A General Hybrid Clustering Technique

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

July 31, 2018

Abstract

Here, we propose a clustering technique for general clustering problems including those 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 step in this stage is 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 arguments to justify the steps in the stages of our methods and we provide examples involving simulated and published data to compare our technique with other techniques.

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

[*]The authors gratefully acknowledge the University of Nebraska Holland Computing Center for invaluable computational support.

1 Introduction

Clustering is an unsupervised technique used to find underlying structure in a data set 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 Kaufman and Rousseeuw (2009), Duda et al. (2012), and Aggarwal and Reddy (2013).

There are numerous clustering procedures 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. Our method is not limited to convex clusters and is not as dependent as other methods on subjective choices of quantities such as dissimilarities. That is, we combine several clustering methods and principles in sequence so that one part of the cumulative method may correct for weaknesses in other parts hence 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 will be asymptotically correct for simple cases and hence approximately correct more generally. 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. Denote a clustering of size K by $\mathcal{C}(K) = (C_{K1}, \ldots, C_{KK})$ and assume that i) there is only one clustering for each K and ii) 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 if needed 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 we recommend it.

We compare our two clustering methods to seven other clustering methods. K-m, as implemented by the R package stats, is the most familiar method so we include it even though it does not to perform well for nonconvex clusters unless they are well separated.

There are many precedents for the hybrid clustering described in stages 1) and 2). Perhaps the closest is Fred and Jain (2005). They create many clusterings of different sizes (by K-m) that can be pooled via a 'co-association matrix' that counts the number of times a given pair of data points is put in the same cluster, over a collection of clusterings, whereas we use a membership matrix. The co-association matrix can be modified or, more precisely, updated to give a dissimilarity so that SL clustering can be used to give a final clustering. Fred and Jain (2005) refer to this as evidence accumulation clustering (EAC). EAC differs meaningfully from our technique in four ways. First, we ensemble (and hence stabilize) directly by membership in terms of Hamming distance whereas EAC ensembles by updating a co-association matrix. Second, our procedure uses an extra step of growing and pruning (see Step 5 in Algorithm #1) to look forward in a dendrogram to search for better clusterings. Third, our method treats outliers explicitly. Fourth, by taking advantage of the chaining property of SL, our method handles non-convex clusters better than the Fred and Jain (2005) 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 the details of the way these techniques are combined. Overall, our 'fine-tuning' of their technique simplifies it and seems to give better results. The code we wrote to implement EAC can be found at `https://github.com/saeidamiri1/GHC/wiki`.

Third, a technique that is similar in spirit to ours is due to Chipman and Tibshirani (2006), 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'. Also, our method has a stabilization stage, uses a 'grow and prune' strategy, and deals with outliers explicitly, unlike CT. So, it is unclear how well CT performs when the true clusters are non-convex. Code implementing CT is in the R package hybridHclust.

Fourth, another method that is similar in spirit to ours is due to Karypis et al. (1999). This method, 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. This method is coded in the software package CLUTO. Our method differs from Karypis et al. (1999) in five key ways. First, we use K-m instead of graph partitioning. Second, we use SL whereas Karypis et al. (1999) 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 uses a 'grow and prune' strategy, and fifth, handles outliers explicitly. Because of its elaborate optimization, CHA can also 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 von Luxburg et al. (2008). This is not the same as strong as consistency of the clustering itself, but it is suggestive. There are many forms of spectral clustering and we simply use the default version in the R package kernlab in R.

Sixth, we also consider a robustified form of clustering called trimmed clustering, TC, and implemented by the R-package tclust, see García-Escudero et al. (2008). 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 García-Escudero et al. (2008) 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 TC 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. TC 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 in Subsec. 4) where TC is unable to accommodate the difference in scales along different directions. Overall, when the specific form of sensitivity to the spread of clusters used in TC 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 D'Urso (2015); see also the references therein. Also, see Ferraro and Giordani (2015) for details of implementation. The version we use here is relatively straightforward, e.g., it is based on K-m (see Bezdek (1981)) 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 D'Urso (2015) for a concise summary. The R-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 suggestion for hybrid methods is in Zhong and Ghosh (2003) who conjectured that using K-m with K too large and SL might enable a technique to find nonconvex clusters.

As clustering is often employed in high dimensional contexts, we consider pre-processing the data by using dimension reduction. Specifically, we test whether using t-distributed stochastic neighbor embedding (t-SNE, R package tsne) to reduce the dimension $m > 2$ to two before applying a clustering method gives better results; see van der Maaten and Hinton (2008). In our examples, t-SNE worsens the accuracy of the clustering methods and this is in keeping with the fact that t-SNE preserves neither distances nor point density and only preserves nearest neighbors to a limited extent. Thus, t-SNE is good for visualization but not in general for clustering, see Wattenberg et al. (2016) for more details. Indeed, t-SNE can have worse or better clustering properties than t-SNE; see Wattenberg et al. (2016) and Kessler et al. (2015), respectively, for examples of these possibilities. Other dimension reduction strategies such as principal components often only seek linear dimension reduction; when there are linear substructures, these method can perform well even as clustering techniques. However, our focus is on nonconvex clusters where linearity cannot be presumed.

In addition to proposing two forms of a new hybrid clustering technique (see Algorithm #1 and Subsec 3.2) we present two ways to estimate the correct value K_T of K via Algorithm #2. We call these EK and EK20; the latter is based on the 20-th percentile of the distances used in SL. Essentially, we combine the first three steps of Algorithm #1 with a modification of Fred and Jain (2005). It is important to note that EAC, like our method, can also be used to estimate the number of clusters in a clustering. Code for EAC, EK, and EK20 can be found at `https://github.com/saeidamiri1/GHC/wiki`. The Gap statistic, Silhouette distance, and Bayes information criterion can also be used to estimate K_T using R packages cluster and mclust. Thus, we compare the accuracy of six methods for estimating K_T for several data sets. Even though the data sets cannot be regarded as outcomes of a single stochastic process, averaging the performance of the six methods gives an 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 sets. 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 published data sets. Again, we see that at least one of our methods is the best. 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 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 that may represent outliers. 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 attempts to include points that are more representative of another cluster. That is, the definition of K^* is a formalization of

```
Algorithm 1: Stabilized Hybrid Clustering (SHC)
1 Given $K$, start by drawing a value of $K_\ell$ and then drawing a value of
  $K_b \sim DUnif(2, K_{max})$ where $K_{max} < K_\ell$, for $b = 1, \dots, B$. For each $K_b$, use
  randomly generated initial conditions to obtain $\mathcal{C}(K_b) = \{C_{b1}, \dots, C_{bK_b}\}$: Use K-m
  clustering to generate a clustering of size $K_\ell$ 'basal' clusters and then use SL
  clustering to form $\mathcal{C}_{K_b}$ by merging the $K_\ell$ basal clusters.
2 For $\mathcal{C}(K_1)$, let $M_1 = (\chi(s,t))_{s=1,\dots,n;t=1,\dots,K_1}$ 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
  $M_1, \dots, M_B$ for clusterings $\mathcal{C}(K_2), \dots, \mathcal{C}(K_B)$, respectively. Concatenating $M_b$'s
  gives the overall membership matrix $M(B) = [M_1, \dots, M_B]$.
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 $x_i = (x_{i,1}, \dots, x_{i,S})$ and $x_j = (x_{j,1}, \dots, x_{j,S})$ and so give
  dissimilarities $d_{ij} = \sum_{m=1}^{S} \mathsf{I}(x_{im}, x_{jm})$ 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
  $H_K$. Let the lengths of the $K$ lines from $H_K$ to the next split in the dendrogram be
  denoted $h_1, \dots, h_K$ with mean $\bar{h} \geq 0$. 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 $K \geq K*$.
5 Now, $\exists v \geq 0$, an integer, so that $K^* = K + v$. 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 $\alpha n$ do not exist, adjust $\alpha$ 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.
```

the concept of the optimal depth down a dendrogram that it is worth going under SL by taking advantage of the chaining property of SL.

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 Fred and Jain (2005). Recall that Fred and Jain (2005) considered a set of 'lifetimes': 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. Fred and Jain (2005) 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 Fred and Jain (2005) Secs. 4 and 5 because they simply generate a co-association matrix from randomly chosen K's (and K-m clustering) and take the estimate of K_T to be the size of the clustering resulting from EAC. Accordingly, EK differs from the way EAC estimates K_T in the same ways that EK differs from EAC.

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 Fred and Jain (2005) 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 than $\alpha 100\%$ 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 the Appendix, 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 $K \to \infty$,

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

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 $\mu_k, \ k = 1,\dots,K$, be the means of unknown classes C_{Km} under clustering $\mathcal{C}_K = \{C_{K1},\dots,C_{KK}\}$ 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

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

Under the K-m optimality criterion, given K there are $\mu_1,\dots,\mu_K \in \mathsf{Supp}(P)$ 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

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

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 Pollard (1982), 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 normal clusters with the same number of outcomes, one with high variance and one with low variance; see Fig. 1. Then, applying K-m with increasing K gives clusterings that partition the two actual clusters ever more finely, but continually assigning 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.

The example can be continued for higher K, higher K_T, and other distributions to argue 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*

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

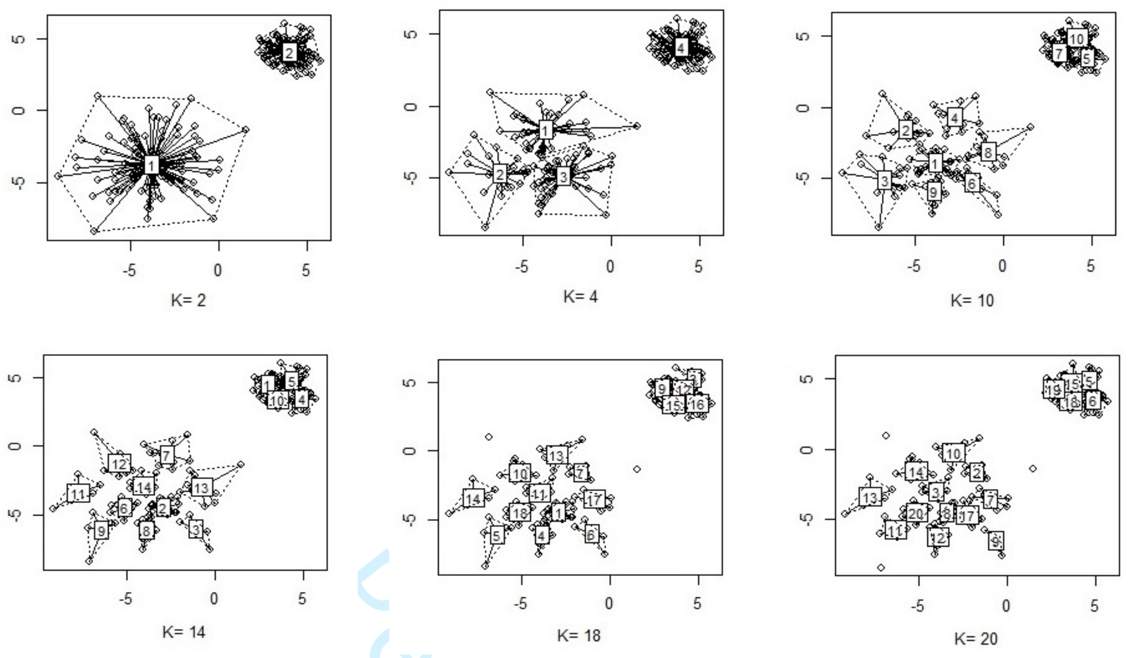

Figure 1: K-m clusterings of two normal clusters, one with a small variance and one with a large variance. The values $K = 2, 4, 10, 14, 18, 20$ were used.

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,

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

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 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 SL 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. We denote the clustering method using d_{20} in place of d_{usual} by SHC20.

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

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

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's 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, may be linked by a very thin line, or may have a common boundary. In these cases, the components of $\mathsf{Supp}(P)$ may not have disjoint closures or the closure of the components may not give $\overline{\mathsf{Supp}}(P)$. By adding an extra layer of limits – approximating the density of P by step functions – the result may be generalized to continuous P's. A discussion of when to use d_{usual} versus d_{20} 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. The reason is that using several random starts for clustering and looking only at which cluster a data point is in ensures that the final clustering is a sort of 'consensus clustering'.

Our use of $M(B)$ means our method is 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. Thus 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 Fred and Jain (2005) 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 Fred and Jain (2005) 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 Fred and Jain (2005) 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, SHC20, 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 Har-Peled and Sadri (2005) 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.

4 Comparisons: Simulated data

We present two examples with simulated data 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 not much worse than the best of the methods against which 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 Wolpert and Macready (1997).

Our two example data sets are named SCALES and FLAME; they 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 the latter $n = 240$. For SHC and SHC20 we always chose $B = 200$ in Algorithm #1.

Our tables show the mean accuracy index (MAI) values rounded to two decimal places and SAI values that are rounded to three decimal places. The MAI can be calculated because, since we know the true cluster for each data point, we can find the proportion of the n data points that are in the correct cluster. The SAI is the standard deviation associated with the MAI.

4.1 Scales

Even though in this simulation the clusters are convex, see Fig. 2, 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

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

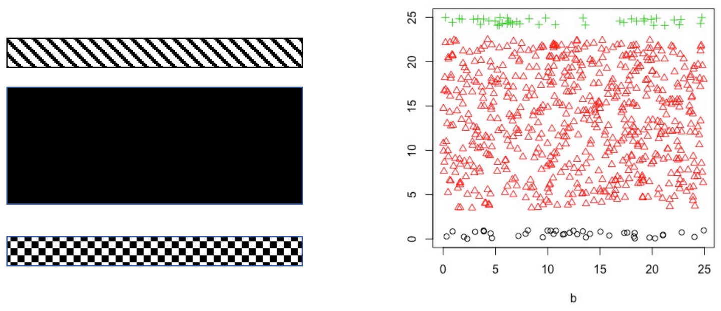

Figure 2: Left: Schematic showing the support of the distribution of the data. Right: A typical data set, $n = 700$, assuming the true density is uniform on the shaded regions.

$n = 700$, SHC 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 also 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 SHC, SHC20, and SPECC, this is a consequence of existing limiting results; for EAC there likely are parallel convergence results. Unsurprisingly, 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	SHC	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

Fu and Medico (2007) developed a fuzzy clustering technique for DNA microarray data which they considered on the test data FLAME given in Figure 3; see Fränti and Sieranoja

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

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

(2018). On this data set, SHC 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.

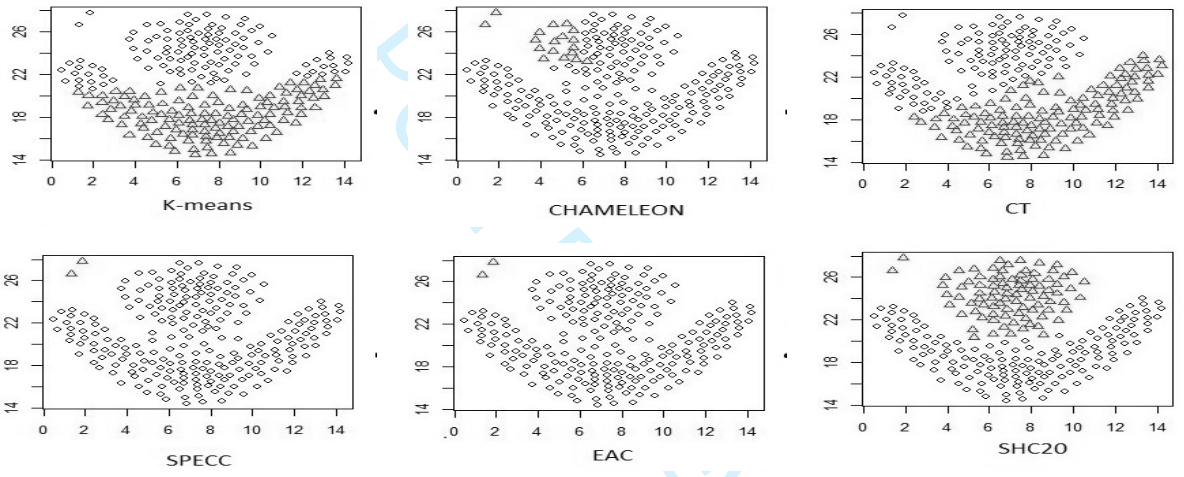

Figure 3: Clusterings of the FLAME data set from Fränti and Sieranoja (2018) using the six methods as indicated on the panels.

5 Comparisons: Published data

In this section, we compare the performance of the two proposed techniques with the existing techniques described in Sec. 1 using two published data sets. They have 916 and 36 dimensions so they are not ‘high dimensional’ in the extreme sense but the data types are commonly occurring and hard to visualize.

The first example uses all nine techniques on the GARBER data set. It is a 916 dimensional microarray data set found in Garber et al. (2001). GARBER is from a classification problem so we have assumed that a unique true clustering exists as defined by the classes.

We also examined a data set generated from wheat metabolomics; see Kessler et al. (2015). It has 36 dimensions and some extra complexities discussed in Subsec. 5.2.

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

Because we can unambiguously assign a 'true' clustering in these data sets, we can 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 Fraley et al. (2012) and is slightly different from the way we calculated AI's and hence the MAI (and SAI) for the simulated data sets. Since there is randomness built into K-m, EAC, SPECC, as well as in 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 is a one pass method making its SAI always zero.

In both examples, we set $B = 200$ for EAC, SHC, and SHC20 to ensure fairness. In GARBER 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 $DU(\lfloor n/6 \rfloor, \lfloor n/4 \rfloor) = DU(12, 18)$. For the wheat metabolomics data set we used $K_{max} = 25$ in SHC. 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. In addition, we used t-SNE as a pre-processing step on the data before clustering so we could compare the combined effect of t-SNE and a clustering method with pure clustering methods. Unsurprisingly, as t-SNE is primarily a visualization technique and hence not very flexible, it never improved performance.

To conclude this section, in Subsec. 5.3 we compare six techniques for estimating K_T for all four data sets. These are from Algorithm #2 (EK and EK20), EAC, the Gap statistic, the silhouette distance, and the BIC. The EK20 method seems to perform best.

5.1 GARBER data

The Garber data are the 916-dimensional microarray gene expression profiles for lung tissue from $n = 72$ subjects reported in Garber et al. (2001). 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. Table 3 presents the results. The top half of the table shows that the two versions of SHC work best and EAC is in second place although only by a small amount. The other techniques perform discernibly worse.

The bottom half of the table shows the corresponding results when t-SNE is used to pre-process the data by reducing its dimension to two. As a visualization technique, t-SNE often performs well. However, when used with a clustering method, it tends to reduce performance (as measured by MAI), at least when the clustering is relatively successful. In Table 3 it is seen that even though EAC is the best clustering strategy to use with t-SNE (even if the improvement over SHC is slight) none of the MAI's are satisfactory.

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

	Without tSNE								
	K-m	CT	CHA	SPECC	EAC	TC	FC	SHC	SHC20
MAI	.70	.63	.54	.71	.80	.70	.33	.82	.82
SAI	.109	0	0	.054	.000	.000	.000	.000	.000
	With tSNE								
data	K-m	CT	CHA	SPECC	EAC	TC	FC	SHC	SHC20
MAI	0.338	0.257	0.263	0.343	0.588	0.550	0.257	0.570	0.558
SAI	0.040	0.027	0.027	0.031	0.025	0.045	0.027	0.019	0.024

5.2 Wheat metabolomics data

The second data set that we use to test our methods is the wheat metabolomics data presented in Kessler et al. (2015). This data set, WHEAT, has 297 observations with 36 continuous variables and two categorical variables, cultivar and year (in which the data were gathered). In addition, the data are from two types of farming, organic and conventional. Here we ignore the cultivar and farming style thereby regarding the data as having two clusters corresponding to two years of data collection. (In fact there were three years of data, but we omitted the middle year of data because only one cultivar was used.) We did this because Kessler et al. (2015) argued that year was the single most important variable and that without correcting for year, little difference could be seen among the cultivars. They similarly argued that without correcting for both year and cultivar, little difference could be seen between the two methods of farming.

Results analogous to Table 3 are given in Table 4. The top half of the table shows the MAI's and SAI's for the nine clustering methods directly applied to the data. It is seen that SHC20 is the best in terms of MAI while other methods such as K-m and CT also perform well. The discrepancy between SHC and SHC20 is quite large and suggests that the clusters may be quite dispersed because taking the 20-th quantile of the distances captures the clusters so much better than using a minimum distance.

The bottom half of Table 4 shows the results when t-SNE is used to pre-process the data by reducing it to two dimensions. As with the GARBER data, no clustering method is improved by the use of t-SNE; all of them give much worse performance. In Fig. 5 of Kessler et al. (2015) the authors also used t-SNE on their data for visualization purposes. They found that when the classes of year and cultivar were colored, the result was useful as a way to visualize the proximity of the data points by way of cultivar and year but that the points did not exhibit strong separation by class. In their Fig. 6, they found a similar result when the points were colored by farming system and year. Thus, our findings on the effect of t-SNE are consistent with their findings.

Table 4: Comparison of the methods on the WHEAT metabolomics data.

	Without tSNE								
data	K-m	CT	CHA	SPECC	EAC	TC	FC	SHC	SHC20
MAI	0.912	0.912	.920	0.585	0.699	0.883	0.912	0.559	0.962
SAI	0	0	0	0	0.187	0.048	0	0.091	0
	With tSNE								
data	K-m	CT	CHA	SPECC	EAC	TC	FC	SHC	SHC20
MAI	0.679	0.680	0.661	0.543	0.582	0.570	0.683	0.546	0.546
SAI	0.071	0.075	0.070	0.010	0.045	0.019	0.067	0.020	0.021

5.3 Estimating clustering size

Estimating the number of clusters is challenging because the goal of learning the structure of the underlying population is often difficult. Nevertheless, there are several methods to produce an estimate $\widehat{K}_T$ the true number of clusters, K_T. Our Algorithm #2 provides one method, EK. Second, we can use the dissimilarity generated by d_{20} to form D leading to a similar method we denote EK20. Third, as mentioned earlier, EAC can be converted into a method for estimating K_T. Fourth and fifth, there are established methods such as the gap statistic (Gap), Hastie et al. (2001) and the silhouette distance, Kaufman and Rousseeuw (2009), respectively. Sixth, we can also estimate K_T using the Bayesian information criterion (BIC). (This requires initializing by a hierarchical clustering as in mclust; see Fraley and Raftery (2007).)

Table 5 shows the estimated number of clusters and the true number of clusters for the four data sets we have examined in Secs. 4 and 5 using the six different methods for estimating K_T. 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 for each method Even though EK and EK20 do not always identify the correct number of clusters, all other methods perform worse (for the data sets considered). Indeed, the errors associated with the gap statistic, silhouette distance, and EAC are twice the errors of the other three methods. Of the other three methods, EK20 is the best although EK is only a little worse and BIC is only slightly worse that EK. This validates our recommendation to use the 20th percentile as a default. (The SD's do not seem to provide a helpful guide as to which methods are good; other computations not presented here suggest they are more data set dependent than method dependent.) The weakness in this argument is that the four data sets we used cannot realistically be taken as representative of a large class of data sets. However, the results are suggestive pending a more systematic comparison.

Table 5: Estimated numbers of clusters and their SD's using six techniques.

data set	actual	EK	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)
Wheat	2	2.03(.11)	2.15(.23)	3.7(2.42)	7.04(3.10)	2(0)	2(0)
AE		3.87	3.30	8.10	7.04	6.85	3.97

6 Conclusions

The overall recommendation from our results here is a three stage process i) Use EK20 to estimate the number of clusters a clustering should have (Algorithm #2), ii) apply SHC20 to cluster the data (Algorithm #2), and, iii) given the clustering, examine it by secondary means as a sort of 'sanity check'. This third stage, which we do not discuss here, may include assessing the clustering (or clusterings of nearby sizes) by stability measures, e.g., the Jaccard or adjusted Rand index, and applying visualization techniques such as t-SNE, principal components, etc. If n is large enough, then disjoint sets of the data can be used for each stage.

We have tested two variants on our basic method of using SL to merge clusters from K-m clustering with K too large. One is the 'pure form' of SL, the other uses the 20-th percentile of distances used to form the SL. We found in our examples that the latter generally worked better than the former even though our theory is for the former. We think this occurs because the kind of robustness built into using quantiles is helpful for many complex data sets (high dimensional or otherwise difficult). Likewise, we have found that stabilizing the corresponding technique for estimating K_T by using the 20-th percentile outperforms the other methods we considered. Essentially, we are building stability into our clustering methodology (mainly via the dissimilarity matrix) albeit differently from consensus clustering techniques or post-clustering evaluations of stability. Note that in the third stage above, it may be reasonable to check whether EK or SHC gives better results.

Note that our methodological recommendation does not start with visualization even though it is not unreasonable to advocate this. The reason is that visualization typically involves dimension reduction and often this does not help; see Kessler et al. (2015). Even when visualization is useful, e.g., by t-SNE, the hypotheses under which it is reliable can be onerous effectively amounting to having a good clustering in hand; see the results in Arora et al. (2018). Classical visualization techniques such as principal component analysis (PCA) and its variants have been studied by different authors. For instance Yeung and Ruzzo (2001) compared the quality of clusters obtained from original data to the quality of clusters obtained after projecting the data onto pairs of principal component axes. They argued that the even when PCA based dimension reduction gives a good clustering it is not in general better than the result of good clustering techniques, often underperforming relative to them – as seen here for t-SNE. Otherwise put, visualization and clustering, although closely related, are distinct goals requiring distinct methods.

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 $\langle \hat{C}_{Km} \rangle |_{K=1}^{\infty}$ 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

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

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

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

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, $z \in \overline{\mathsf{Supp}(P)}$. $\square$

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 such cases, we have often found d_{20} to work better. The examples in Subsec. 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 example in Subsec. 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

Aggarwal, C. C. and C. K. Reddy (2013). *Data clustering: algorithms and applications.* Chapman and Hall/CRC.

Arora, S., W. Hu, and P. Kothari (2018). An analysis of the t-SNE algorithm for data visualization. arXiv:1803.01768v1.

Bezdek, J. (1981). *Pattern Recognition with Fuzzy Objective Function Algorithms.* Norwell, MA: Kluwer Academic Publishers.

Chipman, H. and R. Tibshirani (2006). Hybrid hierarchical clustering with applications to microarray data. *Biostatistics 7*(2), 286–301.

Duda, R., P. Hart, and D. Stork (2012). *Pattern classification.* John Wiley & Sons.

D'Urso, P. (2015). Fuzzy clustering. In C. Hennig, M. Meila, F. Murtagh, and R. Rocci (Eds.), *Handbook of Cluster Analysis*, pp. 545–574. Boca Raton, FL: Chapman and Hall/CRC.

Ferraro, M. B. and P. Giordani (2015). A toolbox for fuzzy clustering using the R programming language. *Fuzzy Sets and Systems 279*, 1–16.

Fraley, C. and A. Raftery (2007). Model-based methods of classification: using the mclust software in chemometrics. *Journal of Statistical Software 18*(6), 1–13.

Fraley, C., A. Raftery, T. Murphy, and L. Scrucca (2012). mclust version 4 for R: Normal mixture modeling for model-based clustering, classification, and density estimation. Technical report, University of Washington, Seattle.

Fränti, P. and S. Sieranoja (2018). K-means properties on six clustering benchmark datasets. `http://cs.uef.fi/sipu/datasets/` Accessed: 2018-07-20.

Fred, A. and A. Jain (2005). Combining multiple clusterings using evidence accumulation. *IEEE transactions on pattern analysis and machine intelligence 27*(6), 835–850.

Fu, L. and E. Medico (2007). FLAME, a novel fuzzy clustering method for the analysis of dna microarray data. *BMC Bioinformatics 8*, 3.

Garber, M. E., O. G. Troyanskaya, K. Schluens, S. Petersen, Z. Thaesler, M. Pacyna-Gengelbach, M. van de Rijn, G. D. Rosen, C. M. Perou, R. I. Whyte, R. B. Altman, P. O. Brown, D. Botstein, and I. Petersen (2001). Diversity of gene expression in adenocarcinoma of the lung. *Proceedings of the National Academy of Sciences 98*(24), 13784–13789. `http://genome-www.stanford.edu/lung_cancer/adeno/` Accessed: 2018-07-20.

García-Escudero, L. A., A. Gordaliza, C. Matrán, and A. Mayo-Iscar (2008). A general trimming approach to robust cluster analysis. *The Annals of Statistics 36*, 1324–1345.

Har-Peled, S. and B. Sadri (2005). How fast is the k-means method? *Algorithmica 41*, 67–82.

Hastie, T., R. Tibshirani, and G. Walther (2001). Estimating the number of data clusters via the gap statistic. *J Roy Stat Soc B 63*, 411–423.

Karypis, G., E.-H. Han, and V. Kumar (1999). Chameleon: Hierarchical clustering using dynamic modeling. *Computer 32*(8), 68–75. `http://glaros.dtc.umn.edu/gkhome/cluto/cluto/overview` Accessed: 2018-07-20.

Kaufman, L. and P. Rousseeuw (2009). *Finding groups in data: an introduction to cluster analysis*, Volume 344. John Wiley & Sons.

Kessler, N., A. Bonte, S. Albaum, P. Mader, M. Messmer, A. Goesman, K. Niehaus, G. Langenkamper, and T. Nattkemper (2015). Learning to classify organic and conventional wheat – a machine learning driven approach using the meltdb 2.0 metabolomics analysis platform. *Front. Bioeng. and Biotech. 3*, 35.

Pollard, D. (1982). Quantization and the method of k-means. *IEEE Transactions on Information theory 28*(2), 199–205.

van der Maaten, L. and G. Hinton (2008). Visualizing data using t-SNE. *J. Mach. Learn. Res. 9*, 2579–2605.

von Luxburg, U., M. Belkin, and O. Bousquet (2008). Consistency of spectral clustering. *Ann. Stat. 36*, 555–586.

Wattenberg, M., F. Vigas, and I. Johnson (2016). How to use t-sne effectively. *Distill*. `http://distill.pub/2016/misread-tsne` Accessed: 2018-07-20.

Wolpert, D. and W. Macready (1997). No free lunch theorems for optimizations. *IEEE Transactions on Evolutionary Computation 1*(1), 67–82.

Yeung, K. and W. Ruzzo (2001). Principal component analysis for clustering gene expression data. *Bioinformatics 17*, 763–774.

Zhong, S. and J. Ghosh (2003). A unified framework for model-based clustering. *Journal of machine learning research 4*, 1001–1037.