A General Hybrid Clustering Technique

Saeid Amiri*
Department of Civil, Polytechnique Montréal, QC, Canada
Bertrand S. Clarke
Jennifer L. Clarke
Department of Statistics, University of Nebraska-Lincoln
Hoyt Koepke
Apple Inc., Cupertino, CA

July 8, 2018

Abstract

Here, we propose a clustering technique for general clustering problems that have non-convex clusters. For a given desired number of clusters K, we use three stages to find clusters. The first stage uses a hybrid clustering technique to produce a series of clusterings of various sizes (randomly selected). The key steps are to find a K-means clustering using K_ℓ clusters where $K_\ell \gg K$ and then join these small clusters by using single linkage clustering. The second stage stabilizes the result of stage one by reclustering via the 'membership matrix' under Hamming distance to generate a dendrogram. The third stage is to cut the dendrogram to get K^* clusters where $K^* \geq K$ and then prune back to K to give a final clustering. A variant on our technique also gives a reasonable estimate for K_T, the true number of clusters.

We provide a series of arguments to justify the steps in the stages of our methods and we provide examples involving real and simulated data to compare our technique with other techniques.

Keywords: Consistency, K-means, lifetimes, outliers, single linkage.

*The authors gratefully acknowledge Holland Computing Center for invaluable computational support.

1 Introduction

Clustering is an unsupervised technique used to find underlying structure in a dataset by grouping data points into subsets that are as homogeneous as possible. Clustering has many applications in a wide range of fields. No list of references can be complete, however, three important recent references are **?**, **?**, and **?**.

There are numerous clustering proceures and they can be grouped into many classes. Because the scope of clustering problems is so big, all of these procedures have limitations. The justification for the new method presented here is that it combines two classes of methods (centroid-based and agglomerative hierarchical) with a careful treatment of influential data points and (i) is not limited to convex clusters or (ii) as dependent on subjective choices of quantities such as dissimilarities. That is, we combine several clustering techniques and principles in sequence so that one part of the technique may correct weaknesses in other parts giving an overall improvement.

A further benefit of our clustering method is that we can prove a theorem ensuring that the basal sets (defined below) cover the regions in a clustering problem, in the limit of large sample size and use this to establish a corollary ensuring that the final clustering from our method, at least in simple cases, will be correct. We also give formal results ensuring that the conditions of our main theorem can be satisfied in some simple but general cases. To the best of our knowledge, there are no fully nonparametric techniques, except for K-means, for which a consistency result such as ours can be established. Among parametric clustering methods, only model based clustering can be shown to be consistent assuming (i) the E-M algorithm converges to the correct weights and (ii) the parametric models in the mixture are correct. Spectral clustering, discussed below, has a consistency property but it is not of the clustering per se, only the features that lead to the clustering.

To fix notation, we assume n independent and identical (IID) outcomes $x_i,\ i = 1, \ldots, n$ of a random variable X. The x_i's are assumed m-dimensional and written as $(x_{i1}, \ldots, x_{im})$ when needed. We denote a clustering of size K by $\mathcal{C}(K) = (C_{K1}, \ldots, C_{KK})$; we assume that for each K only one clustering will be generated and the $C_{K,j}$'s form a partition of $\mathbb{R}^m$.

For a given K, we start by choosing a (large) number K_ℓ and drawing a random K_b,

$b = 1, \ldots, B$ from a distribution that ensures a variety of reasonable clustering sizes will be searched. Then, our method has three generic steps: 1) *Generate clusterings*: Create K_ℓ clusters by K-means, hereafter K-m. Then use single linkage (SL) clustering to take unions of the K_ℓ clusters to get clusterings of size K_b; 2) *Stabilization*: Repeat Step one B times; the result is B clusterings with sizes $K_1, \ldots, K_B$. From these clusterings, define a membership matrix M (see below) that gives a dissimilarity using Hamming distance so SL can be applied. 3) *Choose a final clustering*: Use a 'grow-and-prune' approach on the dendrogram from Stage 2. Cut the dendrogram at some dissimilarity value smaller than H_K, the value of the dissimilarity that gives K clusters. The resulting $K^* \geq K$ clusters are then merged to form K clusters.

In fact, we develop two versions of this procedure. One is the 'pure' version and the other is stabilized by using percentiles of distances so the influence of potential outliers is reduced. We found that the 20-th percentile worked well and it is this second one that we recommend more generally.

We compare our two clustering methods to seven other clustering methods. K-m is probably the most familiar method so we have included it even though it tends not to perform well for nonconvex clusters unless they are well separated.

There are many precedents for the kind of hybrid clustering described in stages 1) and 2) that combine two or more distinct clustering techniques. Perhaps the closest is **?**. Their central idea is to create many clusterings of different sizes (by K-m) that can be pooled via a 'co-association matrix' that weights points in each clustering according to their membership. This matrix can then be modified to give a dissimilarity so that single-linkage clustering can be used to give a final clustering. **?** refer to this as evidence accumulation clustering (EAC) because they are pooling information over a range of clusterings. EAC differs meaningfully from our technique in four ways. First, in our technique we choose a single K_ℓ while EAC uses a range of cluster sizes. Second, we ensemble (and hence stabilize) directly by membership in terms of Hamming distance whereas EAC ensembles by a co-association. Third, our procedure uses an extra step of growing and pruning a dendrogram (see Step 5 in Algorithm #1) that is akin to an optimization over 'main' clusters and therefore handles outliers better. Fourth, by taking advantage of the chaining property of single linkage, our

method handles non-convex clusters better than the ? method. Thus, our techniques and EAC both use K-m and single-linkage, i.e., are hybrid techniques, and can find nonconvex clusters, but they differ in many of the details of the way these techniques are combined. Overall, our 'fine-tuning' of their technique simplifies it and seems to give better results.

Third, a technique that is conceptually similar to ours is due to ?, hereafter CT. First, in a 'bottom-up stage', small sets of points that are not to be separated are replaced by their centroids. Then, in a 'top-down stage', the remaining points are clustered divisively to give big clusters. Then, the bottom up and top-down stages are reconciled to give a final clustering. Our proposed technique differs from CT in four key ways. First, we use K-m in place of CT's 'mutual clusters'. Second, we use single linkage where CT uses average linkage. Third, our technique has a stabilization stage. Fourth, our technique uses a 'grow and prune' strategy, unlike CT. So, it is unclear how well CT performs when the true clusters are non-convex.

Fourth, a technique, conceptually related to ours but nevertheless very different, is due to ?. This technique, CHAMELEON (CHA), rests on a graph theoretic analysis of the clustering problem and uses two passes over the data. The first is a graph partitioning based algorithm to divide the data set into a collection of small clusters. The second pass is an agglomerative hierarchical clustering based on connectivity (a graph-theoretic concept) to combine these clusters. Our method differs from ? in four key ways. First, we use K-m instead of graph partitioning. Second, we simply use single linkage whereas ? combines small clusters based on both closeness and relative interconnectivity. Third, our technique has a stabilization stage to manage cluster boundary uncertainty. Fourth, our technique explicitly uses a 'grow and prune' strategy permitting a 'look ahead' to more clusters than necessary. By contrast, CHA has an elaborate optimization. On the other hand, both can find non-convex clusters.

Fifth, we include spectral clustering since i) it is a qualitatively different clustering approach (based as it is on the graph Laplacian) and ii) it has consistency properties, namely the eigenvectors of the graph Laplacian converge to limiting values, see ?. This is not the same as consistency of the clustering itself, but it is suggestive of clustering cnsistency. There are many forms of spectral clustering and we simply use the default

version in the R package kernlab.

Sixth, we also consider a robustified form of clustering called trimmed clustering (TC) and implemented by the contributed R-package TCLUST, see **?**. The central idea is that the true clustering corresponds to a collection of normal distributions contaminated by outliers. The methodology is the result of a complicated optimization problem, see **?** Sec. 2. Overall, the methodology is based on K-m but treats outlier and cluster spreads more carefully. Indeed, the methodology is, loosely, a variant on model-based clustering in the sense that there is a likelihood and if the components are allowed to have non-convex level sets then TCLUST should be able to capture non-convex clusterings. In addition, the proportion of outliers is controlled by a parameter, not unlike our use of α in Alg. #1 below. TCLUST seems to achieve this by weighting and scaling so as to allow heterogeneous clusters, i.e., unequal covariance matrices for the clusters. However, we give an example (SCALES) where TC is unable to accommodate the difference in scales along different directions. Overall, where sensitivity to the spread of clusters and outliers is not required, which is often the case, our proposed methods are easier and generally perform better.

The seventh and final existing method we included in our comparison is fuzzy clustering, FC. One of the most recent lucid reviews of it can be found in **?**; see also the references therein. Also, see **?** for details of implementation. The version we use here is not sophisticated e.g., it is based on K-m (see **?**) and there is no noise cluster or regularization. However, this method does allow for a soft clustering i.e., a data point is assigned only a probability of being in a cluster. This corresponds to a more complex optimization with a more complex algorithm; see **?** for a concise summary. The package we use, FCLUST, frequently gives clusters that are quite different from K-m and other methods despite its apparent procedural similarity to some of them. It does not readily generate non-convex clusters but the pattern of soft clustering membership values can indicate a convex clustering that may be effective.

To the best of our knowledge, the earliest explicit proposal for hybrid methods is in **?** who conjectured that using K-m with K too large and single linkage might enable a technique to find nonconvex clusters.

In addition to comparisons with other clustering methods, we also pre-process the data

to which we apply our method by dimension reduction. Specifically, we use t-SNE ref? to reduce the dimension to 2 or 3 and then apply our clusteing methods. We see that ********** (this is probably a bad idea)

In addition to proposing two forms of a new hybrid clustering technique (see Algorithm #1 and Subsec 3.2) we present a way to estimate the correct value K_T of K in Algorithm #2. Essentially, we combine the first three steps of Algorithm #1 with a modification of **?**. It is important to note that EAC, like our method, can also be used to estimate the number of clusters in a clustering. The Gap statistic, Silhouette distance, and Bayes information criterion can also be used for this purpose. Thus, we compare the accuracy of six methods for estimating K_T for a series of data sets. Even though the data sets cannot be regarded as outcomes of a single stochastic process, evaluating the performance of the six methods on them gives a useful indicator of performance.

The rest of this paper is organized as follows. In Sec. 2 we present our two algorithms for clustering and estimating K_T. In Sec. 3 we provide justifications for some of the steps in our algorithms. For the steps where we are unable to provide theory, we provide methodological interpretations as a motivation for their use. In Sec. 4 we compare the nine methods discussed here on two simulated data. Both examples are meant to illustrate a qualitative feature of clusters that, if known, could influence the choice of method; the results show our method is either the best among those we considered or nearly the best. In Sec. 5 we compare all nine methods on benchmark data sets. Our concluding remarks are in Sec. 6 and some technical details are in the Appendix.

2 Presentation of techniques

We begin with Algorithm #1 that formalizes our generation of clusterings. It has five steps and five inputs: the number K of clusters to be in the final clustering, a number K_{max} to be the largest number of clusters that we would consider reasonable, a number B of iterations of our initial hybrid clustering technique, a large number $K_\ell \gg K$ of clusters whose points will be merged into larger clusters, and a value α to serve as a cutoff for the size of a cluster as measured by the proportion of how many of the K_ℓ clusters had to be combined to create it. In practice, setting $K_\ell = \lfloor n/5 \rfloor$ worked reasonably well; however,

$\lfloor n/5 \rfloor$ is an arbitrary choice and we found that adding a layer of variability by choosing K_ℓ according to a $DUnif[\lfloor n/4 \rfloor, \lfloor n/6 \rfloor]$ gave improved results. Separately, we also found that larger values of K_{max} seemed to require larger values of B to get good results. We address the choice of B and K_{max} later in Secs. 4 and 5. In our work here, we merely set $\alpha = .05$. This ensured that we got at least K clusters in our examples. Loosely, the more outliers or small clusters there are, the smaller one should choose α. The most important specification is often K; we treat this in Algorithm #2.

We begin with our clustering Algorithm #1 referring to it as SHC.

Note that the number of clusters K^* from the 'growth' part of a grow and prune strategy is defined internally to the algorithm in Step 4. The idea is to get a dendrogram with slightly more than K clusters so the algorithm searches ahead for good clusters. In Step 5, any extra clusters that are found, but not helpful, are pruned away. This strategy formalizes the idea that, for SL, when chaining occurs we should use it to help find points that belong in a cluster but stop when the chaining starts including points that are more representative of another cluster. That is, the definition of K^* is a formalization the the concept of optimal the depth down a dendrogram it is worth going by SL taking advantage of its chaining property. Going further down the dendrogram would mean that the advantages of chaining are being lost: It is no longer reasonable to regard the extra data points in the 'chain' as part of the cluster being agglomerated.

Algorithm #1 can serve as the basis for another algorithm to estimate K_T. We add an extra step derived from the method for choosing K in **?**. Recall that **?** considered a set of 'lifetimes' that were lengths in terms of the dissimilarity. These were the lengths, measured on the vertical axis defined by the dissimilarity, between the values at which one could cut a dendrogram so as to get a collection of clusters with the property that at least one of the clusters emerges precisely at the value on the vertical axis at which the horizontal line was drawn. **?** then cut the dendrogram at the dissimilarity corresponding to the maximum of these vertical distances to choose the number of clusters. Algorithm #2 extends this method by using it once, removing some clusters, and then using it again.

Our general procedure is given in Algorithm #2 and we refer to it as EK. EK differs from the method for estimating K_T in **?** Secs. 4 and 5 in four ways. First, after they generate

Algorithm 1: Stabilized Hybrid Clustering (SHC)

1 Given K, start by drawing a value of K_ℓ and then drawing a value of $K_b \sim DUnif(2, K_{max})$ where $K_{max} < K_\ell$, for $b = 1, \ldots, B$. For each K_b, use randomly generated initial conditions to obtain $\mathcal{C}(K_b) = \{C_{b1}, \ldots, C_{bK_b}\}$: Use K-m clustering to generate a clustering of size K_ℓ 'basal' clusters and then use single linkage clustering to form $\mathcal{C}_{K_b}$ by merging the K_ℓ basal clusters.

2 For $\mathcal{C}(K_1)$, let $M_1 = (\chi(s,t))_{s=1,\ldots,n;t=1,\ldots,K_1}$ be the $n \times K_1$ membership matrix with entries $\chi(s,t) = \begin{cases} 1 & x_s \in C_{K_1,t} \\ 0 & x_s \notin C_{K_1,t}. \end{cases}$ Doing the same for the rest of the $\mathcal{C}(K_b)$'s generates membership matrices $M_1, \ldots, M_B$ for clusterings $\mathcal{C}(K_2), \ldots, \mathcal{C}(K_B)$, respectively. Concatenating M_b's gives the overall membership matrix $M(B) = [M_1, \ldots, M_B]$.

3 From the $n \times \sum_b K_b$ overall membership matrix $M(B)$ we construct a dissimilarity matrix using Hamming distance. Let $S = \sum_b K_b$. That is, the i-th and j-th rows in $M(B)$ are of the form $x_i = (x_{i,1}, \ldots, x_{i,S})$ and $x_j = (x_{j,1}, \ldots, x_{j,S})$ and so give dis-similarities $d_{ij} = \sum_{m=1}^{S} \mathsf{I}(x_{im}, x_{jm})$ where $\mathsf{I}(x_{im}, x_{jm}) = 1$ if $x_{im} \neq x_{jm}$ and zero otherwise. Let $D = (d_{ij})$.

4 Given D, use SL clustering to generate a vertical dendrogram with leaves at the bottom and dissimilarity values on the y-axis. Since K is given, it corresponds to a value H_K on the y-axis, and there will be K branches on the dendrogram that cross H_K. Denote the lengths of the K lines from H_K to the next split in the dendrogram be denoted $h_1, \ldots, h_K$ with mean $\bar{h}$. Now, cut the dendrogram a little further down than H_K, namely at $H_K + \bar{h}$. Cutting at this value gives a number of clusters; denote this number by K^*.

5 Write Since $\bar{h} \geq 0$, $\exists v \geq 0$ so that $K^* = K + v$ with v a non-negative integer. If $v = 0$, the clustering from Step 4 (of size K) is the final clustering. If $v \geq 1$, write $v = v_1 + v_2$ where v_2 is the number of clusters in $\mathcal{C}_{K^*}$ for which $\#(C_{K^*j})/n \leq \alpha$ where $\alpha > 0$ is a pre-assigned tolerance. If K clusters of size at least α do not exist, adjust α downward until they do. Using SL (under the corresponding submatrix of D) recluster the points in the $(K + v_1)$ clusters to obtain K 'main' clusters. Then, use SL clustering to assign points in the remaining v_2 clusters to the K 'main' clusters.

a collection of clusterings they use a co-association matrix where we use ensembling via a dissimilarity matrix. Second, they then use an information-theoretic optimization whereas we do not invoke any optimality principle. Third, they have to stabilize their results whereas our ensembling automatically provides the stability. One of the steps in their procedure uses a 'longest lifetime' approach, parallel to Step 3 in EK, to obtain further stabilization; see **?**, Sec. 5.2, 5.3. In this context, a lifetime is the length of a branch of a dendrogram (as measured by the dissimilarity on the vertical axis) between two adjacent splits. Fourth, in our method, we also account for outliers (or other influential data points) that might affect the set of lifetimes.

Algorithm 2: Estimate of K_T (EK)

1 Use Steps 1-3 from Algorithm #1 to obtain D.

2 Form the dendrogram for the data under D using SL.

3 Use the **?** technique to find the two largest lifetimes.

4 For each of the two largest lifetimes, cut the dendrogram at that lifetime and examine the size of the clusters. Remove clusters that are both small (containing less that $\alpha 100\%$ of the data) and split off at or just below H_K. This gives two sub-dendrograms, one for each lifetime.

5 For each of the sub-dendrograms, cut at H_K. This gives two numbers of clusters. Take the mean of these two numbers of clusters as the estimate of the correct number of clusters.

Our methodology uses K-m and SL. However, we do not advocate these universally because any pair of clustering techniques that can yield and assemble basal clusters is an instantiation of our intuition. Our usage of K-m and SL rests on the established theory for those methods and the new theory we present here. For K-m, this amounts to consistency, so any other consistent methods, e.g., model based clustering, should perform well also in some settings. For SL, the reduction to the distances between the closest points of clusters means that our method should help us find non-convex clusters. As a pragmatic point, we tested variations on our methods in which complete linkage and average linkage were used in place of SL and found the differences small even though they favored SL.

3 Justification

In this section we provide motivation and some properties of our algorithms.

3.1 K-m with large K_ℓ

Let $X \sim P$ be a probability measure with density p and assume that p only takes values zero and a single, fixed constant. The places where p assumes a nonzero value are the clusters of P. Our first result shows that the support of p can be expressed as a disjoint union of small clusters in the limit of large n. Let $A \triangle B$ denote the symmetric difference between sets A and B. Now, given IID data $X^n = x^n = \{x_1, \ldots, x_n\}$ with $x_i \in \mathbb{R}^m$, write $\hat{\mathcal{C}}_K = (\hat{C}_{K1}, \ldots \hat{C}_{KK})$ to be a partition of $\mathbb{R}^m$ into K subsets based on a clustering of x^n. We have the following, based on the theorem in Sec. 7.

Corollary 1. *There exists a K_0 so that for $K \geq K_0$, $\exists m_1, \ldots, m_\ell \leq K$ for some ℓ with*

$$P\left(\mathsf{Supp}(P) \triangle \left(\cup_{j=1}^{l} \widehat{C}_{Km_j}\right)^c\right) < \epsilon. \tag{1}$$

That is, there are rates at which $n \to \infty$, $K \to \infty$ and $\epsilon \to 0^+$, so that in a limiting sense

$$\mathsf{Supp}(P) \approx \cup_{j=1}^{l} \widehat{C}_{Km_j}. \tag{2}$$

This corollary gives conditions under which the procedure of choosing K too large - in K-m, for instance - ensures that the union of the clusters for that K very closely approximates the support of P. The argument extends straightforwardly to P's that are continuous simply by representing those P's as limits of step functions, thereby including much more general clustering problems.

Since Assumptions 3), 4) and 5) in the theorem in Sec. 7 are straightforward to assess, we provide sufficient conditions for Assumptions 1) and 2) for the special case of K-m clustering. For any set A, let $\mathsf{diam}(A) = \sup_{x,y \in A} d(x, y)$ be the diameter of A. Assumption 1) is that $\forall K, m \; \exists \; C_{Km}$ so that as $n \to \infty$

$$P(\hat{C}_{Km} \triangle C_{Km}) \xrightarrow{P} 0$$

and Assumption 2) is that for any m, as $K \to \infty$,

$$\sup_{m=1,\ldots,K} \mathsf{diam}(C_{Km}) \to 0.$$

Starting with Assumption 1), recall that K-m uses the Euclidean distance to define the dissimilarity $d(x, x')$ for points x and x'. Formally, in the limit of large sample sizes, let μ_k, $k = 1, \ldots, K$, be the means of unknown classes C_{Km} under clustering $\mathcal{C}_K = \{C_{K1}, \ldots, C_{KK}\}$ and let C be the membership function that assigns data points x_i to clusters, i.e., $C(i) = m \Leftrightarrow x_i \in C_{Km}$ under the clustering $\mathcal{C}_K$. Then the K-m clustering is the $\mathcal{C}_K$ that achieves

$$\min_K \min_{\mu_1,\ldots,\mu_K} \sum_{k=1}^{K} \sum_{i:C(i)=k} \|x_i - \mu_k\|^2. \tag{3}$$

Under the K-m optimality criterion, given K there are $\mu_1, \ldots, \mu_K \in \mathsf{Supp}(P)$ such that the minimum in (3) can be written as

$$\sum_{k=1}^{K} \int_{C_{Km}} (x - \mu_{Km})^2 \; P(dx),$$

with the property that

$$x \in C_{Km} \Longleftrightarrow d(x, \mu_{Km}) \leq d(x, \mu_{K\ell}), \ell \neq m. \tag{4}$$

Defining the centroid of C_{Km} as

$$\mu_{Km} = E\left(x \mid x \in C_{Km}\right) = \int_{C_{Km}} x \; P(dx),$$

with corresponding estimate defined as

$$\widehat{\mu}_{Km} = E\left(x \mid x \in \widehat{C}_{Km}\right) = \int_{\widehat{C}_{Km}} x \; P(dx),$$

we can quote the following result.

Theorem 1. *(Pollard 1982) Suppose the random variables $X_1, \ldots, X_n$ are IID and have finite second moments. Let K be a positive integer and $\{\hat{\mu}_{K1}, \ldots, \hat{\mu}_{KK}\}$ be a set of values in $\mathcal{R}^m$ where the $\hat{\mu}_{Kj}$'s are distinct with probability one and satisfy (3) for all n. Then, there exists a μ_{Kj} so that for each j as $n \to \infty$, $\hat{\mu}_{Kj} \to \mu_{Kj}$ a.s. Therefore the K-m clustering $\widehat{\mathcal{C}}_K$ is consistent for $\mathcal{C}_K$ for any K, and in particular, for the true value of K.*

Proof. See **?**, Sec. 1. □

Since K-m is a centroid-based clustering, we have that

$$x \in \widehat{C}_{Km} \Longrightarrow \forall \ell \neq m \; d(x, \widehat{\mu}_{Km}) \leq d(x, \widehat{\mu}_{K\ell}),$$

so combining this with (4) we get that

$$\widehat{\mu}_{Km} \longrightarrow \mu_{Km} \Longleftrightarrow P(\widehat{\mathcal{C}}_K \triangle \mathcal{C}_K) \longrightarrow 0, \tag{5}$$

i.e., Assumption 1) is satisfied.

Turning to Assumption 2), consider the following example with K-m to understand the intuition behind it. Suppose a data set is generated as two clusters of the same number of outcomes, one with high variance and one with low variance; see the upper left panel in Fig. 1. Then, applying K-m with increasing K, e.g., $K = 2, 4, 10, 14, 18, 20$ as shown in Fig. 1 reveals that K-m partitions the two clusters more and more finely but continually assigns more clusters to the high variance data. In this context, Assumption 2) means that as n increases, the clusters will appear to 'fill in' yielding K regions with non-void interior for each $K \geq 2$ even if the n required for a given K increases with K.

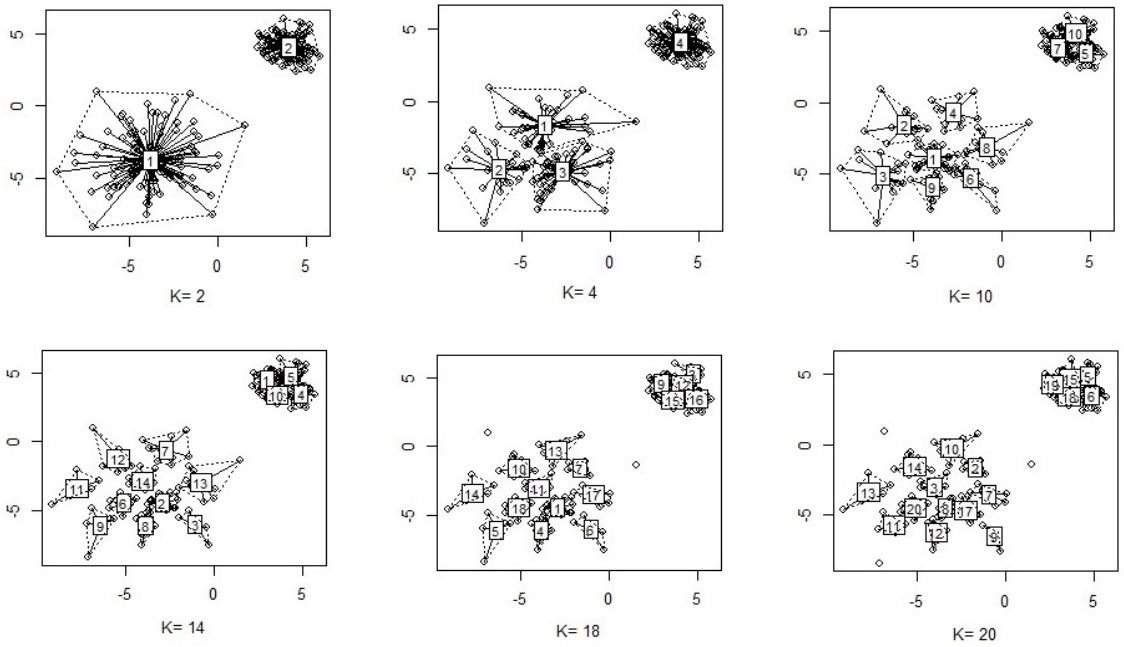

Figure 1: Plots of two clusters using different choices of K to see how K-m clustering divides the true clusters.

The example can be continued for higher K, higher K_T, and other distributions continuing to show that K-m tends to split the largest cluster until it is worthwhile to split the smaller cluster and then resumes splitting the larger cluster, and so on. A consequence of this is that $\hat{C}_{Km}$ tends to decrease in size as K increases and as assumed in Assumption 2) as $n \to \infty$. We state a version of this.

Proposition 1. *Suppose a clustering method continually splits the largest cluster on the population level as K increases. Then, given $\delta > 0$, there is a K_0 so that*

$$k > K_0 \Longrightarrow \mathsf{diam}(C_k) < \delta. \tag{6}$$

Proof. Let $\{\mu_{K1}, \ldots, \mu_{KK}\}$ be the centers of the optimal clusters and write

$$C_{Km} = \{x : |x - \mu_{Km}| < |x - \mu_{K\ell}|\}$$

for $\ell \neq m$. Let $\delta_0 = \max\{\mathsf{diam}(C_{Km}) : m = 1, \ldots, K\}$. Then, for K' large enough,

$$\max\{\mathsf{diam}(\widehat{C}_{K'm}) : m = 1, \ldots, K'\} \leq \frac{\delta_0}{2}.$$

Since this process can be repeated the proposition is established. □

Taken together, these results justify the K-m part of Step 1) of Algorithm #1.

3.2 Merging the 'basal' clusters

Next we turn to justifying the use of single linkage (SL) in the second part of Step 1 in Algorithm #1. Recall, SL means that we merge sets that are closest, i.e., given a distance d on, say, $C_{K1}, \ldots, C_{KK}$, SL clustering merges the two sets that achieve

$$d_{usual}(C_{Km}, C_{Km'}) = \min_{x \in C_{Km}, x' \in C_{Km'}} d(x, x'), \tag{7}$$

where d is a metric, e.g., Euclidean. The question that remains is how to choose d.

Being an order statistic, (7) can be affected by extreme values in the data set. So, we stabilize $d_{usual}(C_{Km}, C_{Km'})$ by replacing it with the 20th percentile of the distances between points in C_{Km} and $C_{Km'}$. That is, for $m \neq m'$, we find the distances

$$\{d(x, x') : x \in C_{Km}, x' \in C_{Km'}\},$$

take their order statistics, and find the approximately $\lfloor .2\#(C_{Km})\#(C_{Km'})\rfloor$ order statistic. We call the resulting dissimilarity d_{20}, i.e.,

$$d_{20}(C_{Km}, C_{Km'}) = 20^{th}\text{percentile of } \{d(x,x') : x \in C_{Km}, x' \in C_{Km'}\}. \tag{8}$$

Thus, with d_{20} we are using single linkage with respect to a dissimilarity that should be robust against extreme values. Other percentiles such as the fifth or tenth can also be used, but they gave values between d_{usual} and d_{20} in the examples we studied. It seemed from our work that d_{20} gave the best results when d_{usual} did not.

For the sake of completeness we next give conditions under which d_{usual} can be expected to perform well. We are unable to demonstrate this for d_{20} but suggest there is an analogous statement since using d_{20} gave performance that was *overall* comparable to d_{usual}. The idea is that if n is large enough, the components of the clustering will 'fill in' and therefore the distinct components will form clearly before any two of them are merged and hence there will be a level of the dendrogram where we can cut and get a perfect clustering.

Theorem 2. *Suppose* $\overline{\mathsf{Supp}}(P)$ *consists of* K_T *regions in* $\mathbb{R}^m$ *each a connected open set and that the open sets have disjoint closures. Let* δ *be the minimum distance between points in disjoint components, i.e.,*

$$\delta = \min_{m,m'=1,\ldots,K_T; m\neq m'} \min_{x\in C_m, x'\in C_{m'}} d(x,x').$$

For each n *let* $\hat{C}(K,n) = (C_{K,1}(n), \ldots, C_{K,K}(n)$ *be a clustering of size* K*. Then, for large enough* n *and* K*, the dendrogram of SL merging of the entries in* $\hat{C}(K,n)$ *under* d_{usual} *can be cut so the* K_T *components are perfectly separated.*

Proof. Suppose n and K are chosen so large that all the $\hat{C}_{K,j}$'s for $j = 1, \ldots, K$ have $\mathsf{diam}(\hat{C}_{K,j}) < \delta$, are nonvoid, and

$$P\left(\cup_{j=1}^{K_T} C_{K,j} \triangle \cup_{j=1}^{K} \hat{C}_{K,j}\right) \leq \epsilon \tag{9}$$

for some small pre-assigned $\epsilon > 0$. Now, holding the regions $\hat{C}_{K,j}$ fixed, let n increase so that the closest data point to any $x_i \in \hat{C}_{K,j}$ is an $x_{i'} \in \hat{C}_{K,j}$. This preserves (9).

Now, if we apply SL with d_{usual} to the entire data set we will always put points or subsets in the same component in the true clustering together before we merge points or

subsets from two distinct components in the true clustering. That is, the dendrogram of SL merging of the entries in $\hat{C}_{K,m}(n)$ under d_{usual} can be cut so the K_T components are perfectly separated, apart from regions of P-probability less than, say, 2ϵ.

This procedure can be done for a sequence of ϵ's tending to zero, requiring possibly larger n's and K' as ϵ decreases. Hence, it follows that SL using d_{usual} can perfectly separate the components of $\overline{\mathsf{Supp}}(P)$, and hence of P, in a limiting sense. □

The key hypothesis of this theorem is that the components of P are disjoint. In fact, this is often not the case – components may touch each other at individual points or may be linked by a very thin line. In these cases, the components of $\mathsf{Supp}(P)$ may not have disjoint closures or the closure of the components may not give $\overline{\mathsf{Supp}}(P)$, respectively. By adding an extra layer of limits – approximating the density of P by step functions – the result may be generalized to continuous P's. An intuitive discussion of when to use d_{usual} versus d_{20} as it relates to the possible non-disjointness of components is in the Appendix.

3.3 Using the overall membership matrix

In Steps 2 and 3 of Algorithm #1, a composite membership matrix $M(B)$ for B clusterings is defined. Then, single linkage clustering is applied to the rows of $M(B)$ in Step 4. Because D is based on $M(B)$ our results should be robust because by using several random starts for clustering and looking only at which cluster a data point is in, we are ensuring that the final clustering is a sort of 'consensus clustering'.

Our use of the matrix $M(B)$ means our method may be regarded as an ensemble approach. Each set of columns in $M(B)$ represents a clustering and pooling over clusterings in Step 4 effectively means that we are analyzing B clustering structures for the data. Ensembling the matrices is done by SL which groups similar clusters together. The result is that the final clustering is stabilized.

3.4 Estimating K_T by using lifetimes

Steps 1 and 2 in Algorithm 2 have been addressed in Subsecs. 3.1, 3.2, and 3.3. So, it remains to justify the use of lifetimes in Steps 3-5 for estimating K_T.

As can be seen in Fig. 3 of **?** where they give an example of lifetimes for a dendrogram, defining clusters by the use of a maximum lifetime has the tendency to amplify the separation between clusters so that points are usually only put in their final cluster near the leaves of the dendrogram. That is, there are often several long lifetimes that give reasonable places to cut the dendrogram so final clusters are well separated.

Our refinement of the technique in **?** is an effort to extend it to cases where the separation among clusters is not as clear. Indeed, removing subsets of data that are too small before applying **?** ensures that likely outliers or other aberrant points will not affect the collection of lifetimes.

3.5 Running time

Since the running times of SHC and EK are approximately the same, we only derive a running time for SHC. Recall SHC has three parts, namely, i) create the K_ℓ basal clusters by K-m; ii) merge them by SL; and iii) repeat these steps to stabilize the procedure. To deal with i) recall that **?** showed that K-m converges after $O(K_\ell n^2 \Delta^2)$ iterations uniformly over dimensions m where Δ is the spread of the point set; the key assumption was that only one point may change clusters per iteration. To deal with ii), observe that merging K_ℓ basal clusters requires the calculation of the distances between the points in the basal clusters. The running time for this is $O(K_\ell(K_\ell-1))$, since each basal cluster has approximately n/K_ℓ data points. The result is a running time for ii) of $O(K_\ell(K_\ell-1)n(n-1)/(K_\ell)^2) \approx O(n^2)$. Thus, the running time for i) and ii) is $O(K_\ell n^2\Delta^2 + n^2)$. For stability, i) and ii) must be run B times. These B iterations can be run independently and therefore can be done on a high throughput computing system. High throughput computing is free and widely available; see, for example, OSG[1]. So, it is reasonable to neglect B. Here $K_\ell = n/5$ and Δ is a constant depending on the data which we can take as bounded. So, the overall running time is $O(n^3)$ independently of m.

[1]http://www.opensciencegrid.org/, a project involving the computing facilities of several universities

4 Comparisons: Simulated data

Here we present two simulated examples to demonstrate how our proposed techniques compare to established methods. We do not argue our method is uniformly best over all clustering problems, only that it is often best and generally no worse than the best of the methods we have compared when variability is taken into account. This is partially because our method is only asymptotically optimal and partially because of the 'No Free Lunch' (NFL) theorems, see **?**.

Our two example data sets are named SCALES, FLAME and represent two paradigm problems – differing scales in different clusters and the effect of outliers. For the former, we used two sample sizes, $n = 700$ and 1400, simulating 200 data sets in each case. For SHCm and SHC20 we always chose $B = 200$ in Alg. #1.

Our tables show the MAI values rounded to two decimal places and SAI values that are rounded to three decimal places. Here, MAI is the mean accuracy index that we can calculate because we know the cluster for each data point and the SAI is the standard deviation associated with the MAI.

4.1 Scales

In this simulation the clusters are not convex, see Fig. **??**. The point is to see how the methods perform when the clusters have very different scales in one dimension (vertical) but similar scales on another (horizontal). The probability density is uniform over three regions C_1 given by $[0, 25]\times[0, 1]$, C_2 given by $[0, 25]\times 2, 23]$, and C_2 given by $[0, 25]\times[24, 25]$. We expect that the results in this example extend to higher dimensions.

Table 1 shows the average performance of the nine methods on the SCALES data. For $n = 700$, SHCm and SHC20 perform the best in MAI, although in fairness, they are not formally distinguishable from EAC or SPECC if SAI's are considered. The other methods fare notably worse which was a bit of a surprise since some of the K-m based methods such as TC do allow different clusters to have different scales. TC performs adequately but seems to be harmed by the fact that points not in a cluster may be closer to a particular cluster center than some points that really are in the cluster.

MAI performance improves with sample size although not quickly. For SHCm, SHC20,

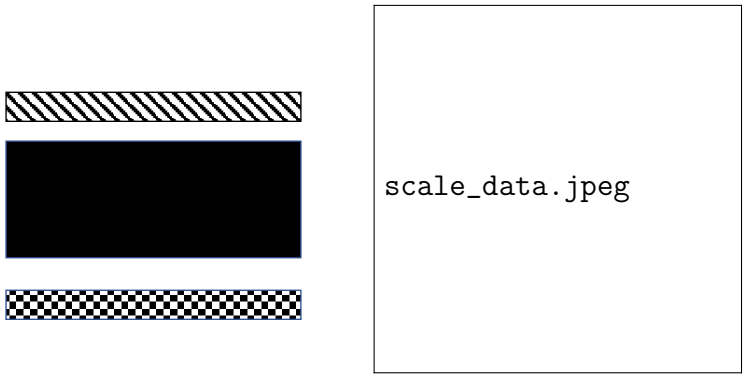

Figure 2: Left: Schematic showing the support of the distribution of the data. Right: A typical data set, $n = 700$.

and SPECC, this is a consequence of existing limiting results; for EAC there likely are parallel convergence results for the co-association matrix. A more substantial improvement is seen in the SAI's as n increases.

Table 1: Comparison of the nine methods on the SCALES data

	K-m	CHA	CT	SPECC	EAC	TC	FC	SHCm	SHC20
MAI ($n = 700$)	.44	.46	.50	.88	.91	.80	.42	.95	.95
SAI ($n = 700$)	.023	.025	.019	.131	.079	.162	.021	.116	.084
MAI ($n = 1400$)	.44	.46	.50	.90	.94	.81	.43	.97	.97
SAI ($n = 1400$)	.017	.019	.014	.147	.014	.150	.017	.021	.026

4.2 FLAME data

? developed a fuzzy clustering technique for DNA micro-array data which they considered on the test data given in Figure 2. The website **?** refers to this as the FLAME data set. On this data set, SHCm and SHC20 are seen to be the methods that best identify the two clusters. None of the other five methods do as well; CHA, EAC, and SPECC fail because they are distorted by the two outliers. K-m and CT do passably, but put too many points in the upper cluster. The overall performance is summarized in Table 2.

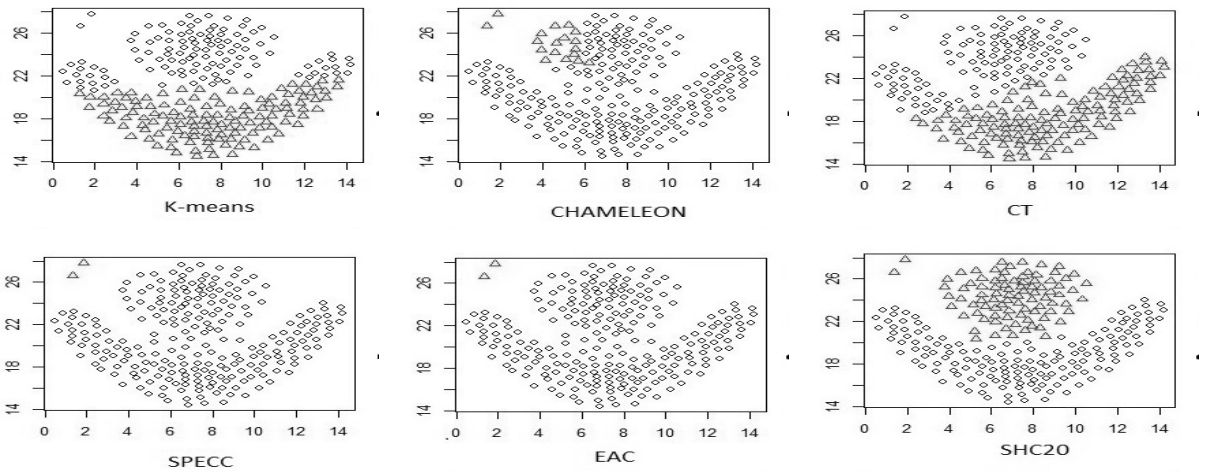

Figure 3: Clustering of FLAME data with six different methods as indicated on the panels.

Table 2: The comparison of the proposed methods on FLAME data.

	K-m	CT	CHA	SPECC	EAC	TC	FC	SHCm	SHC20
MAI	.84	.84	.71	0.7	.65	.75	0.85	.89	0.88
SAI	.031	0	0	.122	.000	.033	.000	.000	.000

5 Comparisons: Benchmark data

In this section, we compare the performance of the two proposed techniques with the existing techniques described in Sec. 1 using three benchmark data sets. They have 916, 37, and 14 dimensions, respectively. These are not 'high dimensionl' is the extreme sense of the term but they data types are commonly occuring and hard to visualize.

The first example uses all nine techniques on the GARBER data. It is a microarray data set found at **?**. It has 916 dimensions.

For these three data sets, we considered nine different clustering techniques: (K-m), EAC, CT, CHA, SPECC, TC, FC, and our two proposed techniques SHCm, and SHC20.

Because we can unambiguously assign a 'true' clustering to these data sets, we can also calculate an accuracy index (AI), i.e., the proportion of data points correctly assigned to their cluster. This was calculated using the software described in **?** and is different than the way we calculated MAI (and SAI) for the simulated data sets. Since there is randomness built into K-m, EAC, SPECC, and our methods, where necessary, we repeated

the techniques and report the mean AI (MAI) and its standard deviation (SAI). This was not necessary for CT since it does not vary over repetitions (hence SAI for CT is always zero).

The Garber data are from a classification problem so we have assumed that a unique true clustering exists as defined by the classes.

In all seven examples, we set $B = 200$ for EAC, SHCm, and SHC20 to ensure fairness. In six of the seven examples we set $K_{max} = 25$ in SHC; in the GARBER data set we chose $K_{max} = 11$ because $K_{max} < K_\ell - 2$ and $\lfloor n/5 \rfloor = \lfloor 74/5 \rfloor = 14$ so that it made sense to draw K_ℓ's from $DU(\lfloor n/6 \rfloor, \lfloor n/4 \rfloor) = DU(12, 18)$. The implication of our examples is that while other methods may equal or even perform slightly better than one or both of the SHC methods in some cases, no competitor beats them consistently by a substantial amount.

To conclude this section, in Subsec. 5.2 we compare six techniques for estimating K_T for all eight data sets. These are from Algorithm #2 (EKm and EK20), EAC, the Gap statistic, the silhouette distance, and the BIC. In this subjective setting EK20 performs best.

5.1 Higher dimensional data

Here we only present tables of MAI and SAI values. As a generality, CHA can only be expected to work well when the clusters are compact and can be separated; this is less likely as dimension increases.

5.1.1 GARBER data

To study the performance of the SHC's with high dimensional data, we used the microarray data from **?**. The data are the 916-dimensional gene expression profiles for lung tissue from $n = 72$ subjects. Of these, five subjects were normal and 67 had lung tumors. The classification of the tumors into 6 classes (plus normal) was done by a pathologist giving seven classes total. Accordingly, we expect seven clusters. The data set was constructed in **?** by filling in missing values estimated by the means within the same gene profiles. Table 3 presents the results. Clearly, the two versions of SHC work best and EAC is in second place although just barely. The other techniques peform discernably worse.

Table 3: Comparison of the methods on the GARBER data.

	K-m	CT	CHA	SPECC	EAC	TC	FC	SHCm	SHC20
MAI	.70	.63	.54	.71	.80	.70	.33	.82	.82
SAI	.109	0	0	.054	.000	.000	.000	.000	.000

5.1.2 Wheat metabolomics data

The other dataset that we want to consider for the evaluation of proposed methods, is about Wheat metabolomics data, **?**. Authors considered the t-Distributed Stochastic Neighbor Embedding (t-SNE) that is a technique for dimensionality reduction which was developed to for the visualization of high-dimensional datasets, see **?**. Dimension reduction has got a lot of attention in statistical inference. The dimension reduction classical techniques, such as principal component analysis (PCA) and its variants studied by different authors, for instance **?** compared the quality of clusters obtained from the original data to the quality of clusters obtained after projecting onto subsets of the principal component axes, they showed that the reduction dimension does not necessarily improve, and often degrade. Unlike PCA that tries to provide a combination of variables with the maximum amount of variation, t-SNE use the probability technique to measures pairwise similarities of the input objects to provide low-dimensional data. To evaluate the t-SNE, we run t-SNE on data, then apply the clustering technique on the reduced data and run the clustering, the results are The dataset has 313 observations with 37 quantitative variables, and we consider "Variety" (Antonius, CCP, Caphorn, DJ, MC2,Probus ,RdB ,R,Sandomir,Scaro,Titlis) as true label to evaluate the clustering techniques.

The results show an increase in accuracy for k-m, but not for SHC20 which developed to extract the non-convex pattern in data. We also run the SHC20 With tSNE on other dataset, the accuracies decrease significantly, for instance in the case of GARBER data, it dropped from 0.82(.000) to 0.567.

Table 4: Comparison of the methods on the what metabolics data.

	Without tSNE								
data	K-m	CT	CHA	SPECC	EAC	TC	FC	SHCm	SHC20
MAI	0.465	0.411	0.520	0.450	0.478	0.245	0.275	0.543	0.544
SAI	0.030	0.015	0.000	0.021	0.017	0.013	0.020	0.005	0.003
	With tSNE								
data	K-m	CT	CHA	SPECC	EAC	TC	FC	SHCm	SHC20
MAI	0.513	0.513	0.479	0.485	0.499	0.402	0.502	0.513	0.504
SAI	0.032	0.033	0.039	0.036	0.030	0.053	0.027	0.041	0.025

5.1.3 Flow cytometry data

5.2 Estimating clustering size

Estimating the number of clusters is challenging because the gial is to learn the structure of the underlying population. Nevertheless, there are several methods to estimate the number of clusters, $\widehat{K}_T$. For instance, **?** uses the gap statistic (gap) and it is implemented in the cluster package in R. Another popular method is the Silhouette distance (Sil), **?**, implemented in the fpc package in R. In addition, one can estimate K_T using the Bayesian information criterion (BIC), initializing by a hierarchical clustering as is implemented, for instance, in mclust in R; see **?**.

Table 5 shows the estimates of the number of clusters and the true number of clusters using six different methods for the data sets in this Section. The numbers in parentheses are the standard deviations (SD's) for the estimates. The bottom row is the absolute error (AE) formed by taking the sum of the absolute differences between the true and the estimated number of clusters. Even though the EK methods based on SHCm and SHC20 do not always identify the correct number of clusters, all other methods perform worse (for the data sets considered). Indeed, the gap statistic, Sil, and BIC do noticeably worse. Only EAC is comparable, and it is slightly worse than EKm which is slightly worse than EK20, again validating our recommendation of using the 20th percentile as a default. (The SD's do not seem to provide a helpful guide as to which methods are good; they are data set

dependent more than method dependent.)

Table 5: Estimated numbers of clusters and their SD's using six techniques. TAKE OUT MCDONAL'S DATA

data set	actual	EKm	EK20	EAC	Gap	Sil	BIC
FLAME	2	2.2(.3)	2.1(.2)	2.1(.3)	2.7(.8)	4(.0)	4(.0)
SCALE	3	4.84(1.32)	4.65(1.20)	3.30(2.31)	2(0)	3.85(0.45)	2.03(0.19)
GARBER	6	4.2(1.5)	4.6(2.2)	12(10.9)	5.7(2.5)	2(0)	5(0)
Cytometry							
Wheat	11	2(0)	2(0)	3.92(2.92)	2(0)	2(0)	7.5(3.2)
AE							

If the MCDONALD's data set is eliminated on the grounds that it is not a valid clustering problem, then the EK methods perform a little better in AE relative to the other methods. DO THIS.

6 Conclusion

Our ensemble clustering approach leads to two natural techniques that differ in the dissimilarity used in the single linkage step of our clustering approach – the usual Euclidean distance versus its 20-th percentile. We can establish formal results for the Euclidean distance and it has a natural geometric interpretation. However, the 20-th percentile dissimilarity gives performance that is no worse and sometimes meaningfully better than the Euclidean distance.

To evaluate the performance of the proposed methods, we tested them on a wide variety of qualitatively different clusterings. Our theory and examples suggest that our methods lead to accurate clusterings and that using $B \approx 200$ to form the membership matrices is usually enough to provide satisfactory stability, although cases requiring larger B can probably be constructed. Moreover, when the clusterings could be visualized in Sec. **??**, they look as if they should be stable in the sense of resampling indices such as Jaccard, adj-Rand, or Hubert-Arabie.

In our examples, our methods equal or outperform many standard or related methods such as spectral clustering, K-m, EAC (**?**), hybrid hierarchical clustering (**?**), CHAMELEON (**?**), spectral clustering, trimmed mean clustering and fuzzy clustering. Likewise, our methods tended to outperform other methods for estimating K_T. Moreover, we have theory to back up the asymptotic consistency of our methodology. Finally, in all the examples here, one of the two forms of our approach always performed better than the other techniques. Overall, we recommend SCH20 and EK20 as the better defaults.

7 Appendix

Here we give the statement and proof of the theorem from which the Corollary in Subsec. 3.1 is derived. Specifically, we have the following.

Theorem 3. *Suppose the following assumptions are satisfied:*

1. $\forall K, m \; \exists \; C_{Km}$ *so that as* $n \to \infty$

$$P(\hat{C}_{Km} \triangle C_{Km}) \xrightarrow{P} 0.$$

.

2. *For any* m, *as* $K \to \infty$,

$$\sup_{m=1,\ldots,K} \mathsf{diam}(C_{Km}) \to 0.$$

3. *For each* m *and* K, $P(C_{Km}) > 0$.

4. *The support of* p, $\mathsf{supp}(P)$, *consists of finitely many disjoint open sets with disjoint closures having smooth boundaries.*

5. *The random variable* X *generating* x^n *is bounded.*

Then for any fixed m, *along any sequence of sets* C_{Km} *with* $P(diam(C_{Km})) > 0$, *there is a* $z \in \overline{\mathsf{Supp}}(P)$, *the support of* P, *so that*

$$E(X|\widehat{C}_{Km}) \longrightarrow z, \tag{10}$$

as $n, K \to \infty$ *at appropriate rates,* n *faster than* K.

Proof. Consider a sequence $\langle \hat{C}_{Km} \rangle \mid_{K=1}^{\infty}$ for which $P(C_{Km}) > 0$; this is possible by Assumptions 1) and 3). By Assumption 2), $P(C_{Km}) \to 0^+$.

Step 1: For such a sequence,

$$E(X|\widehat{C}_{Km}) - E(X|C_{Km}) \xrightarrow{P} 0 :$$

Begin by writing

$$\begin{aligned} E(X|\widehat{C}_{Km}) - E(X|C_{Km}) &= \int_{\widehat{C}_{Km}} \frac{XdP}{P(\widehat{C}_{Km})} - \int_{C_{Km}} \frac{XdP}{P(C_{Km})} \\ &= \int_{\widehat{C}_{Km}} \frac{XdP}{P(\widehat{C}_{Km})} - \int_{\widehat{C}_{Km}} \frac{XdP}{P(C_{Km})} \qquad (11) \\ &\quad + \int_{\widehat{C}_{Km}} \frac{XdP}{P(C_{Km})} - \int_{C_{Km}} \frac{XdP}{P(C_{Km})}. \qquad (12) \end{aligned}$$

Since X is bounded, term (12) goes to zero as $n \to \infty$ by the Dominated Convergence Theorem since $I_{C_{Km}} - I_{\widehat{C}_{Km}} \to 0$ in P-probability under Assumption 1).

To deal with term (11), write it as

$$\int_{I_{\widehat{C}_{Km}}} \left(\frac{1}{P(\widehat{C}_{Km})} - \frac{1}{P(C_{Km})} \right) XdP. \qquad (13)$$

Since X is bounded by M, say, the absolute value of term (13) is bounded by

$$MP(\widehat{C}_{Km}) \left| \frac{1}{P(\widehat{C}_{Km})} - \frac{1}{P(C_{Km})} \right| = M \left| 1 - \frac{P(\widehat{C}_{Km})}{P(C_{Km})} \right|. \qquad (14)$$

Now, by Assumption 1, with probability at least $1 - \eta$, for any $\eta > 0$, we have

$$\frac{P(\widehat{C}_{Km})}{P(C_{Km})} \longrightarrow 1,$$

as $n \to \infty$. So, the factor in absolute value bars in (14) can be made less than any pre-assigned positive number, for instance, η/M, giving that (13) can be made arbitrarily small as $n \to \infty$. Consequently,

$$E(X|\widehat{C}_{Km}) = E(X|\widehat{C}_{Km}) \pm E(X|C_{Km}) = o_P(1) + E(X|C_{Km})$$

and Step 1 is complete.

Step 2: By Assumption 2), $\exists z$ such that $C_{Km} \to \{z\}$. So, by Step 1, $E(X \mid \hat{C}_{Km}) \xrightarrow{P} z$ as $n \to \infty$. Now, to prove the theorem, it remains to show $z \in \overline{\mathsf{supp}(P)}$.

By way of contradiction, suppose $z \notin \overline{\mathsf{Supp}(P)}$. Then, since $\overline{\mathsf{Supp}(P)}$ is a closed set by Assumption 4), its complement is open and hence $\exists \epsilon > 0$ so that $B(z, \epsilon) \subset \overline{\mathsf{Supp}(P)}^c$, where $B(z, \epsilon)$ indicates a ball centered at z of radius ϵ. However, consider a sequence of sets C_{Km} for some fixed m with $\forall K : P(C_{Km}) > 0$. Such a sequence must exist for some m by Assumption 3). By Assumption 2), we have that $\mathsf{diam}(C_{Km}) \to 0$ as $K \to \infty$. So, $\exists K_0$ such that $\forall K \geq K_0$, $C_{Km} \subset B(z, \epsilon)$, and therefore $P(C_{Km}) = 0$ by letting n and K increase at appropriate rates, a contradiction. Hence, $z \in \overline{\mathsf{Supp}(P)}$. □

In Subsec. 3.2, the choice of d_{usual} versus d_{20} was raised. First, when Assumption 4 of Theorem 1 is satisfied, we have found that d_{usual} works well: In a limiting sense, two basal sets from the same component will always be joined before either is joined to another component. However, when Assumption 4 of Theorem 1 is not satisfied, d_{usual} does not have this property. In these cases, we have found d_{20} to work better; see Subsec. **??**. The examples in Subsecs. **??** and 4.2 also do not seem to satisfy Assumption 4 but for these cases d_{usual} and d_{20} give comparable results. This may only mean that Assumption 4 is necessary but not sufficient.

Why does the 20-th percentile work well in cases where Assumption 4 is not satisfied? There are two inuitive answers. If Assumption 4 is not satisfied and the clusters are highly non-convex d_{usual} will be much more sensitive to the boundary values of the clusters than d_{20}. Consequently, there may be overly influential data points that will affect the sequence of merges of the basal clusters in ways that are not representative of the support of P. Using d_{20} in place of d_{usual} reduces the influence of these data points, i.e., the presence of extreme points suggests that d_{20} should be preferred. Indeed, the examples in Subsecs. **??** and 4.2, where d_{usual} and d_{20} give equivalent results, do not have clusters with extreme points.

Second, pre-asymptotitcally, on regions where the density p of P is very small, d_{20} will suffer more than d_{usual} because of the sparsity of data. That is, d_{usual} will be better at joining nearby K-means obtained regions because it is looking at a minimum distance. Asymptotically in sample size, this difference should decrease as the richness of the data to represent P increases.

Acknowledgements: The first three authors gratefully acknowledge support from NSF-

DTRA grant DMS-1120404.

8 Supplementary Materials

Title: Github site with code and example data.

R-code for proposed methods: R-code to perform the methods described in this article, with example data and results, are available from Github at `https://github.com/saeidamiri1/GHC/wiki`

References