ *is not constant* **then**
  Normalize $\mathbf{S}$ using (23);
end
Translate $\mathbf{S}$ using (24);
Sparsify $\mathbf{S}$ using (26) or (27);
Initialize D with n leaves;
Set $\mathbf{S}^1 = \mathbf{S}$;
Determine $\mathbb{S}$ according to (28) and set $\mathbb{S}^1 = \mathbb{S}$;
while $\mathbb{S}^t \neq \emptyset$ **do**
  Find the pair of clusters (k, l) according to (30) with the corresponding AHC method parameters values given in Table 3 ;
  Merge (k, l) into (kl) and update D;
  Update $\mathbb{S}^{t+1}$ from $\mathbb{S}^t$;
  Compute $\mathbf{S}^{t+1}$ by applying (12a) and (12b) with the corresponding AHC method parameters values given in Table 3.
end

Note, however, that if $\mathbf{S}$ is a dense NK matrix which has not been sparsified, and $\mathbf{D}$ is the related distance matrix following (C1), then Algorithm 4 provides the exact same result as Algorithm 2 and thus an output equivalent to the one obtained with Algorithm 1 as well, according to Theorem 8.

As we shall see in section 6 dedicated to the experiments, not only SNK-AHC can be dramatically more efficient than D-AHC from a computational standpoint, but it also enables improving the quality of the clustering results on challenging problems.

5.5 Diagonal Translation Invariance

As highlighted in sub-section 5.3, the SNK matrix $\mathbf{S}$ is not positive semi-definite and we can not assume that the points belong to an Hilbert space any more. However, we can recover this feature quite easily. Indeed, since $\mathbf{S}$ is symmetric and all its diagonal entries are non-negative then, one simple way to make $\mathbf{S}$ positive semi-definite again, is to sufficiently augment the values of the diagonal entries in order to make $\mathbf{S}$ strictly diagonally dominant (see for e.g. (Horn and Johnson, 1986, Theorem 6.1.10)).

While we can always do this, we show, in what follows, that this is not necessary for some techniques.

Let us introduce the following matrix:

$$\mathbf{T} = \mathbf{S} + w\mathbf{I}_n \tag{31}$$

where $\mathbf{I}_n$ is the identity matrix of order n and $w > 0$ is chosen such that $\mathbf{T}$ is positive semi-definite.

Let $M = \{m_1, \ldots, m_{n-1}\}$ be a sequence of merges following Definition 4. By observing that (12a) does not depend on any on-diagonal entry of the SNK matrix then, it is easy to check the following result.

Lemma 16 *Let $\{\mathbf{S}^t\}_{t\in\mathbb{T}}$ and $\{\mathbf{T}^t\}_{t\in\mathbb{T}}$ be the sequences of square matrices with input elements $\mathbf{S}$ and $\mathbf{T} = \mathbf{S} + w\mathbf{I}_n, w \in \mathbb{R}$ and subsequent elements defined by* (12a) *and* (12b)*. Suppose that $\{\mathbf{S}^t\}_{t\in\mathbb{T}}$ and $\{\mathbf{T}^t\}_{t\in\mathbb{T}}$ are determined with respect to the same sequence of merges M. Then, we have:*

$$\mathbf{T}^t_{ij} = \mathbf{S}^t_{ij}, \quad \forall t \in \mathbb{T}, \forall i, j \in \mathbb{C}^t, i \neq j$$

Lemma 16 indicates that when merging the same sequence of pairs of clusters, the off-diagonal entries of similarity matrices $\{\mathbf{S}^t\}_{t\in\mathbb{T}}$ and $\{\mathbf{T}^t\}_{t\in\mathbb{T}}$ are identical. It is only the intra-similarities that are influenced by the diagonal translation.

For group average, Mcquitty and Ward, we have the following relationships.

Lemma 17 *Let $\{\mathbf{S}^t\}_{t\in\mathbb{T}}$ and $\{\mathbf{T}^t\}_{t\in\mathbb{T}}$ be the sequences of square matrices with input elements $\mathbf{S}$ and $\mathbf{T} = \mathbf{S} + w\mathbf{I}_n, w \in \mathbb{R}$ and subsequent elements defined by* (12a) *and* (12b)*. Suppose that $\{\mathbf{S}^t\}_{t\in\mathbb{T}}$ and $\{\mathbf{T}^t\}_{t\in\mathbb{T}}$ are determined with respect to the same sequence of merges M. Then, for group average and Mcquitty, we have:*

$$\mathbf{T}^t_{ii} = \mathbf{S}^t_{ii} + w, \quad \forall t \in \mathbb{T}, \forall i \in \mathbb{C}^t$$

Regarding Ward, we have:

$$\mathbf{T}^t_{ii} = \mathbf{S}^t_{ii} + \frac{w}{|i|}, \quad \forall t \in \mathbb{T}, \forall i \in \mathbb{C}^t$$

Proof Let us consider the group average technique and its parameters values given in Table 3. For $t = 1$, it follows from the definition given in (31) that $\mathbf{T}^1_{aa} = \mathbf{S}^1_{aa} + w, \forall a \in \mathbb{C}^1$. Assume that $\mathbf{T}^t_{ii} = \mathbf{S}^t_{ii} + w, \forall i \in \mathbb{C}^t$ for t. Then, let us prove that the latter relation is true for $t+1$ as well. Suppose that at iteration t, the pair of clusters (k, l) is merged. By applying Lemma 16, it comes:

$$\begin{aligned}
\mathbf{T}^{t+1}_{(kl)(kl)} - \mathbf{S}^{t+1}_{(kl)(kl)} &= \mathfrak{b}(k,l)\mathbf{T}^t_{kl} + \mathfrak{c}(k,l)\mathbf{T}^t_{kk} + \mathfrak{c}(l,k)\mathbf{T}^t_{ll} - (\mathfrak{b}(k,l)\mathbf{S}^t_{kl} + \mathfrak{c}(k,l)\mathbf{S}^t_{kk} + \mathfrak{c}(l,k)\mathbf{S}^t_{ll}) \\
&= \mathfrak{c}(k,l)(\mathbf{T}^t_{kk} - \mathbf{S}^t_{kk}) + \mathfrak{c}(l,k)(\mathbf{T}^t_{ll} - \mathbf{S}^t_{ll}) \\
&= w(\mathfrak{c}(k,l) + \mathfrak{c}(l,k)) \\
&= w
\end{aligned}$$

since $\mathfrak{c}(k,l) + \mathfrak{c}(l,k) = 1$ for group average.
The proofs for Mcquitty and Ward are similar. ■

Theorem 18 *For group average, Mcquitty and Ward methods, Algorithm 4 provides equivalent dendrograms for input similarity matrices $\mathbf{S}$ and $\mathbf{T} = \mathbf{S} + w\mathbf{I}_n$, $w \in \mathbb{R}$.*

Proof Denote respectively $\{\mathbf{\Lambda}^t\}_{t\in\mathbb{T}}$ and $\{\mathbf{\Delta}^t\}_{t\in\mathbb{T}}$, the sequences of penalized similarity matrices obtained using $\mathbf{S}$ and $\mathbf{T}$ as input similarity matrices. We prove that for all $t \in \mathbb{T}$, the couple of clusters maximizing $\mathfrak{p}(i,j)\mathbf{\Delta}^t_{ij}$ is the same as the one maximizing $\mathfrak{p}(i,j)\mathbf{\Lambda}^t_{ij}$. To this end, note that a sufficient condition is $\mathfrak{p}(i,j)(\mathbf{\Delta}^t_{ij} - \mathbf{\Lambda}^t_{ij}) = c$, $\forall t \in \mathbb{T}$, where c is a constant. For $t = 1$, we have:

$$\begin{aligned}\mathfrak{p}(a,b)(\mathbf{\Delta}^1_{ab} - \mathbf{\Lambda}^1_{ab}) &= \mathfrak{p}(a,b)(\mathbf{T}^1_{ab} - \frac{1}{2}(\mathbf{T}^1_{aa} + \mathbf{T}^1_{bb}) - \mathbf{S}^1_{ab} + \frac{1}{2}(\mathbf{S}^1_{aa} + \mathbf{S}^1_{bb})) \\ &= -\frac{\mathfrak{p}(a,b)}{2}((\mathbf{T}^1_{aa} - \mathbf{S}^1_{aa}) + (\mathbf{T}^1_{bb} - \mathbf{S}^1_{bb})) \\ &= \underbrace{-\mathfrak{p}(1,1)w}_{c}\end{aligned}$$

where, by abusing the notation, $\mathfrak{p}(1,1)$ denotes $\mathfrak{p}(i,j)$ whenever $|i|= |j|= 1$.

For $t = 1$, it is clear that using either $\mathbf{S}$ or $\mathbf{T}$ as input matrices, leads to the same merge. As a consequence, by applying Lemma 16, it is sufficient to prove by induction that:

$$-\frac{\mathfrak{p}(i,j)}{2}((\mathbf{T}^t_{ii} - \mathbf{S}^t_{ii}) + (\mathbf{T}^t_{jj} - \mathbf{S}^t_{jj})) = -\mathfrak{p}(1,1)w, \quad \forall t \in \mathbb{T}, \forall i,j \in \mathbb{C}^t, i \neq j \tag{32}$$

Concerning the group average and the Mcquitty techniques, for both cases $\mathfrak{p}(i,j) = 1, \forall i,j \in 2^{\mathbb{O}}$ and thus $c = -w$. Let assume that at iteration t, the pair of clusters (k,l) is merged. By applying Lemma 17, we have:

$$\begin{aligned}-\frac{1}{2}((\mathbf{T}^{t+1}_{(kl)(kl)} - \mathbf{S}^{t+1}_{(kl)(kl)}) + (\mathbf{T}^{t+1}_{mm} - \mathbf{S}^{t+1}_{mm})) &= -\frac{1}{2}(w + w) \\ &= -w\end{aligned}$$

In the case of Ward, $c = -\mathfrak{p}(1,1)w = -\frac{w}{2}$. By using Lemma 17 again, we have:

$$\begin{aligned}&-\frac{\mathfrak{p}((kl),m)}{2}((\mathbf{T}^{t+1}_{(kl)(kl)} - \mathbf{S}^{t+1}_{(kl)(kl)}) + (\mathbf{T}^{t+1}_{mm} - \mathbf{S}^{t}_{mm})) \\ &= -\frac{|(kl)||m|}{2(|(kl)|+|m|)}(\frac{w}{|(kl)|} + \frac{w}{|m|}) \\ &= -\frac{w}{2}\end{aligned}$$

■

Consequently, for these three schemes, the geometrical representation of the objects lying in an Hilbert space is still valid when applying Algorithm 4, even though the SNK matrix $\mathbf{S}$ is not positive semi-definite.

For the other schemes, centroid, median and w-median, some preliminary empirical tests showed that making $\mathbf{S}$ positive semi-definite again is not recommended. Indeed, in these cases, increasing the diagonal provided worst performances in terms of clustering quality. It appears that such a transformation results in a space distortion to which, these latter methods are highly sensitive. Therefore, in Algorithm 4, we do not include a step for diagonal translation by default.

5.6 Clusters as Connected Components

One important issue in clustering is to determine the number of clusters. To some extent, Algorithm 4 is able to address this challenge. In order to detail this property, we place ourselves in the framework of graph theory.

Let G be an undirected graph with $\mathbb{O}$ being the set of nodes and $\mathbb{S}$, defined in (28), being the set of edges. G is connected if for every pair $(a, b) \in \mathbb{O} \times \mathbb{O}$, there is a path joining both nodes. If G is not connected then $\mathbb{O}$ can be separated with respect to its *connected components*. These latter subsets form a partition of $\mathbb{O}$. From a clustering viewpoint, the connected components can be seen as clusters.

One way to determine the connected components of an undirected graph is to use a disjoint sets data structure which typically: (i) puts nodes in a same set if there is a path joining each other and (ii) assigns a representative item to each set (see for e.g. (Cormen et al., 2009, Chapter 21)). In this context, three operations are employed:

- `make_set`(a): creates a set whose only member is a and takes a as its representative.
- `find_set`(a): finds the representative of the set a belongs to.
- `union`(a,b): unites the two disjoint sets that a and b belong to, removes the two latter sets and determine a representative for the new set.

We review in Algorithm 5 the pseudo-code that builds a disjoint sets data structure given a graph $G = (\mathbb{O}, \mathbb{S})$ and outputs the connected components.

Algorithm 5: Connected components determination.

Input: $G = (\mathbb{O}, \mathbb{S})$
Output: The connected components
for $a \in \mathbb{O}$ **do**
  `make_set`(a)
end
for $(a, b) \in \mathbb{S}$ **do**
  if `find_set`(a) $\neq$ `find_set`(b) **then**
    `union`(a,b)
  end
end

As an illustration, we provide an example in Figure 3. We represent a graph with seven nodes and four edges : $\mathbb{O} = \{a, b, c, d, e, f, g\}$ and $\mathbb{S} = \{(a, b), (a, c), (d, e), (e, f)\}$.

Applying Algorithm 5 to this example, provides the following connected components:

- $a : \{a, b, c\}$,
- $d : \{d, e, f\}$.

In this example, the representative item of a subset is chosen with respect to the lexical order.

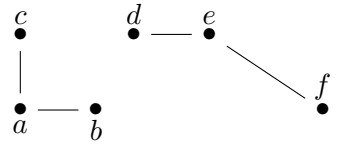

Figure 3: Illustration of a disconnected graph.

In fact, all AHC Algorithms 1-4 we have introduced so far, rely on a bottom-up fusion mechanism that reproduces the same operations than in Algorithm 5. The difference is that instead of scanning all unitary edges $(a, b) \in \mathbb{S}$ in the for loop in Algorithm 5, AHC procedures go through the consolidated edges between the representative items of the disjoint sets (the clusters). Moreover, unlike in Algorithm 5, the edges are not picked randomly in AHC algorithms but they are chosen in the goal of optimizing a criterion which is the weighted dissimilarity value in the case of D-AHC and the weighted penalized similarity value in the cases of K-AHC and SNK-AHC.

Furthermore, in Algorithms 1 and 2, the input proximity matrices are dense and the underlying graphs are thus fully connected. Therefore, these algorithms necessarily produce single trees as outputs. On the contrary, Algorithm 4 uses a sparse similarity matrix and in this case, G might not be connected, especially if $z << n^2$. In such a case, Algorithm 4 outputs a forest and each tree is a connected component which can be considered as a cluster.

This reasoning is summarized in the following statement.

Proposition 19 *Let* $\mathbf{S}$ *be the sparse similarity matrix obtained after the sparsification step of Algorithm 4 and let* $\mathbb{S}$ *be the set of pairs of objects defined by* (28)*. Let* $G = (\mathbb{O}, \mathbb{S})$ *be the associated undirected graph. If* G *is not connected and has* κ *connected components then Algorithm 4 stops at iteration* $n - \kappa - 1$*. Moreover, it outputs a forest where each tree is a connected component.*

Accordingly, note that SNK-AHC output is not a complete dendrogram in general.

6. Experiments

In this section we demonstrate the different properties and advantages of SNK-AHC over D-AHC using different benchmark data sets. We both use artificial and real-world problems which are freely available at (Franti and et al, 2015) and (Lichman, 2013) respectively.

Firstly, we compare the dendrograms obtained by D-AHC and SNK-AHC. Our first purpose is to verify that when no sparsification is performed, SNK-AHC is equivalent to D-AHC. Secondly, we are interested in assessing the proximity between the dendrograms given by D-AHC and SNK-AHC. Thirdly, on medium-size real-world data sets, we demonstrate that Algorithm 4 indeed allows reducing the D-AHC computational costs dramatically. Finally, for all data sets, we show that sparsifying the similarity matrix can also provide better clustering results.

We introduce below the different assessment criteria we used in our experiments, before presenting the benchmarks and the results we obtained.

6.1 Evaluation Measures

In order to measure the proximity between the dendrograms obtained by Algorithms 1 and 4, we use the cophenetic matrices and correlation coefficient (see for e.g. (Everitt et al., 2009, Section 4.4.2)). Given a dendrogram D, the derived cophenetic matrix denoted $\mathbf{C}(D)$, is a pairwise matrix of order n where, for each pair of items $(a, b) \in \mathbb{O} \times \mathbb{O}$, we record the height (D-AHC) or depth value (SNK-AHC) of the node that merges a and b for the first time. Then, let D_{dahc} and D_{snkahc} be the dendrograms obtained by both techniques. The cophenetic correlation is the product moment correlation between the vectorized upper triangular matrices of $\mathbf{C}(D_{dahc})$ and $\mathbf{C}(D_{snkahc})$. We take the opposite value of this measure which is denoted CC. In this case, CC=1 implies that the dendrograms are equivalent.

Next, in order to evaluate the quality of the clustering results we apply an external validation methodology since we are given the correct partition for all data sets. For each obtained dendrogram, we cut the forest so as to obtain the correct number of clusters denoted κ^*. Note that if κ, the number of clusters found by Algorithm 4, is greater than κ^* then, we keep the partition with κ clusters. Afterward, we compare the resulting partition and the ground-truth. The evaluation measure used in this case is the famous adjusted Rand index (Hubert and Arabie, 1985) which is denoted ARI. In this case as well, the greater the better, and ARI=1 means that the ground-truth was recovered perfectly.

Regarding scalability, our baselines are the D-AHC computational costs. Therefore, the memory and running time reductions are reported in comparison to the performances obtained by D-AHC. Relative storage and processing time decreases are thus examined. However, note that these points are mainly analyzed in the case of real-world benchmarks. Indeed, synthetic data sets are small-size and, in these cases it is not worth discussing computational gains in details.

6.2 Artificial Data

Small-size artificial data sets are used in the goal of illustrating the ability of SNK-AHC to address challenging clustering tasks.

In all experiments, before computing the inner-product matrix, we centered and scaled the data matrix with respect to the mean and standard deviation of the variables.

6.2.1 Aggregation Data

The first benchmark is taken from (Gionis et al., 2007). It consists of 788 points in a two-dimensional space. The objects and clusters are represented in Figure 4. There are seven different groups to identify. These clusters have different sizes and shapes. They can be non-convex and connected as well.

In (Gionis et al., 2007), the authors show the shortcomings of classic D-AHC methods such as the single linkage, complete linkage, group average and Ward schemes. The k-means algorithm also fails to recover the seven clusters. In this previous work, Euclidean distances were used.

In Table 5, we report the performance measures obtained by the different schemes used in the framework of SNK-AHC. A Gaussian kernel and the nearest neighbors sparsification operator $\mathbb{NN}_k$ were applied.

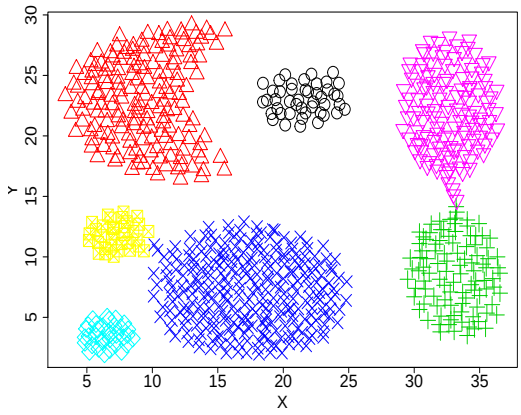

Figure 4: Aggregation data set.

Regarding the Gaussian kernel, we remind its definition below:

$$\mathbf{S}_{ab} = \exp(-\gamma \|\mathbf{x}^a - \mathbf{x}^b\|^2), \quad \forall a, b \in \mathbb{O}$$

We set $\gamma = 1/q$, q being the number of descriptive variables. This default setting is used in popular SVM tools like (Chang and Lin, 2011).

Concerning $\mathbb{NN}_k$, the distinct k values were successively set to (the nearest integer of) $\{100, 90, 75, 50, 25, 10, 1\}$ percent of n, the total number of items. This results in a sequence of sparser and sparser SNK matrices. The two opposite cases are the following ones. $k = n$ corresponds to 100% of the nearest neighbors and in that case, no sparsification is carried out. Applying SNK-AHC without any sparsification is equivalent to K-AHC or D-AHC. This situation corresponds to our baseline. By contrast, setting $k = round(n/100)$ refers to the sparsest similarity matrix that we used.

From Table 5, we observe that:

- For all schemes, when $k = 788$, CC=1. In other words, the obtained dendrograms are equivalent to the ones given by D-AHC. These observations empirically confirm Theorem 8.

- For all methods, a sparse $\mathbf{S}$ matrix that was reduced by up to half of its original memory size, does not alter the quality of SNK-AHC outputs. Therefore, we can improve the scalability without decreasing the ARI values. In addition, we note that the CC values are close to 1. It is likely that the dendrograms we obtain in theses cases are equivalent to the baseline.

- Below fifty percent of nearest neighbors, the performances are not stable and the ARI behavior depends on the method. For median and w-median, the sparsification beyond fifty percent, has a negative effect.

- Conversely, among the interesting cases, group average with only one percent of nearest neighbors was able to recover the correct partition with seven classes (ARI=1).

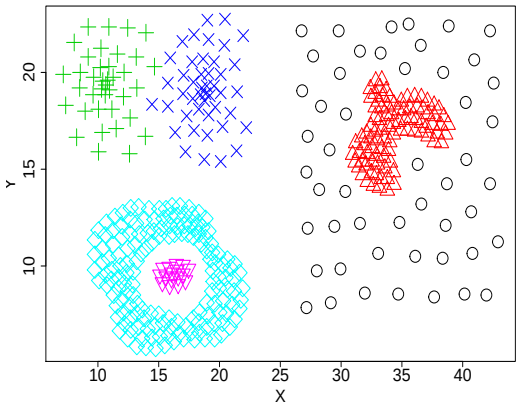

Figure 5: Compound data set.

With the same sparsification level, Ward dramatically improves over its baseline (dense $\mathbf{S}$) since the ARI value increases from 0.688 to 0.965.

- For all methods, κ, the number of clusters found by SNK-AHC, equals one except when $k = 8$. The latter setting corresponds to the sparsest similarity matrix. Only in this case, the underlying graph becomes disconnected and all schemes found five clusters. In Figure 4, the groups that were discovered are the three disconnected clusters (red triangles, black circles and cyan diamonds) and the two pairs of connected clusters which are respectively put together.

6.2.2 Compound Data

The second synthetic data set is a composition of several clustering tasks originally proposed in (Zahn, 1971). It consists of 399 points in a two-dimensional space. The data set is shown in Figure 5. There are six distinct groups of points that are identified with different symbols and colors. This task is particularly challenging since the clusters present very different patterns and are highly non-convex and non-linearly separable.

Similarly to the previous case, we applied a Gaussian kernel using the same default setting. However, the sparsification method we used here is based on a threshold following (26). The different θ values were chosen so that a certain level of sparsity is reached. Precisely, they correspond to the $\{100, 90, 75, 50, 25, 10, 1\}$th percentiles of the similarity values distribution. The 100th percentile does not yield any sparsification. Again, this case is considered as our baseline. On the contrary, the 1th percentile setting means that 99% of the similarity values were thresholded to zero. This latter case is the sparsest $\mathbf{S}$ matrix we experimented with.

The results we obtained are given in Table 6 and we can make the following comments:

- Many observations are actually similar to the ones we made for the previous benchmark. Firstly, when no sparsification is applied, we obtain equivalent dendrograms

Method	$\mathbb{NN}_k$	CC	ARI	κ
Group average	788	1.000	0.991	1
	709	1.000	0.991	1
	591	0.999	0.991	1
	394	0.999	0.991	1
	197	0.954	0.742	1
	79	0.758	0.746	1
	8	-0.834	**1.000**	5
Mcquitty	788	1.000	0.706	1
	709	1.000	0.706	1
	591	1.000	0.706	1
	394	0.998	0.706	1
	197	0.865	0.702	1
	79	0.811	0.675	1
	8	-0.697	0.760	5
Centroid	788	1.000	1.000	1
	709	1.000	**1.000**	1
	591	0.999	**1.000**	1
	394	0.994	**1.000**	1
	197	0.938	0.795	1
	79	0.346	0.815	1
	8	-0.818	0.804	5
Median	788	1.000	0.996	1
	709	1.000	0.996	1
	591	0.998	0.996	1
	394	0.994	0.996	1
	197	0.766	0.621	1
	79	0.450	0.415	1
	8	-0.694	0.798	5
Ward	788	1.000	0.688	1
	709	1.000	0.688	1
	591	0.977	0.688	1
	394	0.998	0.688	1
	197	0.986	0.679	1
	79	0.825	0.562	1
	8	-0.804	0.965	5
W-Median	788	NA	0.780	1
	709	NA	0.780	1
	591	NA	0.780	1
	394	NA	0.780	1
	197	NA	0.690	1
	79	NA	0.664	1
	8	NA	0.590	5

Table 5: Results for Aggregation data set using a Gaussian kernel.

Method	θ	CC	ARI	κ
Group average	0.010	1.000	0.811	1
	0.143	1.000	0.811	1
	0.245	1.000	0.811	1
	0.463	0.999	0.811	1
	0.819	0.947	0.802	1
	0.948	-0.766	0.818	3
	0.996	-0.741	**0.906**	99
Mcquitty	0.010	1.000	0.776	1
	0.143	1.000	0.776	1
	0.245	1.000	0.776	1
	0.463	0.998	0.776	1
	0.819	0.908	0.793	1
	0.948	-0.697	0.808	3
	0.996	-0.718	**0.906**	99
Centroid	0.010	1.000	0.812	1
	0.143	1.000	0.812	1
	0.245	0.999	0.812	1
	0.463	0.999	0.812	1
	0.819	0.836	0.785	1
	0.948	-0.800	0.747	3
	0.996	-0.753	**0.906**	99
Median	0.010	1.000	0.764	1
	0.143	0.981	0.764	1
	0.245	0.987	0.764	1
	0.463	0.997	0.764	1
	0.819	0.746	0.374	1
	0.948	-0.699	0.746	3
	0.996	-0.733	**0.906**	99
Ward	0.010	1.000	0.501	1
	0.143	1.000	0.501	1
	0.245	1.000	0.501	1
	0.463	1.000	0.501	1
	0.819	0.986	0.615	1
	0.948	-0.744	0.440	3
	0.996	-0.628	**0.906**	99
W-Median	0.010	NA	0.547	1
	0.143	NA	0.547	1
	0.245	NA	0.547	1
	0.463	NA	0.547	1
	0.819	NA	0.547	1
	0.948	NA	0.561	3
	0.996	NA	**0.906**	99

Table 6: Results for Compound data set using a Gaussian kernel.

between D-AHC and SNK-AHC. Secondly, an **S** matrix that was reduced by half of its original size, provides the same clustering quality than the dense **S** matrix. This is true for all techniques. As a consequence, it is possible to divide the computational costs by 2 without degrading the ARI values. Still, when 75% and 90% of similarity values are discarded, we do not obtain consistent improvements. Depending on the scheme, these particular thresholding levels do not always produce better ARI values.

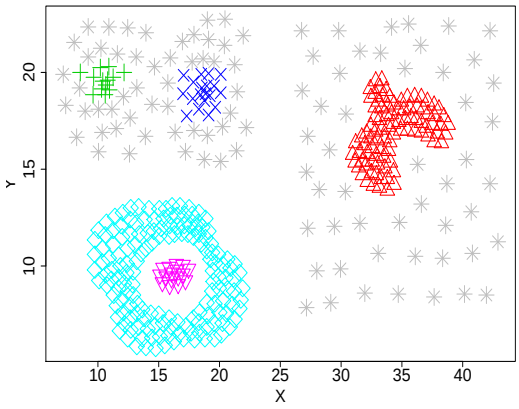

Figure 6: Compound data set results (clusters with size ≤ 3 are all represented by star symbols).

- Considering good performances, we underline the results obtained by the sparsest similarity matrix which only contains the 1% highest similarity values. The ARI value we obtain in this setting reaches 0.906 for all techniques which represents the best overall performance. All methods provided the same partition with 99 clusters. We provide in Figure 6 an illustration of the SNK-AHC outputs. For clarity reasons, we use the same star symbol to represent elements of clusters whose size is lower or equal to three. Note that among the 99 clusters, there are 89 singletons, 3 clusters of size two and 2 clusters of size three. We consider these groups as noise. Consequently, there are 5 "real" clusters that SNK-AHC was able to discover as depicted in Figure 6. Although the high ARI score of 0.906 is partly due to the fact that the SNK-AHC output is a partition with a number of clusters much larger than the ground-truth ($\kappa = 99$ *versus* $\kappa^* = 6$), it is important to precise that this result is a perfect sub-partition of the correct result.

- Finally, we emphasize the fact that SNK-AHC with the sparsest similarity matrix allows us to improve AHC from many viewpoints. Firstly, as we exposed previously, this case gives the best ARI scores and thus shows improvements in comparison with the baselines. The greatest refinement is again observed for the Ward technique whose ARI value increases from 0.501 to 0.906. Secondly, the computational costs of AHC are largely diminished. Last but not least, SNK-AHC was able to detect the core of five correct clusters which have diverse shapes and which were successfully separated from very small groups considered as noise.

6.3 Real-world Data

After having exemplified interesting properties of SNK-AHC on synthetic data sets, we address real-world clustering problems.

In this case, in addition to clustering quality, we discuss in more details the gains that SNK-AHC allow us to achieve in terms of scalability.

Likewise the previous set of experiments, the data matrix was centered and scaled before determining the inner-product matrices.

6.3.1 The Landsat Data Set

The first collection is called the landsat data set[13] which consists of 6,435 items. Each data unit corresponds to a set of 9 contiguous pixels disposed in a 3×3 patch. Each pixel is represented by its four spectral band values which are integer from 0 to 255. Consequently, the objects are described in a vectorial space of 36 dimensions. The task consists in recognizing the nature of an item among six different classes which are: red soil, cotton crop, grey soil, damp grey soil, soil with vegetation stubble, very damp grey soil.

Preliminary experimental results showed that the sparsification based on nearest neighbors provided better clustering results in comparison with the threshold based method. The Gaussian kernel also led to better performances as compared to the linear kernel. Consequently, we report the scores we obtained with these two best performing settings. Concerning $\mathbb{NN}_k$, the sequence of k values used in (27) was set to (the nearest integer of) $\{100, 90, 75, 50, 25, 10\}$ percent of n. As previously, these percentages give an estimation of the density of the $\mathbf{S}$ matrix.

The results we obtained are depicted in Figure 7. Several assessment measures are plotted. ARI and CC curves are represented by dotted lines with triangles and circles respectively. Moreover, the relative memory use and relative running time with respect to the computational costs of the baseline (dense $\mathbf{S}$) are represented. They correspond to solid lines with plus signs and dashed lines with cross signs respectively.

Below are the interesting outcomes we report from this set of experiments:

- On the scalability side, we verify that the sparser the SNK matrix, the lower the memory cost and the processing time since the curves of relative measurements of the two latter criteria clearly decrease. If z is the number of non-null entries in $\mathbf{S}$ then the relative time reduction is linear with respect to this latter variable. In other words, if we reduce $\mathbf{S}$ to 10% of its original memory size then the running time of SNK-AHC will also be reduced to 10% of its initial processing time. This result empirically illustrates Proposition 15.

- On the quality side, we can make the following comments. Firstly, if $k \geq round(n/2)$, the ARI values are not impacted very much whatever the AHC scheme. Likewise the previous benchmarks, we can save nearly half of the initial memory usage and running time without hurting the clustering quality. On the contrary, for the median and w-median approaches, the ARI values are even better. However, when $k < round(n/2)$, the clustering quality is not stable in general. Nevertheless, in the particular cases of group average and Mcquitty, the sparsest similarity matrix given by $k = round(n/10)$ provides the best performances for these techniques. The greatest gain concerns the group average scheme with an ARI score that is more than doubled since it jumped from 0.321 to 0.688.

13. `https://archive.ics.uci.edu/ml/datasets/Statlog+(Landsat+Satellite)`

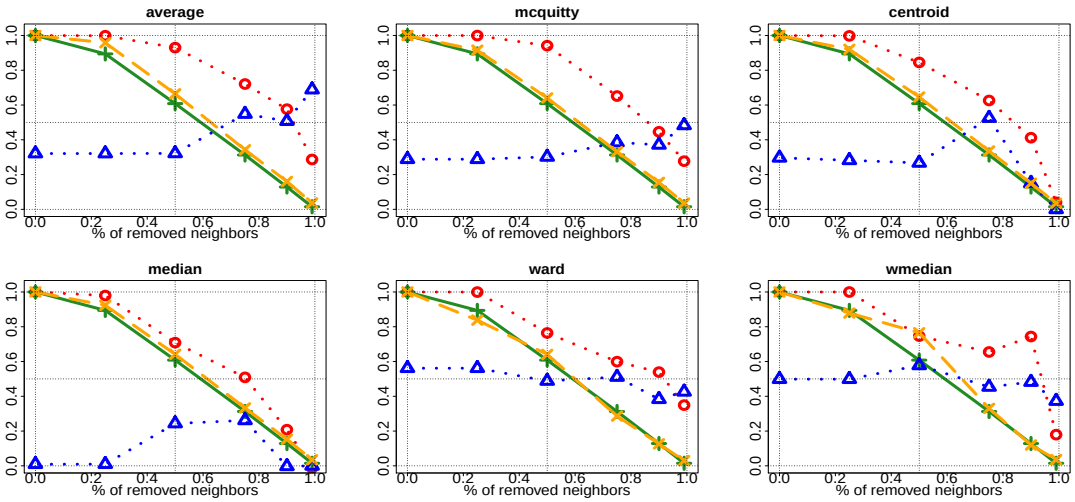

Figure 7: Results for the landsat data set using a Gaussian kernel. The x-axis corresponds to the % of removed neighbors. The y-axis corresponds to the observed values which all belong to $[0, 1]$. Solid lines with plus signs represent the relative memory use, dashed lines with cross signs show the relative running time, dotted lines with circles indicates the CC values, dotted lines with triangles give the ARI values.

- Regarding the new method w-median, it is worth mentioning that it plainly dominates the median scheme. It appears that the non-uniform weight we add to the median technique not only allows obtaining monotonic dendrograms, but it also enables boosting the clustering quality scores.

Note that all similarity graphs are connected even in the case of the sparsest similarity matrix. Thereby, whatever the setting, SNK-AHC always gave a single tree.

6.3.2 The Pendigits Data Set

The second collection we used, is called the pendigits data set[14]. This benchmark consists of handwritten digits that were collected from 44 different writers. Each one of them provided around 250 samples so that the entire collection is composed of 10,992 observations. Each sample is described by 16 numerical features. The 10 digits have equal frequency. Obviously, the task is to recognize groups of elements that correspond to the same digits.

In this case, we report the results we obtained with a linear kernel since they provided similar outcomes than a Gaussian kernel. However, the nearest neighbor sparsification outperformed the one relying on a threshold. Therefore, we report in Figure 8 the scores obtained with this former sparsification technique. We use the same sequence of neighborhood selection as before.

The results we obtain for this benchmark are pretty similar to the landsat case:

14. `https://archive.ics.uci.edu/ml/datasets/Pen-Based+Recognition+of+Handwritten+Digits`

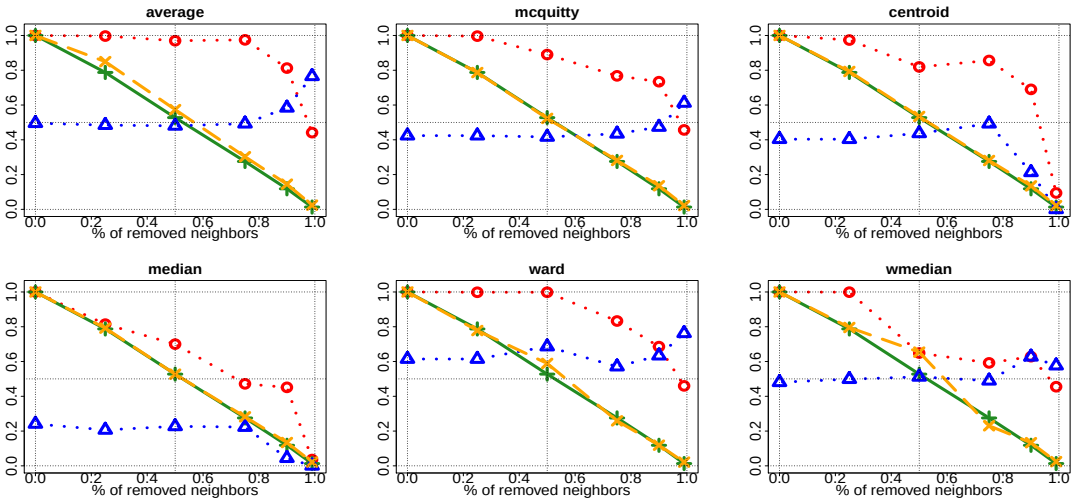

Figure 8: Results for the pendigits data set using a linear kernel. The x-axis corresponds to the % of removed neighbors. The y-axis corresponds to the observed values which all belong to $[0, 1]$. Solid lines with plus signs represent the relative memory use, dashed lines with cross signs show the relative running time, dotted lines with circles indicate the CC values, dotted lines with triangles give the ARI values.

- For all schemes the ARI curves are stable up to 75% of removed neighbors. In regard to scalability, this means that, for any technique, we can save up to ∼70% of memory usage and processing time without degrading the performances. Beyond 75% of removed neighbors, the clustering quality evolution depends on the method. For centroid and median it decreases whereas for the other approaches the ARI curves have a positive slope.

- For group average, Mcquitty and Ward, their respective best ARI scores are achieved with the sparsest $\mathbf{S}$ matrix. Precisely, if we keep only 10% of the nearest neighbors then, the ARI values observed for these techniques clearly outperform their respective baseline. Overall, the best gain and best ARI score is achieved by the group average with a clustering quality going from 0.495 to 0.765 which represents a 54% increase.

- For this data set as well, w-median is superior to median. Moreover, the ARI values we observe for the new method is pretty stable with respect to the level of sparsification.

Similarly to the landsat case, the similarity graph remained connected even with the strongest sparsification we applied.

7. Related Work and Discussion

Our approach is meant to be generic, scalable and effective with respect to challenging clustering tasks where objects belong to non-linear manifolds. Different types of previous

research works are relevant to our framework.

In order to face the inherent scalability issues of hierarchical clustering, several algorithms were introduced in the data mining community. BIRCH (Zhang et al., 1996) and CURE (Guha et al., 1998) are famous examples in that respect. These approaches use random sampling and/or a pre-clustering stage in order to reduce the number of elements to convey to the hierarchical clustering. These methods rely on a vectorial representation of the objects and use classic distances between points.

Carrying out an AHC approach on a sparse graph in the goal of speeding up the hierarchy construction was also studied in (Franti et al., 2006). In this work as well, objects are points in a vectorial space and weighted squared Euclidean distances serve as dissimilarity values. The authors use a directed k nearest neighbors graph. After each merge, the list of the k closest points to each centroid is approximately updated.

These research works provide efficient algorithms. However, they are not generic models of hierarchical clustering. In particular, they assume a feature based description of the items (stored data approach) and in this vectorial representation, the dissimilarities are all related to squared Euclidean distances. In our case, SNK-AHC relies on a generic model that allows designing different kinds of proximity relationships between clusters.

From this standpoint, our approach is in line with the works that were developed during the 1960's in the fields of statistics and data analysis. Overviews of these works can be found in (Gordon, 1987; Everitt et al., 2009; Mirkin, 1996; Murtagh and Contreras, 2012). The LW equation plays a core role in this landscape since it formally represents an infinite family of hierarchical clustering techniques. Furthermore, it makes it possible to algebraically study and define particular sub-families which satisfy different appealing conditions. The guarantee to output a monotonic dendrogram is an example of such conditions. In more general terms, these properties are named admissibility conditions (see for e.g. (Fisher and Van Ness, 1971; Chen and Van Ness, 1994, 1996; Mirkin, 1996)). More recently, Ackerman and Ben-David (2016) introduced other types of properties that characterize a class of linkage-based hierarchical clusterings.

In this context, several research papers have also addressed the scalability problems of D-AHC. We can highlight two research lines in that regard. The first one is essentially a matter of implementation and concerns the whole family of LW clusterings. In fact, by leveraging advanced data structure it is possible to speed-up D-AHC (Anderberg, 1973; Day and Edelsbrunner, 1984). By employing priority queues to efficiently store the nearest neighbors, the running time of the minimum search can be reduced. Maintaining the priority queues can also be done in an efficient fashion and overall, the time complexity of D-AHC can be reduced from cubic to a best-case $O(n^2)$ cost (Day and Edelsbrunner, 1984; Müllner et al., 2013).

In contrast to the previous research avenue, the second research line only involves a subfamily of AHC schemes. It is based on a property called reducibility which was stated by Bruynooghe (1978), and the resulting algorithm is usually named nearest neighbor chains. The reducibility condition is not satisfied by centroid and median for which the latter algorithm is not equivalent to D-AHC. For the other methods, it is an exact procedure that has a worst-case $O(n^2)$ time complexity instead of cubic. The nearest neighbor chains

procedure has been studied and implemented by several authors (de Rham, 1980; Juan, 1982; Benzécri, 1982; Murtagh, 1984; Müllner et al., 2013). More recently, Nguyen et al. (2014) has proposed a memory-efficient online hierarchical clustering called SparseHC which also relies on the reducibility property. However, only single, complete and group average linkages are considered.

These latter research works only focus on the computational efficiency of the usual D-AHC framework. SNK-AHC is able to effectively tackle a more general class of clustering problems.

Complex data such as texts, graphs, images and so on, do not necessarily lie on linear sub-spaces but rather on manifolds. Euclidean distances in the given descriptive space may fail to determine non-convex and arbitrary shapes. In order to better capture the underlying geometry of the data, other different approaches have been proposed in the literature.

In the context of non-parametric hierarchical clustering, one first group of papers, adopts a graph point of view of the clustering task. In particular, the nearest neighbors graph derived from the pairwise proximity values allows a better approximation of the natural geometries of groups of points. It is well-known that the single linkage leads to a chaining effect that is quite effective for arbitrary shapes detection as compared to other schemes. Gower and Ross (1969) pointed out the strong link between single linkage and the minimum spanning tree (MST) problem. Then Zahn (1971) analyzed more in details the application of the MST algorithm to the detection of groups that are non-convex and non-linearly separable. In this context, edge removal appears to be an effective mean to allow the MST to capture a large spectrum of shapes.

Other approaches use non-parametric proximity measures that rely on mutual nearest neighbors and rank-based linkages in order to recover arbitrary clusters shapes (Jarvis and Patrick, 1973; Gowda and Krishna, 1978). More recently, Balcan et al. (2014) uses common nearest neighbors and defines a two-step hierarchical clustering that is robust to outliers and which has interesting properties under some good neighborhood conditions.

Another related work in this context is the CHAMELEON algorithm introduced in (Karypis et al., 1999). It also proceeds in two stages and uses k nearest neighbors graphs. CHAMELEON presents another common point with SNK-AHC since it emphasizes the contrast between inter and intra connectivities of clusters. This so-called dynamic modeling is similar in spirit to our penalized similarities. However, the authors do not propose a generic model unlike our parametric recurrence equations (12a) and (12b).

Yet another research direction for manifold learning is through the use of kernel functions that map the data points from the original description space to a higher dimensional Hilbert space, called the feature space (see for e.g. (Lee and Verleysen, 2007)). It is expected that in the new space, the clusters are easier to detect. To our knowledge, only a few papers have extended D-AHC to kernels (Qin et al., 2003; Endo et al., 2004). In contrast to our model, these papers do not consider the scalability issues. Our previous work (Ah-Pine and Wang, 2016) also studies an inner-product based formulation of the LW equation. In addition, sparse kernel matrices are also employed. Nonetheless, this latter model is different from the framework we present in this paper. In particular, the concept of weighted penalized similarities and the theoretical results we provide are not examined in the framework introduced in (Ah-Pine and Wang, 2016).

Lastly, it is worth emphasizing the relationships between SNK-AHC and spectral clustering (Shi and Malik, 2000; Ng et al., 2001; Von Luxburg, 2007). In the latter family of techniques, kernel functions are employed to construct a similarity graph between objects. Then a sparsification method is applied and the Laplacian of the resulting graph is determined. Theoretical results from spectral graph theory (see for e.g. (Van Mieghem, 2010)) show the links between the eigen-decomposition of the Laplacian and the connected components of the graph. Spectral clustering is a two step procedure which performs a spectral embedding of the objects and subsequently applies a flat clustering method in the new space. The k-means algorithm is usually used in the second step. In this context, roughly speaking, we believe that SNK-AHC is to the classic D-AHC what spectral clustering is to the usual k-means: a significant extension of a conventional clustering method (a sub-family of LW clusterings in our case) which can recover groups of points with non-spherical shapes and which provide an interesting mean to guess the number of clusters. Besides, our approach has all the advantages that hierarchical clustering has over partitional clustering. In addition, since SNK-AHC is much more scalable than D-AHC, it does not have the major drawbacks of hierarchical clustering methods.

8. Conclusion and Future Work

We have introduced K-AHC a generic AHC model which relies on inner-products instead of squared Euclidean distances. Our approach is based on two recurrence formulas which embeds a sub-family of LW clustering techniques. In order to make our model efficient and effective for challenging clustering tasks, we apply K-AHC on a sparsified normalized kernel matrix. In that perspective, the two recurrence formulas highlight aggregation of inter-similarities on the one hand and of intra-similarities on the other hand. Our work can be viewed as a dynamic modeling of weighted penalized similarities of clusters. Moreover, by constraining the bottom-up merging procedure to only fuse pairs of clusters whose inter-similarity value is non-null, our method, SNK-AHC, not only is more scalable than the usual D-AHC, but it is also able to boost the clustering quality and to detect the number of clusters.

However, the performances that SNK-AHC can reach, depend on the way the similarity matrix is sparsified. Note that this is also the case for any method that relies on sparse similarity graphs such as spectral clustering. Therefore, one important future line of research is the study and design of more advanced sparsification techniques. From a clustering quality standpoint, the setting of the sparsification method is an important question to address in practice since it determines the connected components of the SNK matrix and thus the number of clusters our approach will recover. In order to investigate this problem from a theoretical point of view, the cluster tree framework introduced in (Chaudhuri and Dasgupta, 2010) and (Balakrishnan et al., 2013) could be of interest. Regarding the overall complexity of Algorithm 4, techniques that make it possible to exactly or approximately determine nearest neighbors graphs in an efficient manner are important to look at. Indeed, even though the dendrogram building procedure performed by SNK-AHC can be carried out efficiently, the basic computational cost for determining the k nearest neighbors graph remains quadratic and can be a bottleneck in practice.

Still, it is interesting to mention, that, as far as the tree building procedure is concerned, there are already pretty immediate ways to further improve the scalability of our approach. As underlined in the previous section, two directions could be considered. Firstly, we can enhance the complexity of SNK-AHC by using priority queues. Secondly, we can use the nearest neighbor chains approach. In that regard, note that the best performances we observed in our experiments concern schemes that satisfy the reducibility condition.

Finally, our model is generic but we have demonstrated that not all parameters settings are worth considering. From this standpoint, it is interesting to examine how other admissibility conditions are expressed in our framework. In that perspective, a new property which is peculiar to our work is the diagonal translation invariance. We proved that group average, Mcquitty and Ward satisfy this condition. These techniques are among the most effective ones from the experimental results we reported. Accordingly, a characterization of this sub-family of clustering techniques would be beneficial.

Acknowledgments

The author would like to thank the anonymous reviewers for their valuable comments.

References

Margareta Ackerman and Shai Ben-David. A characterization of linkage-based hierarchical clustering. *Journal of Machine Learning Research*, 17:1–17, 2016.

Julien Ah-Pine. Normalized kernels as similarity indices. In *Advances in Knowledge Discovery and Data Mining, 14th Pacific-Asia Conference, PAKDD 2010, Hyderabad, India, June 21-24, 2010. Proceedings. Part II*, pages 362–373, 2010.

Julien Ah-Pine and Xinyu Wang. Similarity based hierarchical clustering with an application to text collections. In *International Symposium on Intelligent Data Analysis*, pages 320–331. Springer, 2016.

Michael R Anderberg. *Cluster analysis for applications*. Academic Press, New York, 1973.

Sivaraman Balakrishnan, Srivatsan Narayanan, Alessandro Rinaldo, Aarti Singh, and Larry Wasserman. Cluster trees on manifolds. In *Advances in Neural Information Processing Systems*, pages 2679–2687, 2013.

Maria-Florina Balcan, Yingyu Liang, and Pramod Gupta. Robust hierarchical clustering. *The Journal of Machine Learning Research*, 15(1):3831–3871, 2014.

Jean-Paul Benzécri. Construction d'une classification ascendante hiérarchique par la recherche en chaîne des voisins réciproques. *Les cahiers de l'analyse des données*, 7 (2):209–219, 1982.

M. Bruynooghe. Classification ascendante hiérarchique des grands ensembles de données : un algorithme rapide fondé sur la construction des voisinages réductibles. *Cahiers de l'analyse des données*, 3(1):7–33, 1978. URL `http://eudml.org/doc/87905`.

Chih-Chung Chang and Chih-Jen Lin. Libsvm: a library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3):27, 2011.

Kamalika Chaudhuri and Sanjoy Dasgupta. Rates of convergence for the cluster tree. In *Advances in Neural Information Processing Systems*, pages 343–351, 2010.

Zhenmin Chen and John W. Van Ness. Space-contracting, space-dilating, and positive admissible clustering algorithms. *Pattern recognition*, 27(6):853–857, 1994.

Zhenmin Chen and John W Van Ness. Space-conserving agglomerative algorithms. *Journal of classification*, 13(1):157–168, 1996.

Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms, Third Edition*. The MIT Press, 3rd edition, 2009. ISBN 0262033844, 9780262033848.

William HE Day and Herbert Edelsbrunner. Efficient algorithms for agglomerative hierarchical clustering methods. *Journal of classification*, 1(1):7–24, 1984.

C. de Rham. La classification hiérarchique ascendante selon la méthode des voisins réciproques. *Les cahiers de l'analyse des données*, 5(2):135–144, 1980.

Daniel Defays. An efficient algorithm for a complete link method. *The Computer Journal*, 20(4):364–366, 1977.

Yasunori Endo, Hideyuki Haruyama, and Takayoshi Okubo. On some hierarchical clustering algorithms using kernel functions. In *IEEE International Conference on Fuzzy Systems, FUZZ-IEEE 2004, Budapest, Hungary, July 25-29, 2004.*, pages 1513–1518, 2004.

Brian S. Everitt, Sabine Landau, and Morven Leese. *Cluster Analysis*. Wiley Publishing, 4th edition, 2009. ISBN 0340761199, 9780340761199.

Lloyd Fisher and John W. Van Ness. Admissible clustering procedures. *Biometrika*, 58(1): 91–104, 1971.

Pasi Franti and et al. Clustering datasets, 2015. URL `http://cs.uef.fi/sipu/datasets/`.

Pasi Franti, Olli Virmajoki, and Ville Hautamaki. Fast agglomerative clustering using a k-nearest neighbor graph. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(11):1875–1881, November 2006. ISSN 0162-8828. URL `http://dx.doi.org/10.1109/TPAMI.2006.227`.

Aristides Gionis, Heikki Mannila, and Panayiotis Tsaparas. Clustering aggregation. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):4, 2007.

Allan D Gordon. A review of hierarchical classification. *Journal of the Royal Statistical Society. Series A (General)*, pages 119–137, 1987.

K. Chidananda Gowda and G. Krishna. Agglomerative clustering using the concept of mutual nearest neighbourhood. *Pattern recognition*, 10(2):105–112, 1978.

John C. Gower and Gavin J.S. Ross. Minimum spanning trees and single linkage cluster analysis. *Applied statistics*, pages 54–64, 1969.

Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. Cure: an efficient clustering algorithm for large databases. In *ACM Sigmod Record*, volume 27, pages 73–84. ACM, 1998.

Roger A. Horn and Charles R. Johnson, editors. *Matrix Analysis*. Cambridge University Press, New York, NY, USA, 1986. ISBN 0-521-30586-1.

Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of Classification*, 2 (1):193–218, 1985. ISSN 1432-1343. doi: 10.1007/BF01908075. URL `http://dx.doi.org/10.1007/BF01908075`.

Raymond Austin Jarvis and Edward A Patrick. Clustering using a similarity measure based on shared near neighbors. *IEEE Transactions on computers*, 100(11):1025–1034, 1973.

J. Juan. Programme de classification hiérarchique par l'algorithme de la recherche en chaîne des voisins réciproques. *Les cahiers de l'analyse des données*, 7(2):219–225, 1982.

George Karypis, Eui-Hong Han, and Vipin Kumar. Chameleon: Hierarchical clustering using dynamic modeling. *Computer*, 32(8):68–75, 1999.

Godfrey N. Lance and Williams T. Williams. A general theory of classificatory sorting strategies: 1. hierarchical systems. *The Computer Journal*, 9(4):373–380, 1967.

John A Lee and Michel Verleysen. *Nonlinear dimensionality reduction*. Springer Science & Business Media, 2007.

Moshe Lichman. UCI machine learning repository, 2013. URL `http://archive.ics.uci.edu/ml`.

Glenn W Milligan. Ultrametric hierarchical clustering algorithms. *Psychometrika*, 44(3): 343–346, 1979.

Boris Mirkin. *Mathematical Classification and Clustering*. Kluwer Academic Publishers, London, 1996.

Daniel Müllner. Modern hierarchical, agglomerative clustering algorithms. *CoRR*, abs/1109.2378, 2011. URL `http://arxiv.org/abs/1109.2378`.

Daniel Müllner et al. fastcluster: Fast hierarchical, agglomerative clustering routines for r and python. *Journal of Statistical Software*, 53(9):1–18, 2013.

Fionn Murtagh. Complexities of hierarchic clustering algorithms: state of the art. *Computational Statistics Quarterly*, 1(2):101–113, 1984.

Fionn Murtagh and Pedro Contreras. Algorithms for hierarchical clustering: an overview. *Wiley Interdisc. Rew.: Data Mining and Knowledge Discovery*, 2(1):86–97, 2012.

Andrew Y Ng, Michael I Jordan, Yair Weiss, et al. On spectral clustering: Analysis and an algorithm. In *NIPS*, volume 14, pages 849–856, 2001.

Thuy-Diem Nguyen, Bertil Schmidt, and Chee-Keong Kwoh. SparseHC: a memory-efficient online hierarchical clustering algorithm. *Procedia Computer Science*, 29:8–19, 2014.

Jie Qin, Darrin P Lewis, and William Stafford Noble. Kernel hierarchical gene clustering from microarray expression data. *Bioinformatics*, 19(16):2097–2104, 2003.

Bernhard Scholkopf and Alexander J Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2001.

Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000.

Robin Sibson. SLINK: an optimally efficient algorithm for the single-link cluster method. *Comput. J.*, 16(1):30–34, 1973. doi: 10.1093/comjnl/16.1.30. URL `http://dx.doi.org/10.1093/comjnl/16.1.30`.

Piet Van Mieghem. *Graph spectra for complex networks*. Cambridge University Press, 2010.

Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4): 395–416, 2007.

Charles T. Zahn. Graph-theoretical methods for detecting and describing gestalt clusters. *IEEE Transactions on computers*, 100(1):68–86, 1971.

Tian Zhang, Raghu Ramakrishnan, and Miron Livny. Birch: an efficient data clustering method for very large databases. In *ACM Sigmod Record*, volume 25, pages 103–114. ACM, 1996.